

Blockchain Platform Performance Assessment for Livestock Traceability

Bernardo J. R. Figueiredo*, Marco P. M. Ferreira*, Marco Cova*, Francisco Faria†, Miguel L. Pardal†

* VOID Labs, VOID Software S.A., Leiria, Portugal

Email: {bernardo.figueiredo, marco.ferreira, marco.cova}@voidsoftware.com

† INESC-ID, Instituto Superior Técnico, Universidade de Lisboa, Portugal

Email: {franciscofaria00, miguel.pardal}@tecnico.ulisboa.pt

Abstract—Ensuring transparency and trust in the livestock supply chain is a significant business challenge in agro-food due to the involvement of multiple stakeholders with often conflicting interests. The complex life cycle of cattle requires reliable data sharing with the ability to verify the authenticity of records. To address this, we developed a livestock traceability application on the Hyperledger Fabric platform, which models livestock as assets and captures their life cycles through event records. This system provides a decentralized, shared ‘single source of truth’ for farmers, transporters, veterinarians, and regulators. As performance is paramount for adoption, we conducted an evaluation of the platform under various configurations. Our test suite comprises 10 runs, each executed on 5 independent peer instances, evaluating 100 assets per run. We compare two specific configurations: Configuration A, featuring single-transaction blocks of 99 MB; and Configuration B, using 49 MB multi-transaction blocks with 10 transactions per block. Both configurations were tested across 10 runs to thoroughly assess their impact on transaction throughput and overall system scalability. The study also included long-duration runs, seven times the length of the main runs, to evaluate sustained performance. Furthermore, we investigated how blocks are filled by testing configurations with a 49 MB block size and a 1000 transaction limit. The results in throughput, latency, storage efficiency, and memory usage show that Configuration B had superior performance in throughput and success rate, exhibiting lower and more consistent latencies. However, Configuration B also showed more pronounced storage growth and memory usage anomalies. Our analysis provides insights and actionable recommendations for optimizing smart contract payloads, implementing compression techniques, and rebalancing the load on orderer nodes, to achieve a sustainable and scalable solution.

Index Terms—Blockchain, Hyperledger Fabric, Performance Evaluation, Livestock Traceability, Supply Chain Management.

I. INTRODUCTION

Traditional livestock supply chains are characterized by fragmented record-keeping and limited interoperability between stakeholders, including farmers, transporters, veterinarians, processors, and regulators [1]. Each participant typically

This work was supported by national funds through Fundação para a Ciência e a Tecnologia, I.P. (FCT) under projects UID/50021/2025 and UID/PRR/50021/2025 (INESC-ID) and by Blockchain.PT - Decentralize Portugal with Blockchain Agenda, (Project no 51), WP 1: Agriculture and Agro-food, Call no 02/C05-i01.01/2022, funded by the Portuguese Recovery and Resilience Program (PPR), The Portuguese Republic and The European Union (EU) under the framework of Next Generation EU Program.

maintains their own siloed databases, leading to inconsistencies, data duplication, and difficulties in verifying the provenance and integrity of livestock records. These challenges are further compounded by regulatory requirements for food safety and traceability, such as those mandated by the European Union [2], which demand accurate and auditable tracking of animals throughout their lifecycle.

Blockchain technology offers a promising solution to these challenges by providing a decentralized, tamper-evident ledger that enables secure and transparent data sharing among all participants. Hyperledger Fabric [3], in particular, supports permissioned networks, fine-grained access control, and modular consensus mechanisms, making it well-suited for enterprise applications in regulated environments. By leveraging Fabric, our platform keeps track of each animal as a digital asset and records key lifecycle events as immutable records on the blockchain.

However, a platform’s real-world viability depends not only on its functionality but also on its ability to handle a high volume of transactions with low latency and consistent stability. In practice, verifying that the platform can process transactions efficiently and operate reliably is necessary for its adoption as a viable business solution.

This article focuses on three key performance aspects: transaction throughput, system stability under load, and resource utilization patterns. Our testing methodology includes controlled stress scenarios and long-running stability tests to identify performance bottlenecks and establish baseline metrics for CPU and memory consumption. The results provide insights and actionable recommendations for optimizing the platform, paving the way for a production-ready solution.

II. EVALUATION

Our evaluation of the livestock traceability platform was conducted using a methodology designed to simulate production-like conditions. The process is structured as follows:

- **Configuration and Setup:** Definition of the hardware and software environment, network topology, and specific component configurations used throughout the study.
- **Success Criteria and Metrics:** Detailed explanation of the key metrics and criteria for assessing platform

performance, including transaction throughput, latency, and resource utilization.

- **Performance Testing Scenarios:** Establishment of the specific performance testing scenarios and configurations, including the controlled stress tests designed to evaluate system scalability under various operational loads.
- **Stress Testing Results:** Presentation of the empirical data collected from our performance and stress tests, providing an objective analysis of transaction throughput, latency, and resource consumption for each configuration.
- **Discussion:** Finally, interpretation of the results. Explanation of key trade-offs and performance characteristics, as well as recommendations for optimizing the platform.

A. Configuration and Setup

The foundation of our performance analysis is a testing methodology, designed to simulate production-like conditions and generate reproducible results. This configuration and setup is based on real data that the authors had access from an ongoing project. Thus, the configuration resembles the worst expected scenarios likely to occur on a real-world use case of livestock traceability. Our test environment was configured as follows:

- **Network Configuration:** The network is deployed using Hyperledger Fabric and consists of six organizations, with each organization running a single peer node. The organizations are: Organization Blue, Organization Orange, Organization Red 1, Organization Red 2, Organization Red 3, and VoidSoftware. The ordering service is composed of three orderer nodes. The network was tested under two distinct transaction configurations to evaluate performance:
 - **Configuration A:** A single-transaction block with a 99 MB block size.
 - **Configuration B:** A multi-transaction block with a 49 MB block size, with ten transactions per block.
- **Hardware specifications:** All nodes in the network share the following hardware specifications: The environment is powered by an Ampere Altra Max processor with 128 cores at 3.0GHz, 512 GB of DDR5 RAM, and a 4 TB SSD.
- **Software stack and versions:** The stack includes Hyperledger Fabric v2.5.13, CouchDB v3.4.2, Hyperledger Fabric CA v1.5.15, and K3s v1.32 for orchestration.

Two configurations were specifically chosen to evaluate the platform's behavior under distinct operational scenarios.

From a technical standpoint, the two configurations were designed to probe different aspects of the Hyperledger Fabric architecture. Configuration A, with its large block size of 99 MB and a single transaction per block, was used to stress the system's ability to handle high-payload, low-throughput workloads. This setup simulates the transfer of large, single data packets, such as comprehensive veterinary records or sensor data archives, and specifically tests the overhead associated with block finalization and distribution

for massive files. The focus here is on the network's capacity to handle a single, bandwidth-intensive event. In contrast, Configuration B, featuring a smaller 49 MB block size but with ten transactions per block, was selected to assess the platform's performance under a more typical, high-throughput environment. This configuration more accurately represents the frequent, smaller data entries associated with a livestock's lifecycle, like daily feed logs or routine movement events. Technically, this setup tests the efficiency of the ordering service's ability to aggregate and serialize multiple concurrent transactions, as well as the peer nodes' capacity for efficient state updates and commitment of multi-transaction blocks. On a side note we selected two significant distinct values for the block size to effectively test and demonstrate the platform's robustness and scalability. The 99 MB block size is the largest value supported in Hyperledger Fabric networks, and the 49 MB as approximately half of this maximum.

From a business and operational perspective, these configurations represent a trade-off between responsiveness and infrastructure cost management. Configuration A's approach of bundling a single, massive transaction per block is more suited for asynchronous tasks and large-scale data imports, such as a regulator performing a daily data sync. However, its low throughput can lead to significant transaction latency for end-users, potentially impacting the usability of a real-time tracking application. For a farmer or veterinarian, a slow system could hinder the immediate logging of an event, which could delay the entire supply chain. Conversely, Configuration B's higher transaction throughput and lower latency could provide a more immediate and responsive user experience. The possible downside could be a higher storage consumption per transaction, which directly translates to increased infrastructure costs over time. A production-ready solution would require a careful balance of these factors to provide both a seamless user experience and a financially viable long-term solution.

B. Success Criteria and Metrics

The success criteria for this evaluation were established to assess the platform's performance, reliability, and scalability. These metrics define a clear, quantifiable benchmark to validate the platform's technical capabilities and readiness for commercial deployment.

- **Reliability:** A 99.9% transaction success rate under normal conditions.
- **Performance:** Average transaction time ≤ 2000 ms under normal conditions.
- **Resource Efficiency:** CPU and memory utilization remain below 80% sustained usage per node.
- **Storage Management:** Storage growth remains proportional (ideally linear) to transaction volume.
- **System Stability:** Minimal variance in resource usage and latency across all peer and orderer nodes.

The criteria were derived from established benchmarks in distributed systems and adapted to the specific demands of a permissioned blockchain within a regulated supply chain. The

goal was to establish a quantifiable baseline that would serve as a clear indicator of production readiness. [4]–[7]

The reliability criteria, a 99.9% transaction success rate, functions as a critical service-level objective (SLO). This high threshold is a direct measure of the network’s fault tolerance and its ability to maintain data integrity under both typical and stress-induced load conditions [4]. Similarly, the performance metric of average transaction latency of ≤ 2000 ms is crucial for user experience. This value strikes a balance between the inherent delays of the distributed consensus process which includes transaction endorsement, ordering, and validation and the need for a responsive system capable of providing timely feedback for commercial applications [5], [6].

From an operational and financial perspective, resource efficiency and storage management are paramount. The constraint of CPU and memory utilization remaining below 80% provides a necessary buffer for handling unexpected load spikes, preventing the system from reaching its performance bottleneck and ensuring predictable behavior. Concurrently, the requirement for proportional storage growth is essential for long-term viability, as it guarantees that operational costs scale linearly with transaction volume, thereby preventing the system from becoming financially unsustainable over time [7].

Finally, system stability, measured by minimal variance in resource usage and latency, is a direct indicator of the overall health of the distributed architecture. Low variance suggests an effective load-balancing mechanism and the absence of fundamental issues in the consensus or communication protocols. Conversely, high variance would point to potential bottlenecks, network instability, or an uneven distribution of workload, all of which could undermine the system’s reliability and predictability [6].

C. Performance Testing Scenarios

The performance and stress testing framework was designed to assess the blockchain platform’s behavior under controlled, production-like conditions. The test suite includes 10 distinct runs, each executed across 5 independent instances, with 100 trackable items per run. Both configurations were tested across 10 runs to evaluate how each setup impacts transaction throughput and overall system scalability. The following subsections present specific runs to highlight important details that demonstrate the impact of each configuration.

1) *Performance - Configuration A - Run#2*: Figure 1 presents the performance profile for Configuration A, Test 2. Over this scenario, 53730 transactions were processed across 5 instances. The results show an average transaction time of 451.73 ms with a notable success rate of 96.31%. This success rate, a significant deviation from a near-perfect rate, suggests potential issues during the transaction validation or commit stages.

An analysis of component-level metrics highlights both systemic and localized behaviors. The overall network storage increased by 6.2 GB, which is proportional to the total ledger activity. At the peer level, however, inconsistencies emerged. While the storage increase was consistent across

Instances	Total Trackables	Total Transaction	Avg. Transaction Time (ms)	Success Rate (%)
5	500	53,730	451,73	96.31

Component	Type	Initial Size (GB)	Final Size (GB)	Storage Increase (GB)	Avg. CPU (millicores)	Avg. Memory (Mi)
Network	N/A	0.431	6.6	6.2	N/A	N/A
VOID	Peer	0.010	0.677	0.667	1474	690
Red 1	Peer	0.093	0.698	0.688	1520	693
Red 2	Peer	0.093	0.700	0.690	1508	703
Red 3	Peer	0.093	0.701	0.691	1446	694
Orange	Peer	0.092	0.701	0.691	1445	693
Blue	Peer	0.093	0.697	0.687	1841	697
Orderer1	Orderer	0.125	0.841	0.716	206	334
Orderer2	Orderer	0.125	0.841	0.716	188	330
Orderer3	Orderer	0.125	0.841	0.716	236	338

Fig. 1. Performance - Configuration A - Run#2

peer nodes at approximately 0.6 GB, the peer named Blue displayed a significantly high average latency of 1841 ms. This indicates a potential bottleneck at this specific node. In contrast, the orderer nodes (Orderer1, Orderer2, and Orderer3) demonstrated highly uniform behavior, with transaction times tightly clustered between 188 ms and 236 ms. This uniformity confirms that the ordering service is stable and well-balanced, and that block dissemination is functioning as expected.

These patterns indicate that the performance degradation, including the high transaction latency and the drop in success rate, is not due to infrastructure saturation or the ordering service. The data points toward architectural bottlenecks at the application and smart contract layer, likely occurring during the endorsement, validation, or commit stages of the transaction process.

2) *Performance - Configuration A - Run#10*: The performance profile for this iteration of Configuration A, detailed in Figure 2, reveals a mixed performance profile. The average transaction latency saw a notable reduction to 314.12 ms, a significant improvement from previous tests. However, the transaction success rate remains a critical concern at 97.14%, indicating that persistent inefficiencies likely remain within the validation and commit stages. The overall network storage increased by a proportional 3.6 GB.

Resource utilization among peer nodes shows shifting patterns. CPU usage decreased, now ranging between 1597 and 1951 millicores, with peer Blue again registering the highest load. This suggests it continues to carry a disproportionate share of validation tasks. Memory consumption across most peers is relatively uniform, but peer Orange stands out with a memory footprint of 5830 MiB, more than five times that of its counterparts. This suggests a persistent inefficiency, possibly from aggressive caching or a misconfigured memory limit. The storage footprint for peers also decreased, with consistent increases between 0.3 GB and 0.5 GB, aligning with the lower number of processed transactions.

The orderer nodes demonstrated consistent behavior in their storage increase (0.5 GB each) and low CPU usage (175 to 232

Instances	Total Trackables	Total Transaction	Avg. Transaction Time (ms)	Success Rate (%)		
5	500	37687	314,12	97,14		
Component	Type	Initial Size (GB)	Final Size (GB)	Storage Increase (GB)	Avg. CPU (millicores)	Avg. Memory (Mi)
Network	N/A	49	52	3.6	N/A	N/A
VOID	Peer	5.4	5.9	0.500	1696	1082
Red 1	Peer	5.4	5.7	0.300	1624	1092
Red 2	Peer	5.4	5.8	0.400	1622	1147
Red 3	Peer	5.4	5.8	0.300	1597	1153
Orange	Peer	5.5	5.8	0.500	1608	5830
Blue	Peer	5.2	5.7	0.400	1951	1124
Orderer1	Orderer	5.2	5.7	0.500	189	5367
Orderer2	Orderer	5.2	5.7	0.500	175	5367
Orderer3	Orderer	5.2	5.7	0.500	232	5362

Fig. 2. Performance - Configuration A - Run#10

millicores). However, their memory usage sharply increased, with all nodes averaging around 5367 MiB. This suggests increased buffering, possibly due to larger blocks or inefficient batching strategies, and points to memory saturation as a potential bottleneck.

In summary, this run shows a partial recovery from previous performance issues, with reduced latency and a slightly improved success rate. Nevertheless, significant architectural anomalies persist, specifically the sustained memory bloat in peer Orange and the dramatic rise in orderer memory usage, which are symptomatic of deeper inefficiencies or configuration misalignments.

3) *Performance - Configuration B - Run#2*: This test instance for Configuration B, detailed in Figure 3, showcases a significantly different performance profile. The system handled a high volume of transactions with a remarkable average transaction time of 227.73 ms, a substantial improvement over Configuration A. The most critical result is the transaction success rate of 99.99%, which strongly reinforces the system's reliability and stability under this specific workload.

A deeper dive into the component-level metrics reveals both consistent behaviors and key divergences. Peer nodes, for instance, display a healthy and uniform storage increase, each growing by approximately 0.92 to 1.06 GB, indicating even block propagation and balanced ledger updates. However, CPU utilization across peers varied by over 300 millicores, which could reflect differences in transaction volume or chaincode execution complexity. Specifically, the Blue peer had the highest CPU usage at 3384 millicores, implying it may have shouldered a heavier chaincode execution load, while the Orange peer showed a more moderate profile with the lowest CPU usage at 2679 millicores.

The ordering service's behavior remains highly consistent and balanced, with each orderer node recording a storage increase of 0.723 GB and modest CPU usage ranging from 93 mC to 149 mC. However, Orderer2 shows a striking anomaly in memory usage, reporting 114 MiB compared to the stable 1147 MiB of the other two orderers. This mirrors

Instances	Total Trackables		Total Transaction	Avg. Transaction Time (ms)		Success Rate (%)
5	500		58373	227,73		99,99
Component	Type	Initial Size (GB)	Final Size (GB)	Storage Increase (GB)	Avg. CPU (millicores)	Avg. Memory (Mi)
Network	N/A	7,5	16	8,5	N/A	N/A
VOID	Peer	0,778	1,7	0,922	2805	1036
Red 1	Peer	0,838	1,9	1,062	2888	1052
Red 2	Peer	0,838	1,9	1,062	2894	1047
Red 3	Peer	0,839	1,9	1,061	2979	1043
Orange	Peer	0,841	1,8	0,959	2679	1101
Blue	Peer	0,833	1,9	1,067	3384	1073
Orderer1	Orderer	0,877	1,6	0,723	122	1147
Orderer2	Orderer	0,877	1,6	0,723	93	114
Orderer3	Orderer	0,877	1,6	0,723	149	1148

Fig. 3. Performance - Configuration B - Run#2

a recurring pattern from previous datasets and suggests a potential underutilization specific to Orderer2, justifying a closer configuration audit for this node.

4) *Performance - Configuration B - Run#7*: This run provides further evidence of both expected performance trends and anomalies that warrant attention for optimization. As detailed in Figure 4, the system demonstrates a robust performance profile.

Across all peer nodes, the final storage size converged near 7 GB, with a relatively uniform increase of approximately 1.2 GB per node. This consistency in ledger growth suggests balanced block dissemination and state database updates. However, variance emerges in CPU and memory consumption, indicating potential inefficiencies. While most peers maintained average memory usage in the 1700-1800 MiB range, the Orange peer stands out with a markedly higher average of 6187 MiB. This disparity could indicate elevated chaincode activity or a memory management issue. Orange's CPU usage, however, remains in line with the others, suggesting its workload is memory-intensive rather than compute-bound. Similarly, the Blue peer exhibits the highest CPU consumption at 3574 millicores, well above the cluster average, which may suggest it is frequently involved in endorsement or is processing more transactions than its peers. These performance skews highlight an opportunity to audit chaincode placement and load distribution.

Turning to the ordering service, we observe an overall consistent storage growth of 0.6 GB across all three orderers. However, a pronounced anomaly is present in Orderer2's resource usage. It maintains an average memory footprint of just 227 MiB, significantly lower than Orderer1 and Orderer3, both of which hover around 4950 MiB. This anomaly, which mirrors the pattern seen in Run#2, suggests that Orderer2 may not be actively participating in consensus or block delivery. This persistent issue indicates an inefficient use of available ordering capacity and a potential fault in high-availability planning.

Instances		Total Trackables	Total Transaction	Avg. Transaction Time (ms)		Success Rate (%)
5		500	58410	268,73		99,97
Component	Type	Initial Size (GB)	Final Size (GB)	Storage Increase (GB)	Avg. CPU (milicores)	Avg. Memory (Mi)
Network	N/A	49	58	9	N/A	N/A
VOID	Peer	5,8	7	1,2	2984	1761
Red 1	Peer	5,8	7	1,2	3032	1728
Red 2	Peer	5,8	7	1,2	3110	1747
Red 3	Peer	5,8	7	1,2	3092	1752
Orange	Peer	5,7	7	1,3	2872	6187
Blue	Peer	5,8	6,9	1,1	3574	1770
Orderer1	Orderer	4,7	5,3	0,6	116	4950
Orderer2	Orderer	4,7	5,3	0,6	88	227
Orderer3	Orderer	4,7	5,3	0,6	144	4952

Fig. 4. Performance - Configuration B - Run#7

D. Stress Testing Results

Our stress testing evaluated the network's capacity to handle peak loads and sustain optimal performance. This section presents a comparative analysis of the system's behavior under the distinct conditions of Configuration A and Configuration B.

1) *System behavior under load:* Configuration B demonstrated its capacity to process a significantly higher transaction volume than Configuration A. The total network storage footprint for Configuration B reached 84 GB, compared to 52 GB for Configuration A, representing a 61% increase in storage demand. Despite this higher absolute demand, resource utilization patterns were more uniform across components in Configuration B, suggesting a more scalable behavior.

2) *Transactional Storage Efficiency Analysis:* An analysis of storage efficiency reveals a key difference between the two configurations. As shown in Figure 5, Configuration A exhibits a stable and proportional relationship between transaction volume, average transaction size, and storage increase. Across all test runs, the average storage per transaction remained within a tight band (90-130 KB), indicating a consistent ledger behavior. Tests with smaller average transaction sizes (95 KB) demonstrated the most efficient storage usage, suggesting that the system operates near an optimal baseline under these conditions. Conversely, tests with higher transaction sizes (e.g., 139.4 KB in Run#3 and 127.9 KB in Run#9) incurred a noticeable increase in storage per transaction, confirming that payload size is the primary driver of ledger growth. Overall, this configuration scales linearly with workload and, from a storage perspective, shows no signs of fundamental architectural inefficiencies. Maintaining average transaction sizes below 110 KB appears to provide the best trade-off between throughput and storage efficiency.

In contrast, Configuration B, represented in Figure 6, consistently had a higher transaction volume per run, averaging around 60000 transactions with larger payloads. While this setup delivers higher throughput, it does so at the cost of

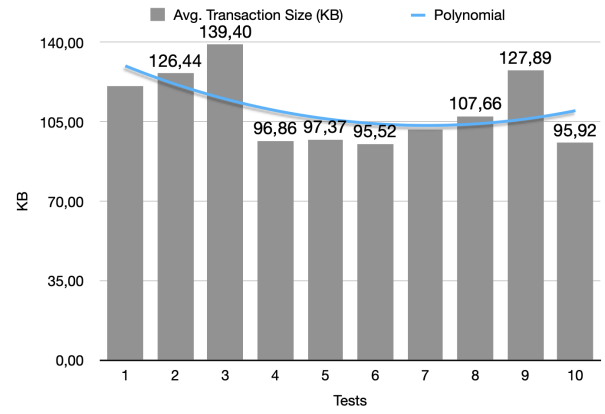


Fig. 5. Average Transaction Size - Configuration A

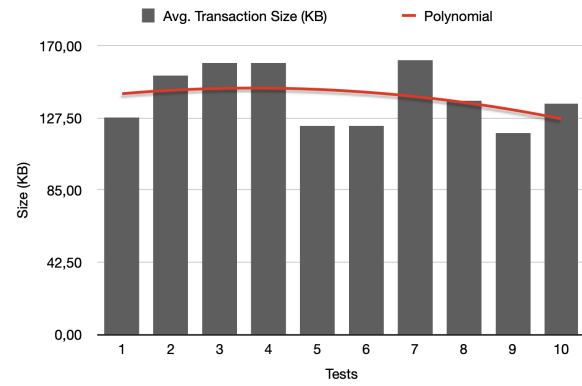


Fig. 6. Average Transaction Size - Configuration B

storage efficiency. The storage growth per transaction is substantial, averaging 133 KB. This is higher than the 90-130 KB band observed in Configuration A. Tests with larger payloads (e.g., ≥ 150 KB) result in nearly proportional storage growth, indicating that ledger writes are storing most of the transaction content without optimization. Even in lower-payload tests, significant storage overhead remains, pointing to a design with heavier metadata or protocol-layer costs.

Overall, Configuration B favors performance and volume over compactness, making it suitable for high-throughput environments but potentially problematic for storage-constrained deployments. This presents a key trade-off between operational costs and system responsiveness.

3) *Component Memory Usage Trends:* The data in Figure 7 illustrates the memory usage patterns among peer and orderer nodes for Configuration A, revealing both predictable scaling and high-variance behavior.

The core peers Red and VOID demonstrate a steady, linear memory growth, increasing from approximately 690 MiB in Run#1 to 1100–1150 MiB by Run#10. This suggests predictable scaling of internal ledger state and caching as new transactions are processed. In contrast, the Orange and Blue peers exhibit high-variance behavior. Orange shows an

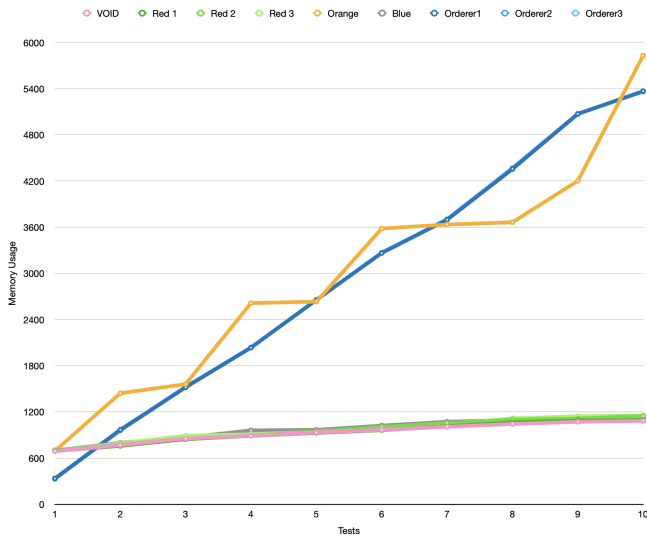


Fig. 7. Memory Usage Overview - Configuration A

increase from 693 MiB in Run#1 to 5830 MiB in Run#10 (an 8.4x increase). Blue also grows rapidly, though less extremely, ending at 1124 MiB. These trends suggest that the Orange peer, in particular, is executing resource-intensive operations, likely related to active chaincode execution with high memory allocation.

All three orderer nodes in this configuration show a consistent but heavy memory growth. They exhibit near-identical and linear memory increases, rising from 330 MiB to 5365 MiB (a 16x increase). This sharp, uniform trend indicates that orderers are likely holding large portions of block history in memory or are experiencing unbounded journaling or buffer growth. The symmetry among the orderers suggests this is a systemic architectural behavior rather than an implementation anomaly.

Memory values for Configuration B in Figure 8 are reported in mebibytes (Mi) and reflect per-component snapshots taken post-execution of each test. All peer nodes demonstrate consistent and predictable memory growth across tests, indicating a typical accumulation of state data and transient caches from transaction processing. The average memory utilization increase from Run#1 to Run#10 is approximately 2.5x to 3x for most peer components. The Red series nodes (Red1, Red2, and Red3) all follow nearly identical memory trajectories, with usage increasing from 877 Mi to 2449 Mi. This symmetry validates a balanced operational load and indicates no significant skew in smart contract interactions or endorsement patterns.

However, a number of anomalies are present. The peers Orange and Blue continue to show elevated memory usage compared to other peers, with Orange in particular exhibiting a steep increase from 865 Mi (Run#1) to 6637 Mi (Run#10), which is more than a 7.5x increase. This stands out significantly and warrants further investigation into potential load balancing issues or a logical role asymmetry. However,

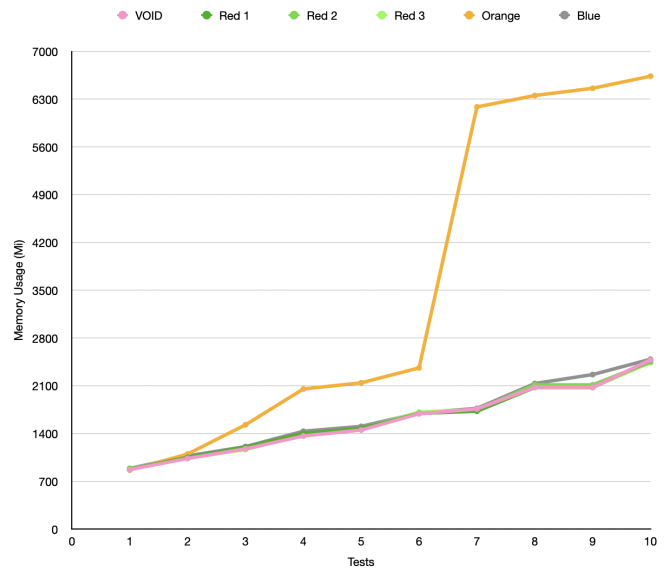


Fig. 8. Memory Usage Overview - Configuration B

a number of anomalies are present. The Orange and Blue peers continued to show elevated memory usage compared to other peers. Orange, in particular, exhibited a steep increase from 865 Mi in Run#1 to 6637 Mi in Run#10, a more than 7.5x increase. This significant disparity warrants further investigation into potential load balancing issues.

A severe imbalance was identified within the ordering service. While Orderer1 and Orderer3 exhibited major memory growth, reaching 6951 Mi and 7768 Mi respectively by Run#10, Orderer2 remained low, rising from just 91 Mi to 316 Mi. This behavior suggests it may be misclassified or idle, which would present a significant underutilization issue. This major memory growth in the active orderers, particularly with multiple nodes surpassing 6 GB, points to systemic memory scaling risks as transaction volume increases. Further steps should include auditing Orange's workload, investigating Orderer2's role, and considering a strategy to rebalance the ordering load to ensure the network can scale effectively.

E. Discussion

An analysis of the test results for both configurations reveals a critical trade-off in the network's performance profile. While Configuration B successfully met the primary performance and reliability targets, neither configuration can be considered fully successful due to fundamental inconsistencies and resource inefficiencies. The established success criteria included a target for reliability (a 99.9% success rate), performance (average transaction latency ≤ 2000 ms), and resource efficiency (proportional growth with minimal variance). The data indicates that the network, in its current state, cannot simultaneously satisfy all of these requirements.

The comparative analysis across ten runs reveals that Configuration B's superior performance is a direct result of superior throughput, lower transaction latency, and significantly

higher transaction success rates. Configuration B consistently maintained success rates above 99.97%, while Configuration A's performance ranged more variably between 93.94% and 97.14%, a major disqualifier for production readiness. Transaction times in Configuration B remained stable and efficient, averaging around 245 ms, in stark contrast to Configuration A's erratic performance, which peaked above 750 ms in some tests. These improvements are accompanied by a marked increase in the total number of processed transactions, with Configuration B handling up to 62000 transactions per test, compared to Configuration A's upper bound of roughly 55000 transactions.

However, this improved performance in Configuration B is coupled with a steeper network storage growth curve, expanding to 84 GB by the final test compared to 52 GB in Configuration A. Memory usage patterns also show significant anomalies. The Organization Orange peer node consistently had a disproportionately high memory footprint, surpassing 6.6 GB, which suggests a load-balancing issue, a unique processing burden, or inefficient state handling. Additionally, the Orderer2 node remained largely inactive, suggesting redundancy without effective utilization.

In conclusion, neither Configuration A nor Configuration B fully satisfies all the defined success criteria. Configuration A fails on the most critical metric, transaction reliability while Configuration B, despite achieving high performance, exhibits systemic and resource inconsistencies that would present major scalability challenges and operational risks in a production environment. Given that Configuration A's failure on reliability is a major blocker, the results suggest that Configuration B is the preferred candidate for production, but with necessary refinements. This approach would leverage its high performance and reliability, while focusing architectural refinement on resolving the identified systemic issues. Key areas for refinement include reviewing smart contract payloads for size and serialization efficiency, investigating the memory and processing behavior of the Organization Orange peer, and resolving the dormancy of Orderer2 to ensure proper load distribution and redundancy.

III. CONCLUSION

Based on the performance evaluation of the blockchain-based livestock traceability platform, the results have shown a significant trade-off between the configurations tested. While neither Configuration A nor B fully satisfied all success criteria out of the box, the data strongly supports Configuration B as the preferred base for a viable business solution.

Configuration B demonstrated superior throughput, lower transaction latency (averaging around 245 ms), and, most critically, a near-perfect transaction success rate consistently above 99.97%. These results directly align with the core performance and reliability criteria for the platform. In contrast, Configuration A failed to meet the essential reliability target, with success rates falling below 99.9%, rendering it unsuitable for production.

To transition Configuration B from a promising candidate to a production-ready system, specific optimizations are required.

The analysis identified key areas of concern, including a steeper network storage growth curve and significant memory usage anomalies in the Organization Orange peer node, which must be investigated to resolve potential load-balancing or state-handling inefficiencies. Furthermore, the dormancy of the Orderer2 node must be addressed to ensure proper load distribution and high availability.

Future work will start by investigating the difference between functional and non-functional requirements of livestock traceability and other cases of general goods' traceability. Furthermore, we will focus on implementing these necessary refinements to Configuration B and extending the analysis to a production-scale version of the platform. By addressing the identified architectural bottlenecks, we expect the platform to provide a robust and scalable technology base for reliable livestock traceability.

REFERENCES

- [1] Moe, T.: Perspectives on traceability in food manufacture. *Trends in Food science & technology* 9(5), 211–214 (1998).
- [2] European Union: Regulation (ec) no 178/2002 of the european parliament and of the council of 28 january 2002 laying down the general principles and requirements of food law, establishing the european food safety authority and laying down procedures in matters of food safety. *Official Journal of the European Communities* 31, 1–24 (2002), also known as the General Food Law Regulation.
- [3] Hyperledger Fabric docs, <https://hyperledger-fabric.readthedocs.io/en/release-2.2/whatis.html>, accessed: 2025-08-11.
- [4] Rajendran, V., Santhosh, V., and Pradeep, L.: Evaluation of Blockchain Service Level Agreement (SLA) Using Hyperledger Fabric (HLF). *International Research Journal on Advanced Science Hub* 5(05) (2023)
- [5] Kumar, R. S., and Ramaswamy, M. A.: Latency performance modeling and analysis for hyperledger fabric blockchain network. *Information Processing & Management* (2020)
- [6] The Linux Foundation: Hyperledger Blockchain Performance Metrics White Paper. LF Decentralized Trust (2019). [Online; accessed 12-August-2025]
- [7] M. Boughdiri, T. Abdellatif and C. Ghedira Guegan, "A Systematic Literature Review on Blockchain Storage Scalability," in *IEEE Access*, vol. 13, pp. 102194-102219, 2025, doi: 10.1109/ACCESS.2025.3578451.