Edge auditing centers to improve blockchain-based reputation system for smart cities

Christian Maekawa[1], **Carlos Senna**[2], **Edmundo Madeira**[1]

[1]Instituto de Computação – Universidade Estadual de Campinas (Unicamp)
Campinas – SP – Brasil

[2]Instituto de Telecomunicações – Aveiro – Portugal

{c231867@dac.unicamp.br,cr.senna@av.it.pt, edmundo@ic.unicamp.br

Abstract. *A smart city must be able to acquire information, whether through sensors or people, and use this information in city management. In this scope, we present a reputation system based on blockchain that evaluates the information collected, whether collected by sensors or coming from people, in order to guarantee its veracity and activate measures when required. We use a distributed approach across clients at the edge of the network where we process information, which is aggregated in the cloud, maintaining global consistency. To demonstrate the usability, efficiency and scalability of our solution, we used real clients installed in the Aveiro Tech City Living Lab, the communications infrastructure of the city of Aveiro, Portugal, where our edge processing centers, installed in Road Side Units (RSUs), provide wireless communication with mobile elements and fixed connections with cloud-centric processing. In this scenario, we compare the response times of our distributed approach with a centralized solution to demonstrate the efficiency of our solution.*

Resumo. *Uma cidade inteligente deve ser capaz de adquirir informações, seja por meio de sensores ou pessoas, e usar essas informações na gestão da cidade. Neste escopo, apresentamos um sistema de reputação baseado em blockchain que avalia as informações coletadas, sejam elas coletadas por sensores ou vindas de pessoas, a fim de garantir sua veracidade e ativar medidas quando necessário. Utilizamos uma abordagem distribuída entre dispositivos na borda da rede onde processamos as informações, que são agregadas na nuvem, mantendo a consistência global. Para demonstrar a usabilidade, eficiência e escalabilidade da nossa solução, utilizamos dispositivos reais instalados no Aveiro Tech City Living Lab, a infraestrutura de comunicação da cidade de Aveiro, Portugal, onde nossos centros de processamento de borda, instalados em Road Side Units (RSUs), fornecem comunicação sem fio com elementos móveis e conexões fixas com processamento centrado na nuvem. Neste cenário, comparamos os tempos de resposta da nossa abordagem distribuída com uma solução centralizada para demonstrar a eficiência da nossa solução.*

1. Introduction

Smart cities are places with infrastructure capable of improving citizens' quality of life, controlling the efficiency of natural resources, promoting sustainable lifestyles, and efficient long-term waste management. An important part of this infrastructure is the most varied types of Internet of Things (IoT) clients spread throughout the city, which give it sensory capacity. This sensitive mesh allows the acquisition of information in real-time and facilitates city management, streamlining decision-making, strongly impacting mobility and quality of life [Zhou et al. 2020].

In this context, we understand that people can actively participate in this sensitive network, providing information about events such as traffic accidents, run-overs or signaling malfunctions that impact mobility in the city. To enable this participation, we are proposing a blockchain-based reputation system with audit centers that evaluate the veracity of event reports collected from people or sensors within the city [Firdaus et al. 2021]. In addition, the system allows voting on reports already posted, confirming their veracity or indicating possible distortions. In this way, our solution counts upvotes and downvotes, using them to update the reputation of informants in order to identify and block potential fraudsters from the system. Collected information is auditable across a fast in-memory database at the edge [Xue et al. 2023] for immediate evaluation, a relational database in the cloud for consolidated data management, and a blockchain ledger that immutably records reputation updates and event reports. Furthermore, clients such as cameras, radars, and lidars, installed in Road Side Units (RSU), can identify accidents of various natures and serve as a parameter to assess reports already posted to either reaffirm or deny them.

Reports about events tend to occur in a short space of time and are sent by people who are close to the location of the event, which indicates strong geographic regionalization. By exploring this important characteristic, we are proposing a decentralized architecture with distributed services at the edge of the network, capable of dealing with abrupt increases in demands when events occur. Our edge blockchain centers have a lightweight database and data structures where the reputation of informants is maintained for the period in which reports are posted, speeding up the assessment of their veracity. Furthermore, our edge centers are elastic and can allocate resources vertically, handling any flow of reports, evaluating their veracity and updating the reputation of informants locally during the period of high flow. Besides, we do not forget the general view of the city. When identifying a decrease in posts, the edge centers synchronize with the central control in the cloud where the entire blockchain is updated. This way, our solution responds to information peaks quickly and more complex tasks, such as transactions and information consensus, are handled in the cloud where higher capacity servers are located.

To demonstrate the usability, efficiency, and scalability of our solution, we chose a scenario with a city communications infrastructure, with emphasis on the Vehicular Ad Hoc Network (VANET). Our edge processing centers are installed in Road Side Units (RSUs) that provide wireless communication with mobile elements (vehicles, people, etc.) and fixed connections with cloud-centric processing. In this application scenario, we perform experiments in two aspects. In the first one, we demonstrate the correct functioning of the functionalities, the quality of the service, and the scalability of our solution in a laboratory testbed. Moreover, we evaluated the performance of our solution on real clients used in the Aveiro Tech City Living Lab (ATCLL) [Rito et al. 2023], a communication infrastructure installed in the city of Aveiro, Portugal.

The remainder of this paper is structured as follows: The related work is discussed in Section 2, the architecture of the reputation system proposed is described in Section 3, and the results of the evaluation carried out are discussed in Section 4. Finally, Section 5 concludes the paper and comments on future work.

2. Related work

The evolution of cooperative communication technologies such as Vehicle-to-everything (V2X) has contributed to solving problems in smart cities. By interconnecting vehicles with public infrastructure, sensors, computing nodes, pedestrians and other elements, it becomes a comprehensive information exchange platform [Zhou et al. 2020]. We are proposing a solution capable of using this extensive sensitive network to inform relevant occurrences within the city in order to improve the quality of life of users. However, it is important to guarantee the accuracy of reports and part of our solution measures and uses the reputation of participants, minimizing distortions. Traditional reputation systems generally separate the user interaction and the trust-building process, that is, the interaction is separated from the reputation [Hendrikx et al. 2015]. Our system innovates in such a way that the reputation is implicit in the interaction itself, it takes just one step, and when a user starts to interact they also present their reputation. Reliability is implicit in the message/interaction, not requiring knowledge about who provides the information.

Centralized reputation management solutions such as [Shrestha et al. 2018, Cui et al. 2019] face constraints such as failures at the central point and high network latency. Furthermore, they increase the chance of attacks precisely on the way to the centralized point. Our solution is based on Blockchain architectures where the use of distributed ledger technology allows the establishment of trust and decentralized relationships between untrusted entities, adding transparency to the information exchanged, guaranteeing the secure storage and sharing of data [Jain et al. 2021].

Our architecture uses a distributed approach across clients at the edge of the network, RSUs in our application scenario, where all information processing is carried out and which communicates with a central node in the cloud, maintaining global consistency of the entire system. With this strategy, we bypass the time spent by consensus algorithms and updating information on centralized blockchains. Along this path, there are some solutions [Xue et al. 2023]. Examples along these lines were presented in [Yang et al. 2019, Firdaus et al. 2021, Shen et al. 2022], but are not intended for reputation systems. Liu et al. [Liu et al. 2023] proposed a reputation management scheme based on an enhanced multi-weight and multi-source subjective logic algorithm, which fuses the direct and indirect opinion feedback of nodes through the subjective logic trust model considering factors such as event validity, familiarity, opportunity, and trajectory similarity. Vehicle reputation values are updated periodically and abnormal vehicles are identified using reputation thresholds. However the solution is aimed at improving security in VANET environments. Our strategy based on edge computing makes our system more efficient than centralized solutions as we demonstrate in our results. Furthermore, it scales more easily with the development of the city, whether with new regions or even in high-volume locations where multiple RSUs are needed to meet demand.

3. Proposed system architecture

In urban environments, a prominent event is expected to be noticed by most people around the event. Additionally, surrounding clients also send periodic notifications until the situation changes. This way, a large number of reports will be submitted in a short space of time, meaning the system must be able to handle a burst of reports. To meet these requirements, we are proposing a balanced three-tier architecture (Figure 1) composed of IoT/Users (Client layer - green area in Figure 1), edge blockchain centers installed on RSUs (Edge layer - blue part in Figure 1), and aggregated cloud services (Cloud layer - demarcated in orange in Figure 1).

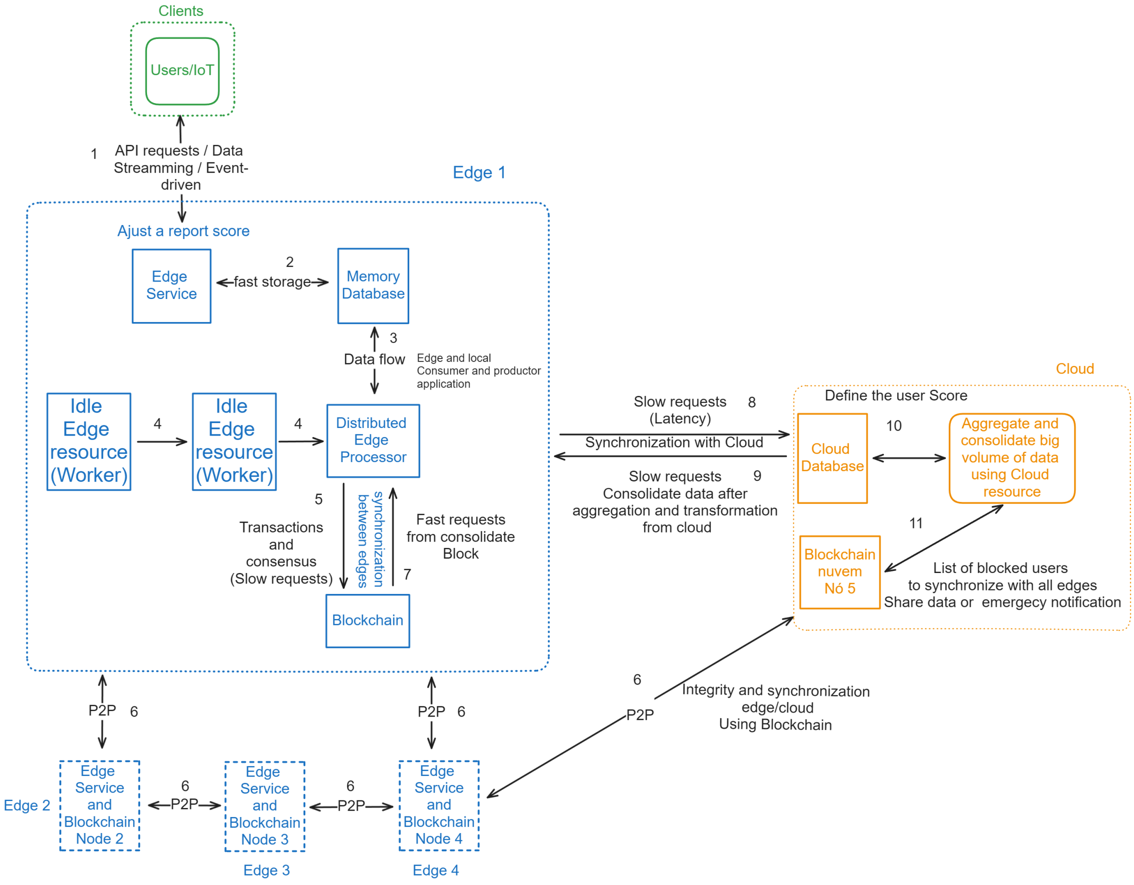

Clients
Users/IoT
1 API requests / Data Streamming / Event-driven
Edge 1
Ajust a report score
Edge Service
2
fast storage
Memory Database
3
Data flow
Edge and local Consumer and productor application
Idle Edge resource (Worker)
4
Idle Edge resource (Worker)
4
Distributed Edge Processor
5
Transactions and consensus (Slow requests)
synchronization between edges
Fast requests from consolidate Block
7
Blockchain
Slow requests (Latency) 8
Synchronization with Cloud
Slow requests 9
Consolidate data after aggregation and transformation from cloud
Cloud
Define the user Score
Cloud Database
10
Aggregate and consolidate big volume of data using Cloud resource
11
Blockchain nuvem Nó 5
List of blocked users to synchronize with all edges
Share data or emergecy notification
P2P 6
P2P 6
6
P2P
Integrity and synchronization edge/cloud Using Blockchain
Edge 2
Edge Service and Blockchain Node 2
6
P2P
Edge Service and Blockchain Node 3
Edge 3
6
P2P
Edge Service and Blockchain Node 4
Edge 4

Figure 1. Macro architecture.

Our small processing and memory cache centers installed in RSUs at the edge of the network are located close to the clients that generate the reports. In these edge centers, we distribute the blockchain, maintain the reports, and evaluate the reputation of the reporters during peak activity. After processing the peak volume of reports, the edge centers update the central control (cloud) which consolidates the information and activates cloud services to synchronize the consensus information across all nodes.

In the macro architecture shown in Figure 1, we have numbered the arrows that indicate the connections between the various components of our architecture. We use this numbering in the overview of the flow of operations that we detail below. IoT/Users send reports to the edge hub closest to their location (1); the local edge service receives the reports and stores them in the local database (2); the local database notifies the distributed edge processor (3) that runs the producer/consumer application (4) and distributes them to the local blockchain (5 and 7) to maintain reports and evaluate the reputation of the report sources during peak activity. After processing the peak volume of reports, the edge hubs update the central control in the cloud (8) where the aggregation and consolidation of the reports, votes, and reputation occurs. After consolidation, the cloud node synchronize the consensus information to the distributed nodes in the edge hubs (6 and 11).

The Client layer adapts to different clients by providing a user interface for reporting and a communication module for transmission. For IoT clients, the same components are present, one for collecting reports and another for transmission to the associated RSU. This layer handles data heterogeneity with drivers for multiple communication technologies (ITS-G5, WiFi, Ethernet) and supports both human and IoT device users, making reporting capabilities transparent for management at the edge or in the cloud.

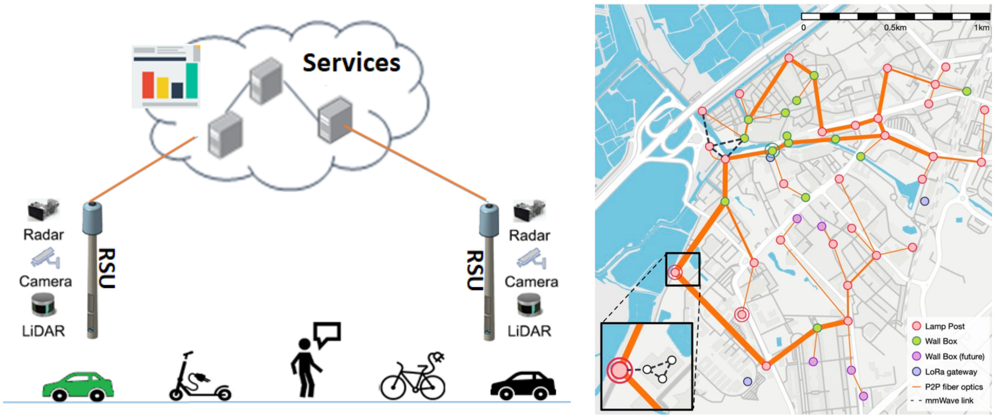
The Edge layer receives the standardized report, verifies its authenticity, and stores it locally. It manages reporting and reputations using its local blockchain node, mirroring the cloud service. It also has a broker, queue manager, and local database to handle load increases. The Edge layer evaluates reports as event announcements, upvotes, or downvotes. The first report is considered the announcement, and subsequent reports are evaluated as confirming or contradicting the initial information. A Beta probability density function calculates a report's score (0 to 1, starting at 0.5), adjusted by Bayes' theorem as positive or negative votes arrive, influencing the confidence score. The report information, including the score, is stored in the edge's in-memory database and sent to the cloud for global management. Our architecture scales by decoupling task generation and processing and processes tasks in the background without impacting the end-user. The broker also adds fault tolerance. Reports in the local in-memory database move to a persistent cloud database via an event-driven architecture (EDA), which schedules periodic synchronization. This process uses a Producer-Consumer system where tasks are queued and activated. The report is then converted into a blockchain transaction for simultaneous sharing among all nodes, ensuring consensual information.

Finally, the Cloud layer processes all report data for users, generates a user-oriented score, and updates all the edge nodes of blockchain. This can include blocking or restricting users with updated reputation scores.

4. Results

Most environments under the umbrella of IoT make use of numerous intercommunication clients, each with their own attributes and limitations, stimulating the need for high adaptation of both the network and the applications within it to face the arising challenges.

One of these cases is the scenario of a smart city where several data collection units composed of sensors, with low computational capacity, low storage capacity and connected to the city's communication infrastructure through fixed or wireless connections, can be placed in a city, recording information about its surroundings. This is the application scenario we chose to evaluate our solution for collecting reports and analyzing the reputation of those who made them. Figure 2a illustrates our scenario made up of mobile elements such as vehicles, bikes, and people, from which reports come. The mobile elements, through wireless communication technologies such as WiFi and ITS-G5, send reports to edge clients hosted by RSUs installed on smart poles where they are processed immediately. Additionally, IoT clients such as cameras, radars, and lidars installed on RSUs send reports to our edge component that help evaluate the reputation of informants. In this way, we distribute regional report processing centers that add agility and scalability to our solution. However, it is necessary to have an overview of the city. To this end,



(a) Scenario elements.

(b) ATCLL infrastructure.

Figure 2. Application scenario.

the RSUs are connected to the fixed mesh spread across the city, using it to send regional information to be collected in the cloud and redistributed to the edge, maintaining the consistency of the entire incident reporting and reputation measurement system.

An example of this infrastructure is the Aveiro Tech City Living Lab (ATCLL) installed in the Aveiro city, Portugal [Rito et al. 2023]. The ATCLL comprises a large number of IoT clients with communication, sensing and computing capabilities, constituting a smart city infrastructure. As can be seen in Figure 2b, city access communication points (RSUs) are connected via fiber and millimeter wave (mmWave) links and integrate mobile nodes (OBUs) via WiFi, ITS-G5, C-V2X, 5G and/or LoRa links. All of these points combine and interconnect a set of sensors, such as mobility sensors (radars, LiDARs and video cameras) and environmental sensors. The map shows how the ATCLL's communications infrastructure is distributed in the central and touristic part of Aveiro. It is the place with the greatest potential number of reports and very interesting for testing the efficiency of our solution.

We carried out experiments in two aspects. In the first of them, our objective was to demonstrate the correct functioning of the functionalities, the quality of the service, and the scalability of our solution. The main points of interest are discussed in Section 4.1 and the results presented in the next sections. In a second aspect, we evaluate the performance of our solution on clients installed in ATCLL in conditions similar to those found in RSUs installed in the city of Aveiro. We present and analyze the results of this second aspect in Section 4.4. Our testing strategy was focused on measuring the execution time of these functionalities: Creating a user; Retrieving the wallet code; Creating a report and Making a vote.

4.1. Laboratorial testbed and basic functionalities

This study investigates the performance of an edge computing framework under different deployment scenarios. Tests were conducted on a local container-based architecture as well as in the Aveiro infrastructure, which includes client, edge, and cloud components. The local testing focused on debugging the cloud environment, aiming to assess the framework's performance without the latency issues between the client and cloud networks. Since the cloud module shared the same communication module as the edge, this local testing allowed us to evaluate the framework without network interference. The experiments were carried out on the researcher's personal computer as well as a PC Minisforum. Additionally, tests were performed on the client-edge-cloud architecture where we allocate seven clients, a PC Minisforum as Cloud node, four APUs as edge nodes, and one VM with four coitainers as clients simulating regional interaction with edge. The test consists of simulating simultaneous node access with at least one user interaction until fifty users for each edge region.

Single interaction baseline (Part 1): A single interaction performed with the system. This provides a baseline measurement of the time taken for each functionality. The test is run on both a personal computer and a simulated Cloud environment for comparison. Note that Cloud with user interaction is similar architecture than a user with edge, the difference is that the node does not have an SQL database.

Concurrent edge interaction (Part 2):

The proposed system operates across a three-tiered distributed architecture. This architecture consists of four client nodes, four edge nodes, and a single cloud node. Each edge and cloud node has five internal blockchain containers encompassing the Practical Byzantine Fault Tolerance (PBFT) an algorithm of consensus engine, transaction settings, validator logic, and a REST API. Each client node is equipped with a container for statistical analysis. All nodes within the system, regardless of their tier (client, edge, or cloud), are uniquely identified through their IP addresses.

Four containers interact with another individual edge of the system simultaneously. Each edge is then tested sequentially with 1000 interactions. This measures the system's performance under concurrent load from multiple sources. The experimental process follows the flow shown in Figure 3.

Each client deploys 1000 calls of key functions for count and measure analysis. The Cloud receives the database requests from the edge, meanwhile, the client interacts with the edge. The cloud to communicate with cloud clients needs a step interaction with the edge. So more time is expected for cloud requests.

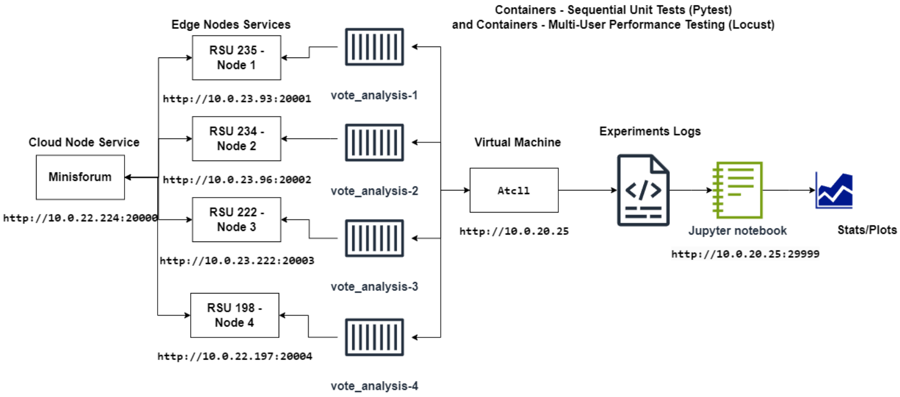


Figure 3. Concurrent edge interaction and IoT client evaluation.

IoT client evaluation (Part 3):

The Locust library is used to simulate a IoT client evaluation environment (Figure 3). The simulation consists of 50 simulated users interacting randomly and concurrently with four nodes. This tests the system's performance under a more realistic and unpredictable load.

To simulate the application scenario, we set up a testbed with seven clients, whose technical specifications are in Table 1.

Table 1. Computational resources description.

	PC Minisforum	VM	Desktop	APUs
Processor	AMD Ryzen 9	Xeon 4215	Intel i7 9700	AMD GX-412TC
Cores	16	4	8	4
Memory	16GB	8GB	48GB	4GB
Storage	256GB	64GB	2TB	256GB
O. S.	Ubuntu 22	Ubuntu 20	WSL 2 Windows 11	Debian 11

The core development of the system relied on Python and its associated libraries. We utilized pytest for simulating client behavior in unit tests, and Locust for simulating realistic user load. Client interactions were managed through a FastAPI API. For asynchronous task execution and message queuing, we implemented Celery with Redis and RabbitMQ as brokers. Blockchain functionality was tested using Hyperledger Sawtooth, employing the (PBFT) consensus mechanism, and was chosen for its open-source nature. Finally, a custom Python class was created to model and simulate the variations of a beta distribution for our Bayesian simulation, and the results were stored in Redis. For statistical analysis, we used the Scipy and Statsmodels libraries, and for generating visualizations, we employed Seaborn.

4.2. Single interaction baseline

The initial test evaluated a baseline architecture where the testbed facilitated direct communication between the client and the cloud node. This experiment demonstrated that even within the containerized environment, interaction time with the RDBMS was significantly higher (2500%) than with the in-memory database, even when testing a single instance. This substantial latency difference highlights a critical concern given the system's need to accommodate burst requests and maintain low latency. Therefore, deploying a dedicated database per node is impractical for two primary reasons: the performance penalty would be unacceptable, and it would lead to significantly higher expenses for edge management. As an alternative, direct communication with the in-memory database proved remarkably efficient, approaching 0.01 milliseconds.

Direct interaction with the blockchain ledger is also not recommended. The consensus algorithm introduces significant overhead, potentially exceeding 10 seconds per operation. To address these constraints, our middleware handle data interactions, providing a low-latency interface between the application and both the in-memory database and the blockchain.

To further validate our initial findings on the performance difference, we conducted a t-test with an additional 753 samples, the overall behavior of which is shown in Figure 4 and some details are shown in Table 2. Our null hypothesis (H_0) stated that there is no significant difference in interaction times between the in-memory database and the RDBMS. The t-test yielded a T-statistic of 292.58 and a p-value of 0.00, which is far

Table 2. Operation time of container interaction on desktop.

Operation	count	mean	median	std	min	max
user	753.0	0.24	0.24	0.02	0.01	0.37
report	1.0	0.01	0.01	0.00	0.01	0.01
reputation/vote	751.0	0.01	0.01	0.0	0.01	0.04
wallet/test2	1.0	0.0	0.0	0.00	0.0	0.0

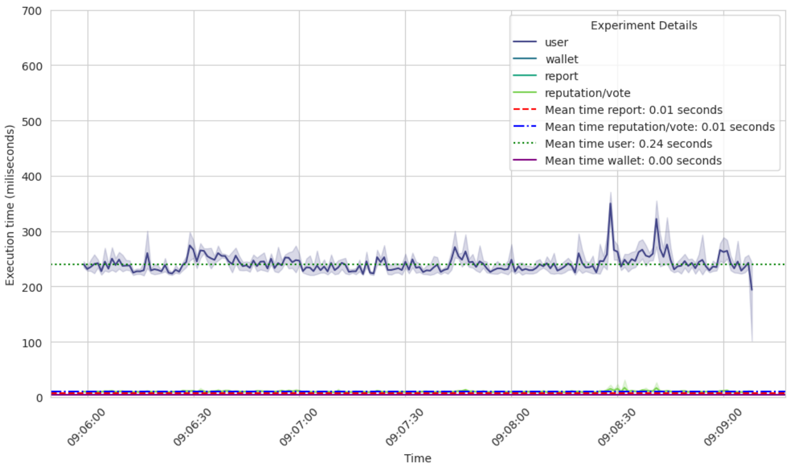


Figure 4. Sequential interaction in personal desktop.

below our significance level of 0.01. Therefore, we reject the null hypothesis, confirming a statistically significant difference in performance between the two variables.

To validate the primary findings, the experiment was replicated using a Minisforum PC minitower, which was connected to the Aveiro's Network. Experimental data from multiple trials are presented in Table 3.

Table 3. Operation time of container interaction on minitower.

Operation	count	mean	median	std	min	max
user	1000.0	0.23	0.23	0.02	0.22	0.67
report	487.0	0.01	0.01	0.00	0.01	0.02
reputation/vote	250.0	0.01	0.01	0.00	0.01	0.02

The overall performance patterns observed on the Minisforum, similar to those observed in the previous experiment, were consistent with those seen on personal computers. This consistency suggests that the core application performance scales uniformly across different hardware platforms, even though external factors on the shared Minisforum system might introduce some variability. Additionally, our separate analysis of database performance showed a significant difference between the in-memory database and the RDBMS (T-statistic = 1589.4, $p < 0.01$), highlighting the potential impact of database choice on overall application performance.

The number of trials in the experiment varied, as the primary goal was to test specific code conditions. This included scenarios where users attempted to create duplicate names or submit identical votes, which did not need a fixed number of trials per test case.

4.3. Concurrent edge interaction

The second experiment evaluated performance in a distributed scenario, simulating regional nodes handling client load. The experiment aimed to demonstrate potential performance gains compared to a centralized approach. Specifically, the experiment involved transmitting reports from a virtual machine (VM) representing a client to an edge node, and cloud node. This VM was accessed remotely over the Internet from the analysis notebook and was physically isolated, lacking a direct connection to the report-generating device. Our distributed solution was deployed across three hardware: a Minisforum PC minitower serving as the cloud node, the RSU (APU 93) acting as the edge node, and the VM hosting the client to simulate interactions with both edge and cloud components. As part of smart city infrastructure, the RSU (edge node) and the cloud VM were interconnected via a Gigabit wired network. In this test, it was checked the performance of the concurrent experiment. Table 4 presents the result for only one node, and the behavior of the test is available on Figure 5.

Table 4. Testing 1000 interactions with a client, edge, and cloud.

Omnimo	Operation	Count	Mean (s)	Std (s)
APU 93	user	1001.0	1.18	0.02
	reputation/vote	1000.0	0.07	0.01
	report	1.0	0.04	0.00
	wallet	1.0	0.02	0.00

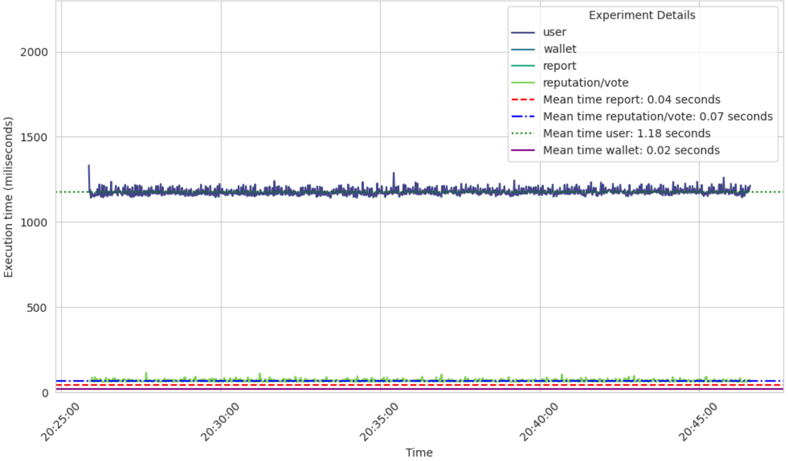


Figure 5. Performance over 1000 interactions with a client, edge, and cloud.

We investigated the impact of concurrency on database access performance using the same APU 93 edge in concurrency experiments, as Figure 6 and Table 5. We compared the time taken to create a user on the database under two conditions: a single client accessing the database and four clients concurrently accessing the database, and we measured the time to vote that uses the in-memory database. The Shapiro-Wilk tests indicated that the performance data for both the 1-edge (W = 0.894, $p < 0.001$) and 4-edge (W = 0.986, $p < 0.001$) conditions were not normally distributed. Therefore, we used the Mann-Whitney U test to compare the distributions of performance times. The results showed a statistically significant difference between the two conditions

(U = 706864, $p < 0.001$). This difference is likely attributable to the 3-point communication path required for database access, where clients communicate through an edge server before reaching the cloud database. The cloud database experiences simultaneous read and write operations from multiple clients, leading to more variable performance times compared to the edge, which uses distributed in-memory databases for each client.

In contrast, we evaluated the impact of concurrency on memory-based database access operations using the same APU 93 edge. Again, Shapiro-Wilk tests indicated that neither the 1-edge (W = 0.841, $p < 0.001$) nor the 4-edge (W = 0.868, $p < 0.001$) performance data were normally distributed. Using the Mann-Whitney U test, we found no statistically significant difference in performance between the two conditions (U = 499492.5, p = 0.969).

While concurrency significantly impacted the time taken to create a user on the cloud database, we found no statistically significant difference in the time taken for memory-based database access when using four clients concurrently compared to a single edge. This indicates that concurrency does not significantly affect the performance of the memory-based database access operation.

Table 5. Statistics for 4 clients concurrency.

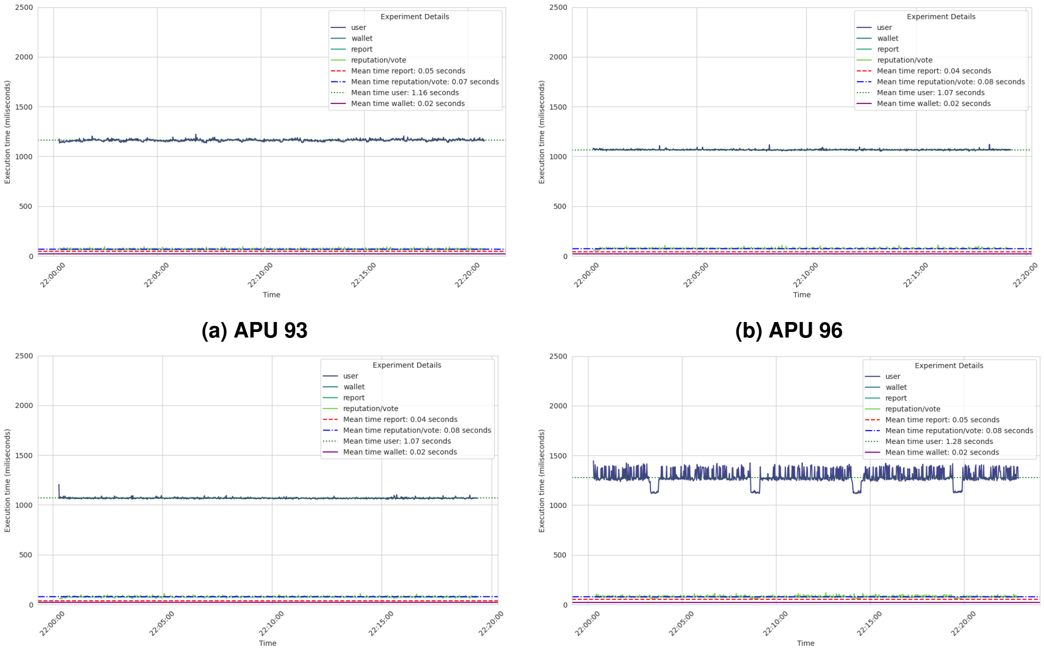
Omnimo	Operation	Count	Mean (s)	Std (s)
APU 93	user	1001.0	1.16	0.01
	reputation/vote	1000.0	0.07	0.01
	report	1.0	0.05	0.00
APU 96	user	1001.0	1.07	0.01
	reputation/vote	1000.0	0.08	0.01
	report	1.0	0.04	0.00
APU 222	user	1001.0	1.07	0.01
	reputation/vote	1000.0	0.08	0.01
	report	1.0	0.04	0.00
APU 197	user	1001.0	1.28	0.07
	reputation/vote	1000.0	0.08	0.01
	report	1.0	0.05	0.00

4.4. IoT client evaluation

Our solution is designed to work in conjunction with smart city communication infrastructures such as the ATCLL installed in Aveiro, Portugal. This type of communication infrastructure is ideal for installing and using our reputation system. The mobile nodes (on board units - OBUs) would receive the client part of our software through which users can submit their reports. Our edge centers would be installed in the RSUs and would quickly address reports sent by mobile nodes within their communication area. The global module would be installed in the ATCLL datacenter, which has a fiber connection to all RSUs.

This Section presents a test scenario that is closer to what happens in the real world with multiple clients posting reports at random times and in random quantities. To achieve this, we set up a simulation with multiple users accessing a single node with different behaviors.

In this experiment, we observed a higher degree of variability in the results due to the increased number of interacting variables. Specifically, the simultaneous access



(a) APU 93

(b) APU 96

(c) APU 222

(d) APU 197

Figure 6. Comparing sequential performance.

by approximately 50 users, coupled with operations occurring at different times, introduced more variability in the measured performance. Despite this increased variability, we observe a consistent pattern shown in Figure 8, indicating a significant impact on performance due to hardware limitations and networking bottlenecks under heavy load.

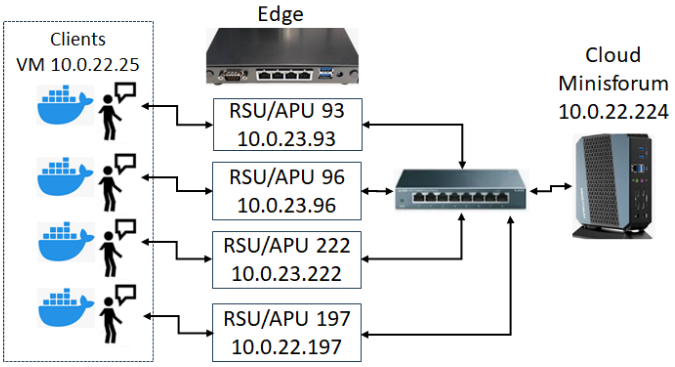


Figure 7. APUs testbed.

To evaluate the performance of our solution on real clients, we set up a testbed with ATCLL RSU edges, the PC Engines APU boxes. The configuration and functionalities of this testing environment, as shown in Figure 7, are the same as the laboratory testing environment. As well as the specification of the computational resources used previously described in Table 1. In this way, we were able to evaluate the performance of the edges and use our results to design solutions that are compatible with the reporting flow of each region of the city. To do that, we repeated the same experiments performed in Section 4.1. The Figure 8 summarizes the performance of the four RSUs and presents details about the behavior of RSUs 93 and 222.

Omnimo	Operation	Count	Mean (s)	Std (s)
APU 93	user	935.0	12.62	8.69
	reputation/vote	906.0	1.13	0.21
	report	906.0	1.93	2.83
	wallet	907.0	14.80	12.40
APU 96	user	980.0	14.35	9.07
	reputation/vote	936.0	0.99	1.01
	report	938.0	1.70	2.73
	wallet	955.0	12.19	10.95
APU 222	user	711.0	17.12	9.38
	reputation/vote	638.0	2.36	3.55
	report	650.0	3.18	4.33
	wallet	698.0	18.07	12.20
APU 197	user	710.0	16.56	9.27
	reputation/vote	630.0	1.96	2.08
	report	635.0	2.88	3.58
	wallet	683.0	20.01	11.50

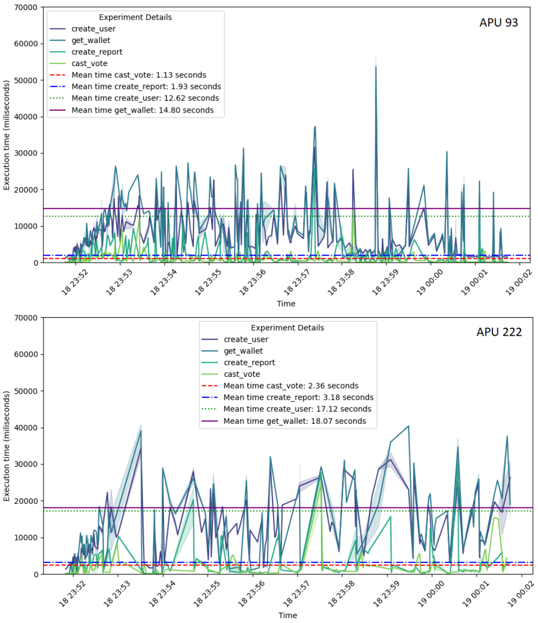


Figure 8. Concurrency and parallel performance.

The average times shown in the table indicate that RSUs 93 and 96 have similar performance with better times than the other two. This difference is clearer in the details of the individual graphs of RSUs 93 and 222. This heterogeneous performance is common in real installations composed of different production batches. However, it is important to be aware of the differences in order to design configurations that are more appropriate for the information flow in each region of the city.

The low times for reporting and voting operations prove the efficiency of our architecture in organizing information at the edge. Synchronization to save edge reports and standardize poster reputations during idle times, done between the edge and cloud points, has no impact on the average response time of the system. Our results also demonstrate that our architecture is scalable, supporting any volume of posts. Furthermore, if a region has a significant volume of posts due to the large number of vehicles and people, it is possible to install more edge instances in the same RSU, easily handling these regional requirements.

5. Conclusion and future work

In the paper we present a reputation system that manages and evaluates the quality of information disseminated by citizens in a smart city. Furthermore, our solution allows the city's sensitive network (sensors of various types) to report incidents or vote on people's reports, supporting reputation validation to ensure veracity or indicating potentially false reports. For its implementation, we proposed the use of a blockchain system to maintain the history of reports, reputations and other important information for the solution. To overcome the known slowness of consensus algorithms, our solution operates distributed on the edge, which in our application scenario are RSUs distributed throughout the city. As our results showed, our solution is faster and more scalable than cloud-centric solutions. In future work we will integrate our solution into the ATCLL, where it will be used in conjunction with travel planners (Waze), citizen support systems, and administration support systems, to contribute as a tool for city management.

Acknowledgements

This work is funded by PRR – Plano de Recuperação e Resiliência and by the NextGenerationEU funds at Universidade de Aveiro, through the scope of the Agenda for Business Innovation "ATT – Agenda Mobilizadora Acelerar e Transformar o Turismo" (Project no. 47 with the application C645192610-00000060). This research was partially sponsored by CNPq grant #405940/2022-0 and CAPES grant #88887.954253/2024-00.

References

Cui, J., Zhang, X., Zhong, H., Ying, Z., and Liu, L. (2019). Rsma: Reputation system-based lightweight message authentication framework and protocol for 5g-enabled vehicular networks. *IEEE Internet of Things Journal*, 6(4):6417–6428.

Firdaus, M., Rahmadika, S., and Rhee, K.-H. (2021). Decentralized trusted data sharing management on internet of vehicle edge computing (iovec) networks using consortium blockchain. *Sensors*, 21(7).

Hendrikx, F., Bubendorfer, K., and Chard, R. (2015). Reputation systems: A survey and taxonomy. *Journal of Parallel and Distributed Computing*.

Jain, S., Ahuja, N. J., Srikanth, P., Bhadane, K. V., Nagaiah, B., Kumar, A., and Konstantinou, C. (2021). Blockchain and autonomous vehicles: Recent advances and future directions. *IEEE Access*, 9:130264–130328.

Liu, Q., Gong, J., and Liu, Q. (2023). Blockchain-assisted reputation management scheme for internet of vehicles. *Sensors*, 23(10).

Rito, P., Almeida, A., Figueiredo, A., Gomes, C., Teixeira, P., Rosmaninho, R., Lopes, R., Dias, D., Vítor, G., Perna, G., Silva, M., Senna, C., Raposo, D., Luís, M., Sargento, S., Oliveira, A., and de Carvalho, N. B. (2023). Aveiro tech city living lab: A communication, sensing, and computing platform for city environments. *IEEE Internet of Things Journal*, 10(15):13489–13510.

Shen, M., Lu, H., Wang, F., Liu, H., and Zhu, L. (2022). Secure and efficient blockchain-assisted authentication for edge-integrated internet-of-vehicles. *IEEE Transactions on Vehicular Technology*, 71(11):12250–12263.

Shrestha, R., Bajracharya, R., and Nam, S. Y. (2018). Centralized approach for trustworthy message dissemination in vanet. In *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*, pages 1–5.

Xue, H., Chen, D., Zhang, N., Dai, H.-N., and Yu, K. (2023). Integration of blockchain and edge computing in internet of things: A survey. *Future Generation Computer Systems*, 144:307–326.

Yang, Z., Yang, K., Lei, L., Zheng, K., and Leung, V. C. M. (2019). Blockchain-based decentralized trust management in vehicular networks. *IEEE Internet of Things Journal*, 6(2):1495–1505.

Zhou, H., Xu, W., Chen, J., and Wang, W. (2020). Evolutionary v2x technologies toward the internet of vehicles: Challenges and opportunities. *Proceedings of the IEEE*, 108(2):308–323.