Vol. 8, No. 2, April 2026

Journal of Information Systems and Informatics

Published By Asosiasi Doktor Sistem Informasi Indonesia

Comparative Quality of Services and Resource Utilization Analysis of Free5GC and Open5GS in Resource-Constrained Private 5G Networks

Naufal Hanan Lutfianto[1], Budi Prasetya[2], Vivi Monita[3]

[1,2]Telecommunication Engineering Study Program, School of Electrical Engineering, Telkom University, Bandung, Indonesia

[2]Smart City Information System Study Program, School of Applied Science, Telkom University, Bandung, Indonesia

Received:
September 10, 2025
Revised:
March 10, 2026
Accepted:
April 11, 2026
Published:
April 22, 2026

Corresponding Author:
Author Name*:
Naufal Hanan Lutfianto
Email*:
naufalhananl@telkomuniversity.ac.id

DOI:
10.63158/journalisi.v8i2.1513

Abstract. This research compares the performance of two widely used open-source 5G core (5GC) platforms, Free5GC and Open5GS, in a resource-constrained private network environment. While previous studies have mainly focused on feature comparison or large-scale deployments, performance under limited computational resources has received less attention, particularly for small-scale enterprise use cases. In this work, both platforms are integrated with UERANSIM to emulate end-to-end 5G communication and evaluated under dynamic user equipment (UE) scaling. Each 5GC instance and simulator component is allocated one CPU core and 2 GB of memory. Performance is assessed using key Quality of Service (QoS) metrics, including throughput, latency, packet loss, and resource utilization (CPU and memory), under both TCP and UDP traffic. The results show that Open5GS consistently provides better performance than Free5GC. It achieves up to 10.58 Mbps throughput compared to 9.22 Mbps and maintains lower latency around 0.72–0.73 ms, while Free5GC reaches up to 1.20 ms as the number of UEs increases. In addition, Free5GC reaches high CPU utilization earlier under increasing load. These differences are mainly related to its microservice-based architecture, which introduces additional processing overhead.

Keywords: Free5GC; Open5GS; UERANSIM; private 5G network; open-source 5GC; QoS evaluation

1. INTRODUCTION

The rapid development of 5G networks has introduced new requirements for high data rates, low latency, and flexible network deployment. Unlike previous generations, the 5G Core (5GC) adopts a service-based architecture (SBA) [1], enabling scalable and cloud-native network functions that support diverse applications such as enhanced mobile broadband (eMBB), ultra-reliable low-latency communication (URLLC), and massive machine-type communication (mMTC) [2], [3], [4].

In addition to commercial deployments, open-source 5GC platforms have gained increasing attention in research and private network environments due to their flexibility, cost efficiency, and support for rapid prototyping. Among these platforms, Free5GC and Open5GS are widely used for implementing and evaluating 5G core functionalities [5], [6]. Despite their similar objectives, the two platforms differ significantly in architectural design. Free5GC adopts a microservice-based approach, while Open5GS follows a more integrated architecture. These differences may have a direct impact on performance, scalability, and resource efficiency [7].

Several previous studies have evaluated open-source 5GC platforms, primarily focusing on feature comparison, control-plane performance, or large-scale deployments. However, limited attention has been given to evaluating their performance under resource-constrained environments that reflect small-scale private or enterprise deployments [8]. In such conditions, computational limitations can significantly affect system stability, packet processing efficiency, and overall Quality of Service (QoS) [9], [10].

To address this gap, this research presents a structured performance evaluation of Free5GC and Open5GS in a controlled, resource-constrained testbed [11]. The evaluation is conducted using dynamic user equipment (UE) scaling and focuses on key QoS metrics, including throughput, latency, packet loss, and resource utilization. In addition to quantitative measurement, this study provides an architecture-aware analysis that links performance behaviour to system design characteristics.

The main contributions are as follows:

1) A controlled experimental comparison of Free5GC and Open5GS integrated with UERANSIM in a resource-constrained private 5G testbed.
2) A comprehensive evaluation of QoS performance and system efficiency using throughput, latency, packet loss, and CPU and memory utilization under dynamic UE scaling.
3) Practical insights into deployment suitability, highlighting the trade-offs between flexibility and efficiency in open-source 5GC implementations.

2. METHODS

This research evaluates the performance of Free5GC and Open5GS in a controlled and resource-constrained testbed designed to represent small-scale private 5G network deployments. Both platforms are deployed on the same physical host to ensure consistent hardware conditions and eliminate variability caused by different system environments [12]. The experiments are conducted on a Lenovo X1 Carbon laptop equipped with an Intel Core i7 (9th generation) processor and 16 GB of RAM. The system runs a virtual machine environment using VMware with Ubuntu 22.04 as the operating system. All 5G core components and UERANSIM instances are executed within the virtual machine to provide isolation and consistent resource allocation during testing.

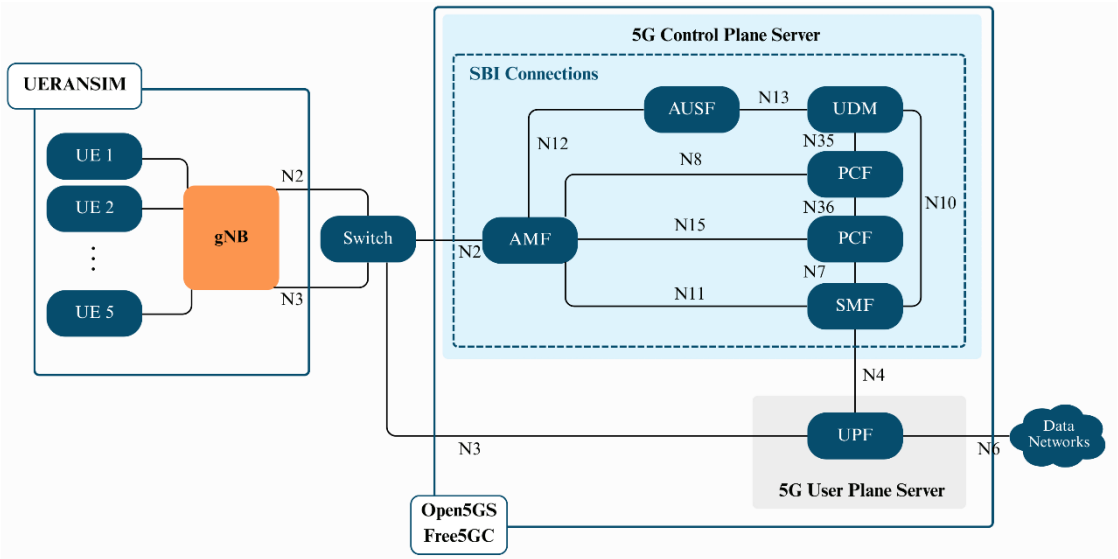


Figure 1. Experimental testbed architecture of Free5GC and Open5GS integrated with UERANSIM

Each 5G core instance, including both control plane and user plane functions, is configured with one dedicated CPU core and 2 GB of memory. The UERANSIM components, including the simulated UE and gNodeB (gNB), are also assigned the same resource limits [13]. This setup is designed to emulate constrained computational environments commonly found in small-scale enterprise or laboratory deployments [14], [15]. Free5GC and Open5GS are integrated with UERANSIM to enable end-to-end 5G communication, including UE registration, session establishment, and user plane data transmission [16], [17]. The overall system architecture is illustrated in Figure 1. All components are interconnected through a consistent virtual network configuration within the virtual machine. To ensure a fair comparison, both platforms are evaluated under identical configurations, including network topology, resource allocation, and traffic conditions. Background processes are minimized, and each experiment is executed under the same initial system conditions. Traffic generation is performed using iperf3 under both Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) to evaluate network performance under reliable and best-effort transmission conditions. Continuous traffic is generated between the simulated UE and the core network during each test scenario.

2.1 Dynamic UE Scaling Scenario

To evaluate system scalability, a dynamic UE scaling scenario is implemented by gradually increasing the number of UEs connected to the network. The experiments are conducted at three UE levels, namely 10, 30, and 60 UEs, representing low, moderate, and relatively high load conditions within the constrained testbed environment [18], [19]. Each scenario is executed using the same traffic configuration and resource allocation for both Free5GC and Open5GS. The scaling process is performed in defined steps, allowing observation of performance trends as the number of active UEs increases. All scenarios are conducted under identical conditions for both platforms to ensure that performance differences are primarily influenced by architectural characteristics rather than external factors.

2.2 Performance Metrics

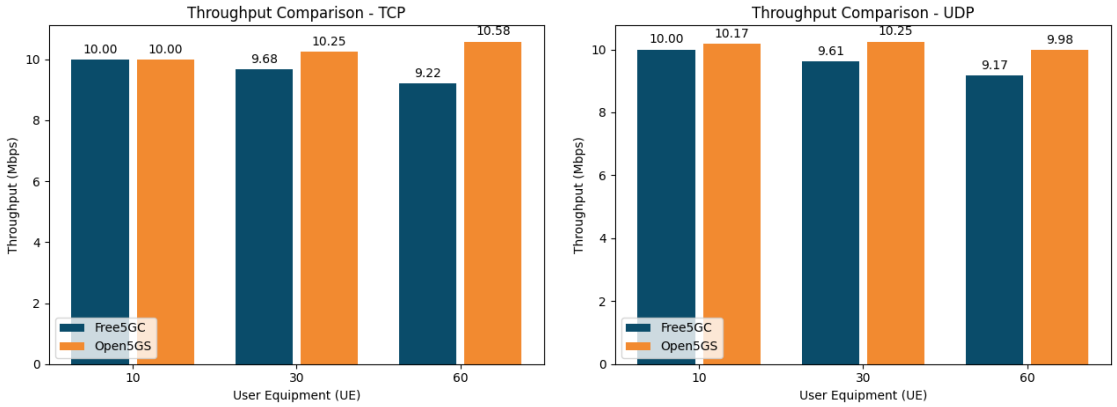
The performance evaluation focuses on key QoS metrics, including throughput, latency, packet loss, and resource utilization (CPU and memory) [20], [21]. Throughput represents the effective data transmission rate achieved during each test [22]. Latency reflects the

delay in packet delivery between endpoints, while packet loss indicates the reliability of data transmission under increasing load conditions [23][24]. CPU and memory utilization are analyzed to assess resource efficiency and system scalability in constrained environments [25]. Each test is conducted for a duration of 10 minutes to observe system behavior under sustained load conditions. To ensure consistency and reliability of the results, each scenario is repeated 10 times, and the reported values represent the average of these measurements for each performance metric.

3. RESULTS AND DISCUSSION

3.1 Throughput

The throughput results under dynamic UE scaling reveal a clear performance difference between Free5GC and Open5GS for both TCP and UDP traffic, as illustrated in Figure 2.



(a) (b)

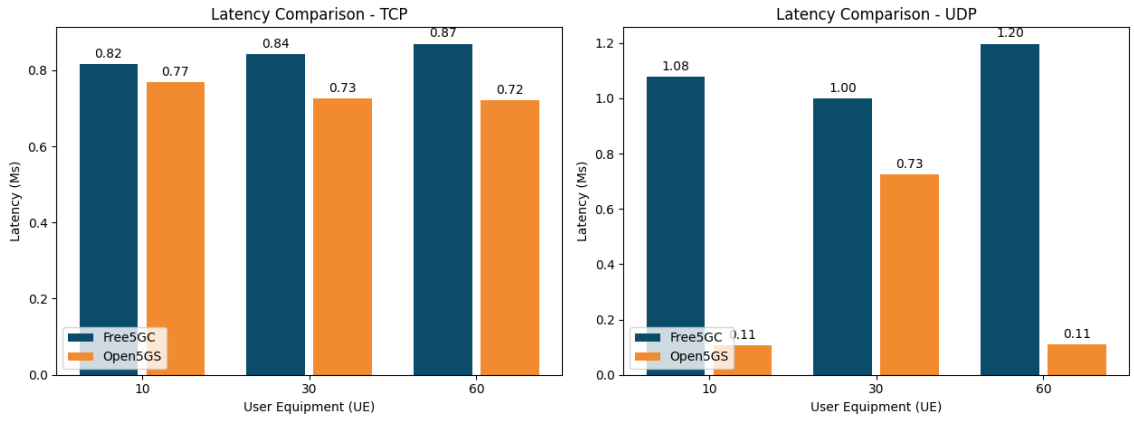
Figure 2. TCP (a) dan UDP (b): Throughput comparison for dynamic UE under low-resource conditions

Under TCP traffic, both platforms achieve similar throughput at low load (10 UEs), reaching approximately 10 Mbps. However, as the number of UEs increases, Free5GC shows a gradual decrease in throughput, dropping to 9.22 Mbps at 60 UEs. In contrast, Open5GS maintains stable performance and even shows a slight increase, reaching up to 10.58 Mbps at 60 UEs. A similar trend is observed under UDP traffic. While both platforms start with comparable throughput at 10 UEs, Free5GC experiences a consistent decline as the load increases, decreasing to 9.17 Mbps at 60 UEs. Meanwhile, Open5GS maintains higher throughput, reaching 10.25 Mbps at 30 UEs and 9.98 Mbps at 60 UEs, indicating more

stable packet handling under increasing load conditions. This behavior can be explained by the architectural differences between the two platforms. Free5GC adopts a microservice-based architecture, which introduces additional overhead due to inter-process communication between network functions. As the number of UEs increases, this overhead leads to higher CPU utilization and reduced packet processing efficiency, resulting in throughput degradation. In contrast, Open5GS employs a more integrated architecture with lower communication overhead between components. This enables more efficient packet processing and better utilization of limited computational resources, allowing it to maintain stable throughput even under higher UE load.

3.2 Latency

The latency results demonstrate a consistent and significant performance difference between Free5GC and Open5GS under both TCP and UDP traffic conditions, as shown in Figure 3.



(a) (b)

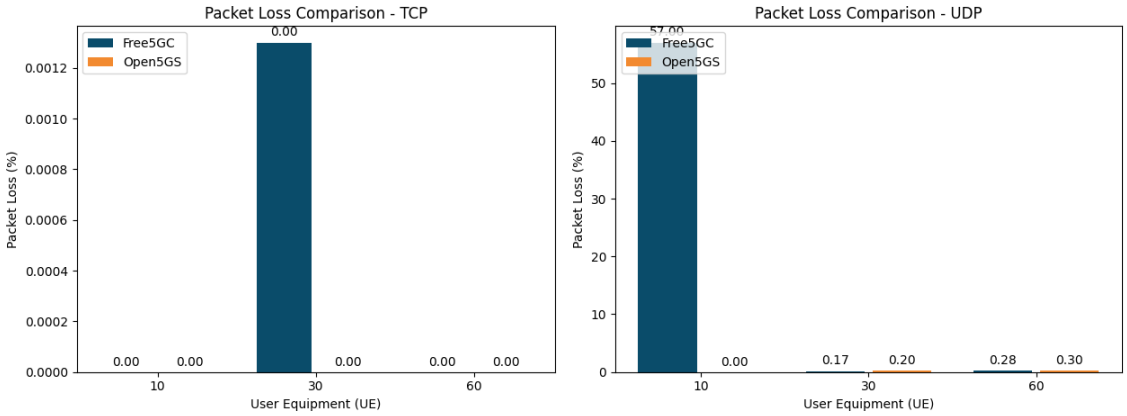
Figure 3. TCP (a) dan UDP (b): Latency comparison for dynamic UE under low-resource conditions

Under TCP traffic, Open5GS consistently achieves lower latency compared to Free5GC across all UE levels. At 10 UEs, Free5GC records 0.82 ms, while Open5GS achieves a lower latency of 0.77 ms. As the number of UEs increases, Free5GC shows a gradual increase in latency, reaching 0.87 ms at 60 UEs. In contrast, Open5GS maintains more stable performance, with latency decreasing slightly to 0.72 ms at higher load levels. The difference becomes more pronounced under UDP traffic. Free5GC exhibits higher and

more variable latency, increasing from 1.08 ms at 10 UEs to 1.20 ms at 60 UEs. Meanwhile, Open5GS maintains significantly lower latency, with values as low as 0.11 ms at 10 UEs and 60 UEs, and 0.73 ms at 30 UEs. This indicates that Open5GS handles best-effort traffic more efficiently, even under increasing load conditions. These results can be attributed to the architectural differences between the two platforms. Free5GC adopts a microservice-based architecture, where communication between distributed network functions introduces additional processing delay, especially under higher load conditions. This results in increased latency and variability as system resources become constrained. In contrast, Open5GS utilizes a more integrated architecture, which reduces inter-process communication overhead and enables faster packet processing. This leads to lower and more stable latency across different traffic conditions and UE levels.

3.3 Packet Loss

The packet loss results highlight the reliability performance of Free5GC and Open5GS under both TCP and UDP traffic conditions, as shown in Figure 4.



(a) (b)

Figure 4. TCP (a) dan UDP (b): Packet loss comparison for dynamic UE under low-resource conditions

Under TCP traffic, both platforms demonstrate near-zero packet loss across all UE levels. This is expected, as TCP incorporates retransmission mechanisms that ensure reliable data delivery even under increasing network load. A negligible packet loss is observed in Free5GC at 30 UEs; however, its impact is minimal and does not significantly affect overall performance. In contrast, the UDP results reveal a more distinct difference

between the two platforms. At 10 UEs, Free5GC exhibits a significantly high packet loss of approximately 57%, while Open5GS maintains zero packet loss under the same condition. This indicates that Free5GC experiences instability under certain initial conditions when handling best-effort traffic. As the number of UEs increases, packet loss in Free5GC decreases substantially, reaching 0.17% at 30 UEs and 0.28% at 60 UEs. Meanwhile, Open5GS shows slightly increasing packet loss, reaching 0.20% at 30 UEs and 0.30% at 60 UEs. Despite this increase, Open5GS maintains a more consistent and predictable behavior across different load levels. The high packet loss observed in Free5GC at low load may be attributed to initialization overhead or transient instability in the microservice-based architecture, where multiple network functions must coordinate during early stages of communication. As the system stabilizes, packet handling becomes more efficient, resulting in improved performance at higher UE levels.

3.4 Resource Utilization

Efficient resource distribution is critical in large-scale 5GC deployments, as it directly affects the system's capacity to manage a growing number of UE connections. This section focuses on evaluating memory utilization as a key indicator of resource efficiency.

3.4.1 Memory

Resource utilization tests were conducted in a low-resource environment, using both TCP and UDP protocols on Free5GC and Open5GS. Memory usage patterns were observed as the number of UEs increased, and the results are presented in Figure 5 to 8 for both protocols across the two platforms. The findings indicate a linear increase in memory consumption for both Free5GC and Open5GS as the number of UEs increases. However, Free5GC consistently consumed more RAM than Open5GS under identical conditions.

This behavior can be attributed to Free5GC microservice-based architecture, which involves running multiple independent network functions, each requiring its memory context and inter-process communication overhead. In contrast, Open5GS exhibits lower memory usage, benefiting from its monolithic and tightly integrated architecture, which reduces the need for extensive inter-process messaging and context switching. As a result, Open5GS achieves more efficient memory handling, particularly in scenarios with limited computational resources. Notably, the results also show that memory usage does

not reach full saturation, even as the number of UEs increases. This suggests that RAM is not the primary bottleneck in these low-resource environments. Instead, performance limitations are more closely linked to CPU constraints, which result in processing delays and ultimately lead to system instability under high-load conditions. These observations emphasize the importance of optimizing both memory and CPU usage in 5GC implementations, particularly for scalable deployments where resource efficiency is a key operational requirement.

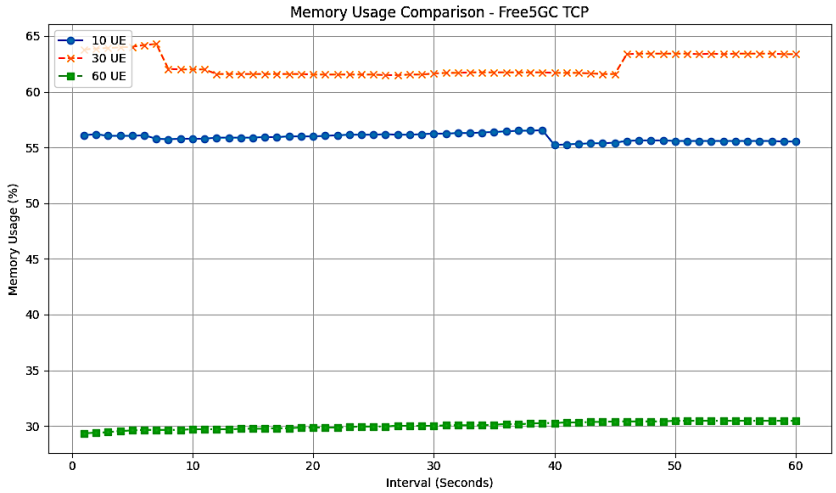


Figure 5. TCP: Memory usage comparison for dynamic UE under low-resource of Free5GC

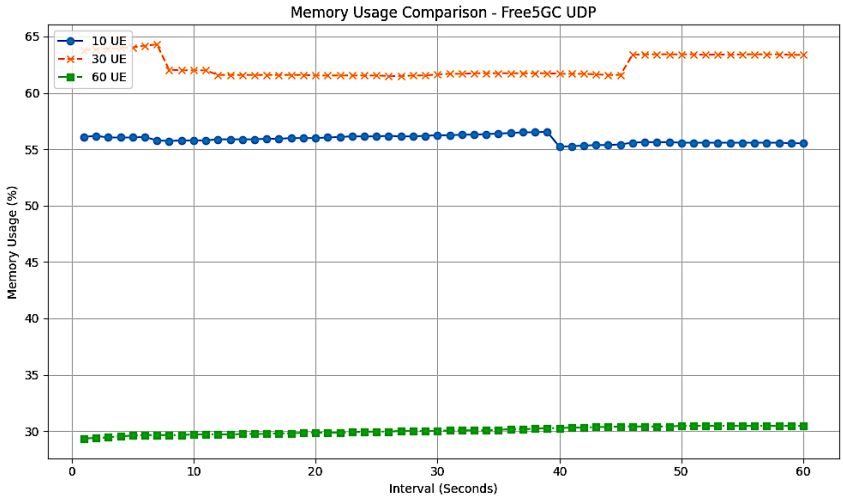


Figure 6. TCP: Memory usage comparison for dynamic UE under low-resource of Free5GC

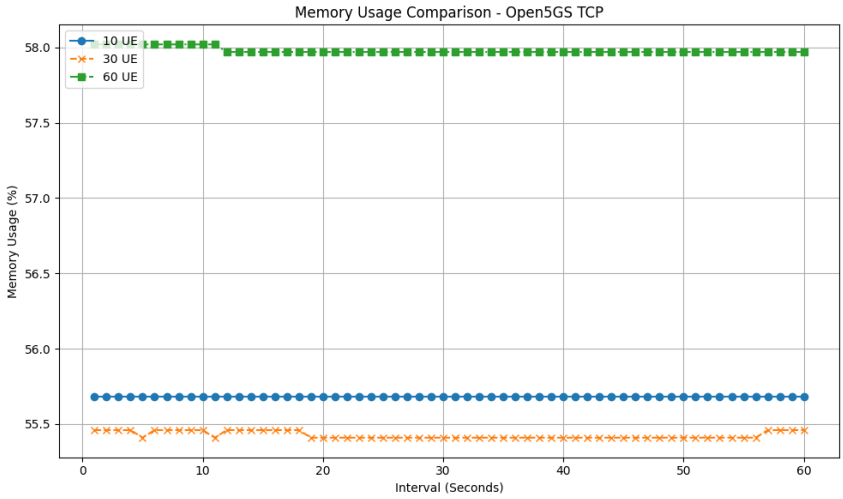


Figure 7. TCP: Memory usage comparison for dynamic UE under low-resource of Open5GS

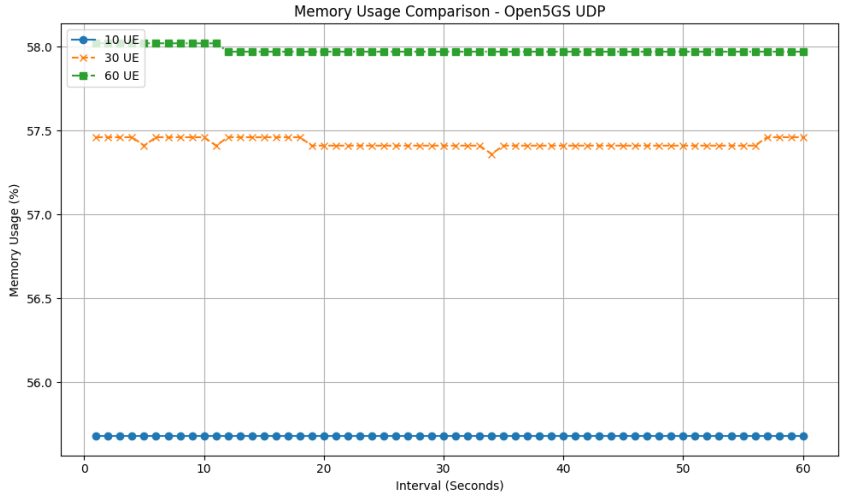


Figure 8. TCP: Memory usage comparison for dynamic UE under low-resource of Open5GS

3.4.2 CPU

CPU utilization testing was conducted under resource-constrained conditions using TCP and UDP traffic for both Free5GC and Open5GS. Figures 9 and 10 illustrate the CPU usage of Open5GS under TCP and UDP workloads, respectively, while Figures 11 and 12 present the corresponding results for Free5GC. As the UE load increases, both platforms exhibit a progressive rise in CPU utilization, reflecting the growing computational overhead required to handle control-plane signaling and user-plane packet processing during dynamic UE scaling. At the initial stage, CPU usage remains low, reflecting minimal

processing demand. However, as additional UEs are introduced, CPU utilization rises sharply, approaching 90% usage on both systems under pressure. Notably, Free5GC reaches this high utilization threshold earlier than Open5GS.

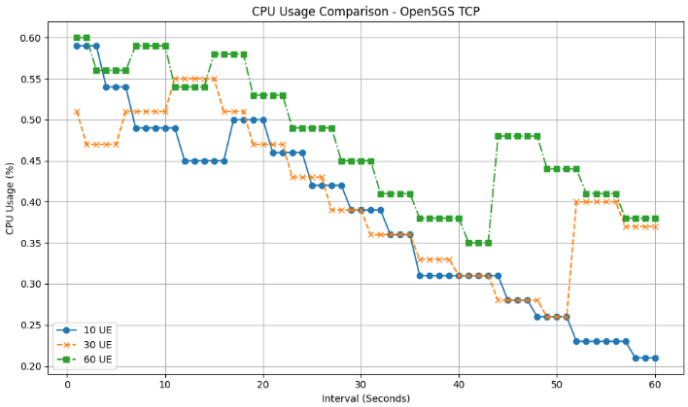


Figure 9. TCP: CPU usage comparison for dynamic UE under low-resource of Open5GS

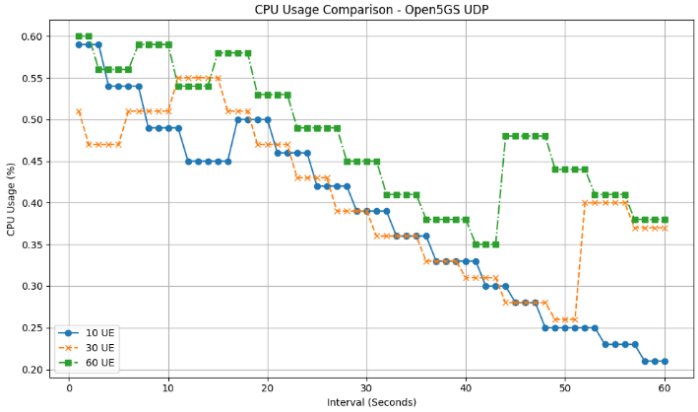


Figure 10. UDP: CPU usage comparison for dynamic UE under low-resource of Open5GS

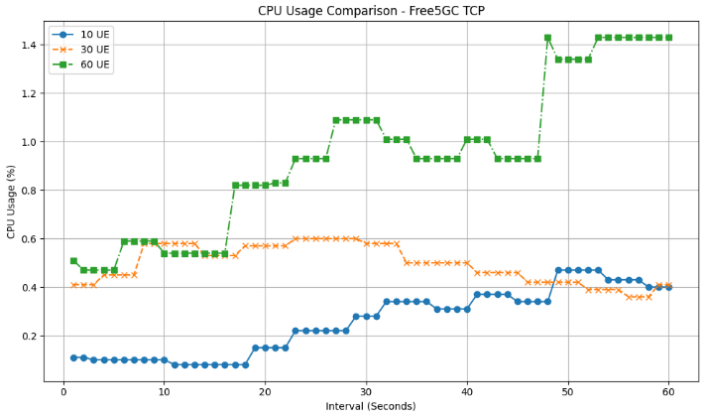


Figure 11. TCP: CPU usage comparison for dynamic UE under low-resource of Free5GC

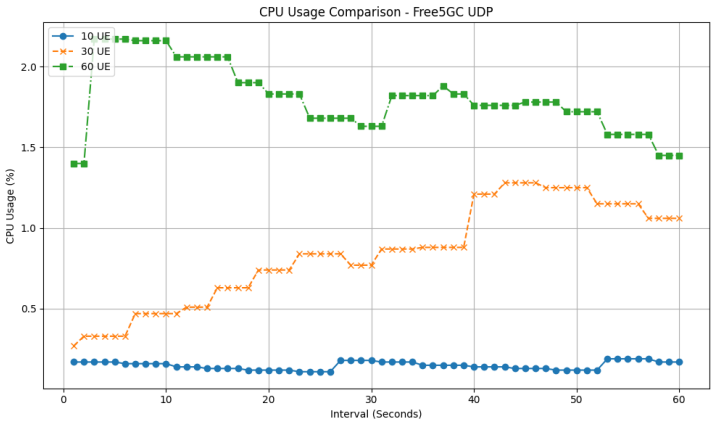


Figure 12. UDP: CPU usage comparison for dynamic UE under low-resource of Free5GC

This is primarily due to the inherent overhead in its SBA, which relies on microservices that communicate through asynchronous messaging. The resulting frequent inter-service interactions significantly increase processing demand, accelerating CPU exhaustion.

Conversely, Open5GS exhibits a more gradual increase in CPU usage, benefiting from its monolithic architecture. This design minimizes inter-process communication, thereby enabling more efficient handling of growing network load. The steadier growth in CPU usage reflects better scalability and load management under resource-constrained conditions. These findings highlight the trade-offs between flexibility and efficiency in architectural design. While Free5GC modular design offers greater flexibility for testing and service customization, it incurs higher computational costs. Open5GS, with its simpler architecture, demonstrates greater processing efficiency, making it more suitable for private 5G networks and environments with limited computing resources.

3.5 Discussion

The results show that the performance of open-source 5G core platforms in a constrained environment is strongly influenced by their architectural design. Although both Free5GC and Open5GS successfully supported end-to-end communication with UERANSIM under all evaluated scenarios, their behavior diverged as the number of UEs increased. This confirms that, beyond functional compliance, implementation strategy plays an important role in determining the operational efficiency of a 5GC deployment, particularly in small-scale private or research environments where computational resources are limited [5]–[10], [14], [15]. Since both platforms were tested under identical

hardware, topology, and traffic conditions, the observed differences can be reasonably attributed to internal architectural characteristics rather than to external experimental variation [12]–[19].

A key finding of this study is that Open5GS maintained more stable throughput than Free5GC under both TCP and UDP traffic as UE load increased. At lower load levels, the two platforms delivered comparable throughput, indicating that both implementations are capable of supporting baseline user-plane communication effectively. However, as the number of active UEs increased, Free5GC exhibited a consistent throughput decline, whereas Open5GS preserved a relatively stable data rate. This behavior is consistent with the known architectural distinction between the two platforms: Free5GC adopts a microservice-based approach, while Open5GS follows a more integrated design [5]–[7]. In a resource-constrained setting, the service decomposition used by Free5GC likely introduces additional inter-process communication and scheduling overhead, reducing packet-processing efficiency when the system is subjected to higher signaling and traffic demand. Open5GS, by relying on tighter functional integration, appears to reduce this overhead and therefore achieves better throughput sustainability under load [7], [20]–[22].

The latency results reinforce this interpretation. Across all evaluated UE levels, Open5GS consistently produced lower latency than Free5GC for both TCP and UDP traffic. This difference became especially visible under UDP, where Free5GC showed both higher average delay and greater variability. In the context of 5G systems, low and stable latency is essential for service quality, especially for use cases associated with real-time communication and responsive network behavior [2]–[4]. The higher latency observed in Free5GC can be linked to the additional processing path created by distributed service interaction within the service-based architecture introduced in 5GC [1]. Under limited CPU allocation, the cost of communication among network functions becomes more significant, contributing to queueing and processing delay. By contrast, Open5GS appears to process signaling and user-plane traffic more directly, resulting in lower end-to-end delay. These findings suggest that, in low-resource deployments, architectural integration can provide a measurable advantage in maintaining latency performance [7], [20], [23], [24].

The packet loss analysis provides further insight into the reliability characteristics of both platforms. Under TCP traffic, packet loss remained negligible for both Free5GC and Open5GS, which is expected due to TCP retransmission and flow-control mechanisms [22]–[24]. Under UDP, however, the behavior was more distinctive. The unusually high packet loss observed in Free5GC at 10 UEs suggests that performance instability may emerge even before the system reaches high load. Since UDP lacks retransmission, such instability becomes immediately visible in the measured loss rate. One possible explanation is that the initialization and coordination overhead among Free5GC network functions affects packet handling during early-stage traffic establishment. After this initial condition, packet loss decreased considerably, indicating that the system became more stable once internal states and service interactions were fully established. Open5GS, in contrast, showed a more consistent pattern across all load levels, with only a slight increase in packet loss as the number of UEs rose. This suggests that Open5GS offers more predictable behavior under best-effort traffic, which is desirable in practical private-network operation where traffic conditions may vary over time [7]–[10], [23], [24].

The resource utilization results help explain why these QoS differences emerged. Memory consumption increased approximately linearly with the number of UEs on both platforms, which is expected because each additional UE introduces more context information, session state, and traffic-related processing demand [20], [21], [25]. However, Free5GC consistently consumed more memory than Open5GS. This is consistent with the characteristics of a microservice-based design, where multiple network functions operate as separate processes with their own memory allocation and communication buffers [1], [7]. Open5GS, by adopting a more consolidated architecture, reduces duplication of runtime context and therefore uses memory more efficiently. At the same time, the results indicate that memory usage did not reach saturation during the experiments. This suggests that RAM was not the dominant bottleneck in the evaluated environment, even though memory efficiency remains relevant for scalability and deployment density in constrained systems [14], [15], [25].

In contrast, CPU utilization emerged as the primary limiting factor for both platforms. As UE count increased, CPU usage rose sharply, but Free5GC reached critical utilization levels earlier than Open5GS. This strongly indicates that processing overhead, rather than memory exhaustion, was the main source of throughput degradation and latency growth

in the testbed. This finding is particularly important because it clarifies the operational trade-off between flexibility and efficiency that has been noted in prior comparisons of open-source 5GC implementations [5]–[8]. Free5GC, through its service-based and modular architecture, offers structural flexibility that is useful for experimentation, function isolation, and cloud-native development. However, this flexibility comes at a computational cost, especially when the available processing resources are tightly bounded. Open5GS, with its more integrated implementation, appears better suited to low-resource execution because it minimizes inter-function coordination overhead and preserves more CPU capacity for actual control-plane and user-plane processing [7], [14], [15], [25].

From a deployment perspective, these findings have practical implications for private 5G networks, small enterprise environments, and laboratory-scale testbeds. In such scenarios, infrastructure is often limited, and performance stability may take priority over architectural extensibility [8]–[10], [14], [15]. Based on the observed results, Open5GS appears to be more appropriate when the objective is to achieve efficient and stable operation on modest hardware. Free5GC remains highly relevant, but its strengths may be better aligned with use cases that prioritize modularity, service separation, and architectural experimentation over computational efficiency. This distinction is important because open-source platforms are frequently selected not only for cost reasons, but also for their suitability to specific operational and research objectives [5], [6], [8].

The discussion should also be interpreted in light of the study limitations. The experiments were conducted in a single-host virtualized environment using a controlled low-resource configuration, which was intentionally designed to emulate constrained deployment conditions [12], [14], [15]. While this supports fairness and repeatability, the results may differ in larger distributed deployments, multi-host topologies, or cloud-native orchestration environments where network functions can be allocated additional resources or separated across nodes. The present evaluation also focused on aggregate QoS and resource metrics, without isolating the contribution of individual network functions or measuring signaling-stage delay in detail [20]–[25]. Future work could therefore extend the analysis by profiling specific network functions, examining container-based deployment scenarios, and evaluating larger UE populations to determine whether the observed architectural trends remain consistent at greater scale.

Overall, the findings indicate that Open5GS provides better efficiency and more stable QoS under resource-constrained UE scaling, while Free5GC offers greater architectural modularity at the cost of higher processing overhead [5]–[8]. This outcome is consistent with the broader design trade-off implied by the 5GC service-based architecture itself, where function separation improves flexibility and cloud-native alignment but may reduce efficiency when resources are limited [1], [7]. Therefore, the choice between Free5GC and Open5GS should depend on deployment priorities: Open5GS is more suitable for constrained operational environments that require stable performance, whereas Free5GC is better suited for development and research scenarios that benefit from a modular and service-oriented core network structure.

4 CONCLUSION

This research evaluates the performance of two widely used open-source 5GC platforms, Free5GC and Open5GS, in a resource-constrained private network environment using dynamic UE scaling. The evaluation focuses on key QoS metrics, including throughput, latency, packet loss, and resource utilization. The results show that Open5GS provides more stable and efficient performance compared to Free5GC under constrained conditions. Open5GS maintains higher and more consistent throughput, while Free5GC experiences performance degradation as the number of UEs increases. In terms of latency, Open5GS achieves lower and more stable delay, particularly under UDP traffic, whereas Free5GC shows increased latency under higher load. Both platforms demonstrate near-zero packet loss under TCP traffic; however, under UDP conditions, Open5GS exhibits more consistent performance, while Free5GC shows higher variability. From a resource perspective, Open5GS achieves better efficiency, while Free5GC incurs higher overhead due to its microservice-based architecture.

REFERENCES

[1] K. Veluchamy, R. R. Pavitra A., M. Isaivani, and J. M. Guerrero, 'Research Review on Performance Evaluation of Fifth Generation (5G) Technologies and Protocols':, in *Advances in Information Security, Privacy, and Ethics*, G. Prabhakar, N. Ayyanar, and S. Rajaram, Eds, IGI Global, 2024, pp. 55–76. doi: 10.4018/979-8-3693-2786-9.ch003.

[2] H. Holma, S. Kalyanasundaram, and V. Venkatesan, '5G Performance', in *5G Technology*, 1st edn, H. Holma, A. Toskala, and T. Nakamura, Eds, Wiley, 2024, pp. 239–303. doi: 10.1002/9781119816058.ch10.

[3] P. Mahadevan, H. M. Alabdeli, S. I B, I. Berejnov, D. Madrakhimova, and U. R, 'Dual Connectivity Management in 5G Mobile Internet Infrastructures', *J. Internet Serv. Inf. Secur.*, vol. 15, no. 2, pp. 256–270, May 2025, doi: 10.58346/JISIS.2025.I2.018.

[4] J. Navarro-Ortiz, P. Romero-Diaz, S. Sendra, P. Ameigeiras, J. J. Ramos-Munoz, and J. M. Lopez-Soler, 'A Survey on 5G Usage Scenarios and Traffic Models', *IEEE Commun. Surv. Tutor.*, vol. 22, no. 2, pp. 905–929, 2020, doi: 10.1109/COMST.2020.2971781.

[5] J.-C. Chen and K. K. Ramakrishnan, 'free5GC '25: The 1st free5GC World Forum', in *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security*, Taipei Taiwan: ACM, Nov. 2025, pp. 4928–4929. doi: 10.1145/3719027.3769096.

[6] R. Zhang, Y. Lin, S. Chen, and Z. Mo, 'A Multi-Node 5G Core Network Testbed Developed from Open5GS', in *2023 9th International Conference on Computer and Communications (ICCC)*, Chengdu, China: IEEE, Dec. 2023, pp. 1038–1043. doi: 10.1109/ICCC59590.2023.10507325.

[7] G. Zu *et al.*, 'Demo: Real-World Integration and Evaluation of Open-Source 5G Core with Commercial RAN', in *MILCOM 2025 - 2025 IEEE Military Communications Conference (MILCOM)*, Los Angeles, CA, USA: IEEE, Oct. 2025, pp. 881–882. doi: 10.1109/MILCOM64451.2025.11310344.

[8] T. Mukute, L. Mamushiane, A. A. Lysko, E.-R. Modroiu, T. Magedanz, and J. Mwangama, 'Control Plane Performance Benchmarking and Feature Analysis of Popular Open-Source 5G Core Networks: OpenAirInterface, Open5GS, and free5GC', *IEEE Access*, vol. 12, pp. 113336–113360, 2024, doi: 10.1109/ACCESS.2024.3441725.

[9] R. Reddy, M. Gundall, C. Lipps, and H. D. Schotten, 'Open Source 5G Core Network Implementations: A Qualitative and Quantitative Analysis', in *2023 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*, Istanbul, Turkiye: IEEE, Jul. 2023, pp. 253–258. doi: 10.1109/BlackSeaCom58138.2023.10299755.

[10] V. T. Van, N. P. Lam, D. M. Thang, D. X. Loc, and V. T. Duc, 'Design and Implementation of a 100Gbps FPGA-Based UPF for 5G Private Networks', in *2025 IEEE 26th International Conference on High Performance Switching and Routing (HPSR)*, Suita, Osaka, Japan: IEEE, May 2025, pp. 1–4. doi: 10.1109/HPSR64165.2025.11038876.

[11] M. Barbosa, M. Silva, E. Cavalcanti, and K. Dias, 'Open-Source 5G Core Platforms: A Low-Cost Solution and Performance Evaluation', in *2025 International Conference on Information Networking (ICOIN)*, Chiang Mai, Thailand: IEEE, Jan. 2025, pp. 99–104. doi: 10.1109/ICOIN63865.2025.10992769.

[12] T. Park, H. Lee, H. Kim, S. Han, T. Kim, and S. Pack, 'Divide and Cache: Design and Implementation of Control Plane Framework for Private 5G', *IEEE Trans. Netw. Serv. Manag.*, vol. 21, no. 2, pp. 1550–1560, Apr. 2024, doi: 10.1109/TNSM.2023.3334875.

[13] J. P. Ferreira, V. C. Ferreira, S. L. Nogueira, J. M. Faria, and J. A. Afonso, 'A Flexible Infrastructure-Sharing 5G Network Architecture Based on Network Slicing and Roaming', *Information*, vol. 15, no. 4, p. 213, Apr. 2024, doi: 10.3390/info15040213.

[14] P. Vanichchanunt, O. Ritruechai, N. Wuttiananchai, P. Thossaporn, L. Wuttisittikulkij, and S. Paripurana, 'Implementation of 5G Network Slicing Using Open Source Software', in *2024 12th International Electrical Engineering Congress (iEECON)*, Pattaya, Thailand: IEEE, Mar. 2024, pp. 1–6. doi: 10.1109/iEECON60677.2024.10537902.

[15] Y. Mirajkar, S. Ravichandra, A. K. Yadav, and P. Acharjee, 'Implementing 5G Core Network for Finding Metrics and Analyzing Network Performance Under DoS Attack', in *Data Science and Applications*, vol. 1797, S. J. Nanda, R. P. Yadav, M. Prasad, and M. Saraswat, Eds, in Lecture Notes in Networks and Systems, vol. 1797. , Cham: Springer Nature Switzerland, 2026, pp. 134–143. doi: 10.1007/978-3-032-15410-1_11.

[16] B. Zivkovic and Z. Cica, 'Multi-Connectivity Framework Based on Open-Source 5G Network Core', in *2024 32nd Telecommunications Forum (TELFOR)*, Belgrade, Serbia: IEEE, Nov. 2024, pp. 1–4. doi: 10.1109/TELFOR63250.2024.10819121.

[17] G. A. Santos, J. P. J. Da Costa, and A. A. S. Da Silva, 'Towards to Beyond 5G Virtual Environment for Cybersecurity Testing in V2X Systems', in *2023 Workshop on Communication Networks and Power Systems (WCNPS)*, Brasilia, Brazil: IEEE, Nov. 2023, pp. 1–7. doi: 10.1109/WCNPS60622.2023.10344440.

[18] R. Dhuny and F. Ying, 'Enhancing Education Accessibility: Portable Microservers for Computer-Based Testing in Resource-Constrained Environments', in *2025 22nd International Learning and Technology Conference (L&T)*, jeddah, Saudi Arabia: IEEE, Jan. 2025, pp. 36–41. doi: 10.1109/LT64002.2025.10940413.

[19] S. Brown, D. Harman, C. Anderson, and M. Dwyer, 'Characterizing Distributed Inferencing at the Edge in Resource-Constrained Environments', in *MILCOM 2023 - 2023 IEEE Military Communications Conference (MILCOM)*, Boston, MA, USA: IEEE, Oct. 2023, pp. 45–50. doi: 10.1109/MILCOM58377.2023.10356309.

[20] L. A. De Oliveira, A. L. De Oliveira, and E. F. Silva, 'Evaluation of EAP Usage for Authenticating Wi-Fi Enterprise Users in 5G Networks', in *2025 13th Wireless Days Conference (WD)*, Niterói, Rio de Janeiro, Brazil: IEEE, Dec. 2025, pp. 1–5. doi: 10.1109/WD67713.2025.11302618.

[21] J. Chen and Z. Tao, 'Quantitative Analysis and Mitigation of the Impact of Network Latency on Video Conferencing Communication Efficiency', *Hum. Factors J. Hum. Factors Ergon. Soc.*, vol. 68, no. 4, pp. 470–486, Apr. 2026, doi: 10.1177/00187208251398477.

[22] M. K. P and M. Supriya, 'Throughput Analysis with Effect of Dimensionality Reduction on 5G Dataset using Machine Learning and Deep Learning Models', in *2022 International Conference on Industry 4.0 Technology (I4Tech)*, Pune, India: IEEE, Sep. 2022, pp. 1–7. doi: 10.1109/I4Tech55392.2022.9952579.

[23] A. Morato, E. Ferrari, S. Vitturi, and F. Tramarin, 'A TSN-Based Technique for Latency Measurement in Real-Time Wireless Communication Networks', in *2024 IEEE International Symposium on Measurements & Networking (M&N)*, Rome, Italy: IEEE, Jul. 2024, pp. 1–6. doi: 10.1109/MN60932.2024.10615590.

[24] P. Wang, A. Zhang, G. Lin, and S. Wang, 'Research on the Relationship between Resource Utilization Rate of 5GC Network Elements and Network Traffic Volume', in *2024 9th International Conference on Intelligent Computing and Signal Processing (ICSP)*, Xian, China: IEEE, Apr. 2024, pp. 1321–1325. doi: 10.1109/ICSP62122.2024.10743930.

[25] K. L. Devi and S. Valli, 'Time series-based workload prediction using the statistical hybrid model for the cloud environment', *Computing*, vol. 105, no. 2, pp. 353–374, Feb. 2023, doi: 10.1007/s00607-022-01129-7.