Optimizing Intelligent Camera Surveillance in Smart Buildings: An SPN-based Edge-Fog Analysis

José Miqueias Araújo [1], Lucas Silva Lopes[1], Luiz Nelson Lima[1]
Vandirleya Barbosa[1], Arthur Sabino[1], Leonel Feitosa[1],
Flavia C. Delicato[2], Tuan Anh Nguyen [3,*] and Francisco Airton Silva[1]

[1]PASID Lab, Universidade Federal de Piauí (UFPI)

[2]Universidade Federal Fluminense (UFF) – RJ – Brasil

[3]Department of Software Engieering, Faculty of Information Technology,
Ho Chi Minh City University of Industry and Trade, HUIT, Ho Chi Minh city, Vietnam

{jmiqueias,lucaslopes092020,luizznelson}@ufpi.edu.br

{vandirleya.barbosa,arthursabino,leonel,faps}@ufpi.edu.br

fdelicato@gmail.com, anhngt@huit.edu.vn

Abstract. *Surveillance cameras play a pivotal role in modern security strategies, yet real-time video analysis demands substantial computational capabilities. This research employs Stochastic Petri Net (SPN) models to refine the planning and optimization of video surveillance infrastructures within intelligent buildings, leveraging Edge and Fog computing paradigms. By systematically evaluating key performance metrics such as mean response time, throughput, resource utilization, and drop probability, the proposed models inform more judicious resource allocation and scaling decisions. Our findings indicate that increasing Fog layer processing cores to 10 reduces the drop probability to around 35% at an arrival rate of 47.37 msg/ms, and maintains a mean response time below 10 ms at moderate arrival rates (up to approximately 29 msg/ms). These insights facilitate the design of more efficient, reliable, and scalable surveillance solutions, ensuring prompt incident responses and optimized resource usage within smart building environments.*

1. Introduction

Surveillance cameras are extensively employed for real-time monitoring and have demonstrated effectiveness in substantially reducing crime rates in supervised areas [Chen et al. 2023, Piza et al. 2019]. The global surveillance camera market is projected to surpass 62 billion USD by 2027, reflecting their growing importance and value across various sectors [Laricchia 2024]. This expansion is propelled by advances in intelligent systems that not only analyze real-time data and video streams but also identify potential anomalies and autonomously make decisions, enhancing security and operational efficiency [Ibrahim 2016].

However, deploying such intelligent surveillance solutions in buildings demands significant computational power, often reliant on power-hungry and costly GPUs

[0]Corresponding author: anhngt@huit.edu.vn

[Cob-Parro et al. 2021]. Moreover, as the workload (quantified by the volume of images or frames processed per second) escalates, so do the computational requirements to sustain performance [Brito et al. 2024].

Edge and Fog computing paradigms have emerged as transformative approaches to address the needs of next-generation applications, particularly those with stringent low-latency requirements. By bringing computation closer to the data source, these technologies minimize transmission delays and allow the creation of scalable and autonomous monitoring systems [Khan et al. 2019, Srirama 2024]. Nevertheless, implementing Edge and Fog computing in smart buildings is not without difficulties, including high initial installation costs, the necessity of technical proficiency, and challenges in accurately sizing and scaling resources. Underestimating or overestimating capacity may lead to inefficiencies, unnecessary expenses, or compromised system reliability [Hurbungs et al. 2021, Fé et al. 2024]. Analytical modeling has proven effective in anticipating these issues, enabling scenario-based simulations to optimize resource allocation, mitigate risks, and improve overall system performance [Rodrigues et al. 2021].

Stochastic Petri Nets (SPNs) provide a rigorous mathematical framework for modeling and analyzing complex distributed systems where concurrency, synchronization, and queuing phenomena intertwine. By associating exponential or general distribution firing times to transitions, SPNs capture the probabilistic behavior inherent in real-world surveillance settings. In this study, SPNs are employed to represent the interplay between Edge and Fog layers, evaluating system performance under varying configurations of queue sizes, service rates, and computational resources. Such an approach not only elucidates bottlenecks and resource utilization patterns but also underscores how the synergy between advanced modeling and distributed architectures can optimize intelligent camera surveillance in modern smart buildings.

Key Contributions:

- *Development of a Stochastic Petri Net (SPN) model for Edge-Fog surveillance:* We formulate a mathematically grounded SPN model that captures the key aspects of camera-based surveillance in intelligent buildings, accounting for concurrency, queuing behavior, and service delays.
- *Performance Metrics and System Scalability:* Our approach quantifies mean response time, drop probability, throughput, and resource utilization under varying core counts, arrival rates, and queue capacities, providing a holistic view of system scalability.
- *Experimental Validation:* The SPN model is validated against real experiments run on three physical machines, confirming its accuracy and bolstering confidence in its applicability to real-world deployments.

Novelty of the Research: While various efforts have highlighted the potential of Edge-Fog computing for low-latency applications, few studies focus on rigorous performance planning and system dimensioning for camera-based surveillance in smart buildings. By integrating an SPN-based methodology with experimental validation, our research addresses this gap, offering a robust blueprint for balancing computational workloads between Edge and Fog layers and ensuring reliable coverage of security-intensive environments.

Key Findings: A critical insight from our experiments and modeling indicates

that the choice of core allocation in the Fog layer profoundly affects the system's ability to handle higher arrival rates without significant increases in response time or discard probability. Specifically, scaling up to 10 Fog cores yields a discard probability near 35% at 47.37 msg/ms, while maintaining average response times below 10 ms for moderate arrival rates (up to 29 msg/ms). This underscores the importance of careful resource planning to attain operational efficiency and maintain real-time responsiveness.

In this work, we incorporate Edge and Fog computing paradigms into a surveillance framework, focusing on distributed data processing to improve operational agility and system responsiveness. Edge computing executes data processing at the network periphery, close to the source of data generation [Kong et al. 2022], while Fog computing interposes a distributed layer between Edge and Cloud, further decentralizing processing and storage [Laroui et al. 2021]. Although various studies emphasize the implementation of Edge and Fog-based systems, few concentrate on planning aspects, such as resource dimensioning and how these configurations directly influence system performance outcomes.

This study distinguishes itself by introducing a methodology that integrates experimental validation and modeling techniques to accurately represent real scenarios. We propose a Stochastic Petri Net (SPN) model tailored to evaluate key performance indicators—mean response time (MRT), drop probability (DP), system throughput (TP), and resource utilization—in surveillance systems leveraging Edge and Fog computing [Silva et al. 2024, Sabino et al. 2024]. SPN models are particularly adept at capturing concurrency, synchronization, and intricate system dynamics. By validating the model with experiments on three physical machines, we ensure its fidelity and practical applicability. These insights enable more informed decisions for planning, resource allocation, and scaling in surveillance solutions.

Ultimately, this research contributes to more robust, cost-effective, and adaptable surveillance infrastructures in intelligent buildings, capable of accommodating growing demands and ensuring prompt and efficient responses to security threats.

Remainding Content of the Paper: The remainder of this paper is organized as follows. In Section 2, we review the relevant literature, highlighting current approaches in camera-based surveillance and their limitations. Section 3 describes the proposed system architecture, focusing on how Edge and Fog layers interoperate. Section 4 introduces our Stochastic Petri Net (SPN) model, detailing its structure, workflow, and metrics. We validate this model using real experiments in Section 5, ensuring accuracy and practical viability. Section 6 presents a case study that explores the impact of varying key parameters on overall system performance. Finally, Section 7 concludes the paper, summarizing our main contributions and discussing potential directions for future research.

2. Related Work

This section reviews recent literature on performance evaluation within camera-based surveillance systems, focusing on studies published in the last five years. We exclude papers centered on machine learning or image processing metrics, as our primary goal is to benchmark system performance rather than image-level analysis. Instead, we highlight works that address system architectures, performance metrics, capacity considerations, validation methodologies, and evaluation approaches, as summarized in Table 1.

Tabela 1. Related Studies

Work	System Architecture ◇	Metrics	Processing Capacity Analysis	Validation	Evaluation Method
[Sharifi et al. 2021]	Edge Computing*, Cloud Computing*	Processing Rate, Performance Levels of Processors, Avg. Computational Times	×	✓	Queue Model
[Usmanova et al. 2023]	Cameras**	Average Delay, Jitter, Packet Transit Time	×	✓	Simulation Model
[Kim and Jeong 2023]	Cameras**	Successful Detection Time, Failure Detection Time, Surveillance Resolution	×	×	Simulation Model
[Singh and Singh 2023]	Cameras**, Fog*	Delay, Network Usage, Energy Consumption	×	×	Simulation
[Borges et al. 2023]	Cameras**	Energy Consumption, Utilization, Network Traffic, Storage	✓	×	Measurement
[Kim et al. 2024]	Cameras**	MRT, Utilization	✓	×	Measurement
[Sabino et al. 2024]	UAVs**, Cameras**, Edge Computing*	MRT, Utilization, Throughput, Drop Probability, CDF, MTTA	✓	×	SPN Model
[Zhou et al. 2021]	UAVs**	Delay, Throughput	×	×	Measurement
[Cui et al. 2020]	Cameras**, Cloud Computing*	Average Arrival Rate, CDF, Delay	×	✓	Queue Model
This Work	Cameras**, Edge Computing*, Fog Computing*	MRT, Utilization, Throughput, Drop Probability	✓	✓	SPN Model

Note ◇: — Computational Paradigms (*) — Device Types (**).

Earlier works have explored various system components including cameras, edge, fog, and cloud computing, as well as unmanned aerial vehicles (UAVs). Our proposed approach integrates both surveillance cameras and Edge-Fog architectures, focusing on efficient and low-latency data processing. Unlike works that address isolated components [Usmanova et al. 2023, Kim and Jeong 2023], this approach captures how different system layers interact.

Performance metrics—such as MRT, utilization, drop probability, throughput, and delay—are critical for comprehensively understanding system behavior. While prior studies have examined some of these metrics [Singh and Singh 2023, Cui et al. 2020], our work extends the analysis to incorporate a wide range of them simultaneously. The ability to manipulate processing capacity, considering core counts and queue sizes, is another key aspect that sets our work apart.

Finally, we rely on SPN modeling combined with experimental validation, an approach that is less common than simulation or direct measurements. While simulation and measurement approaches [Borges et al. 2023, Kim et al. 2024] provide insights, SPN models provide a mathematically rigorous representation of concurrency and synchronization. While the work by [Sabino et al. 2024] also uses SPN to evaluate the performance of an architecture, it focuses on a different context. While our study analyzes a surveil-

lance camera-based monitoring infrastructure, [Sabino et al. 2024] evaluates a fire monitoring architecture using drones. Furthermore, they did not perform an experimental validation, making our approach more robust by aligning the theoretical models with real experiments. By validating the SPN model against real experiment data, our work increases confidence in the predictive power of the model, ensuring that the observed behaviors align with practical implementations.

Research Gap: Despite the breadth of studies addressing either camera-centric architectures or Cloud-based offloading for intelligent surveillance, there remains a gap in research that methodically analyzes how resource provisioning in Fog layers directly impacts end-to-end performance for real-time video processing. Most existing works emphasize either algorithmic efficiency or single-layer improvements, without a holistic approach to sizing both Edge and Fog layers under fluctuating workloads. In response, this study presents a Stochastic Petri Net model that captures multi-layer concurrency effects and rigorously evaluates metrics like Drop Probability and Mean Response Time. In doing so, it bridges the gap by offering a validated analytical framework for dimensioning Edge-Fog surveillance systems according to practical performance goals.

3. System Architecture

This section describes the proposed architecture for modeling and analyzing the performance of a camera-based surveillance infrastructure in smart buildings. Figure 1 depicts a layered framework that integrates Edge and Fog computing to achieve efficient and scalable image processing. Although both Edge and Fog layers are modeled, the framework can adapt to scenarios where only the Edge layer is available. The architecture requires a distributed computing environment with data-generating devices, processing nodes, and communication links, supporting a variety of configurations.

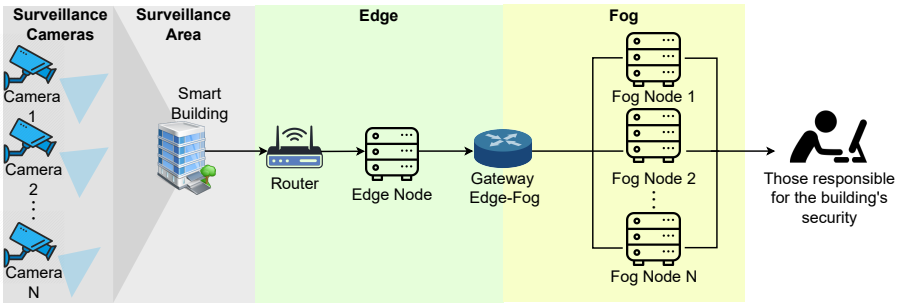


Figura 1. Proposed Edge-Fog Architecture

The architecture comprises five layers. The first (device layer) involves cameras capturing real-time images. The second layer manages the transmission of data from the cameras to the Edge via a router. The Edge (third) layer is the intermediate processing layer between the data-generating devices (cameras) and the Fog layer, responsible for receiving and pre-processing these images before routing them through a gateway to the Fog layer. The Fog (fourth) layer performs the main image processing tasks using multiple containers that distribute the workload among the processing nodes. The fifth layer focuses on delivering processed results or alerts to the building security team.

Surveillance cameras positioned throughout the building feed video streams to the Edge device via a router.These cameras reflect a continuous surveillance system, where

data is transmitted uninterruptedly to the processing infrastructure. In this way, the system maintains a constant flow of data between the Edge and Fog layers, enabling continuous monitoring. The Edge then passes this data through a gateway to Fog layer containers, which handle advanced analytics tasks such as anomaly detection [Saini et al. 2016], crowd counting [Patwal et al. 2023], or facial recognition [Khairuddin et al. 2021]. Each container processes data independently, allowing flexible scalability. The processed information is ultimately transmitted to security managers, via automated alerts or other reporting mechanisms.

To simplify modeling, certain assumptions are adopted: all cameras collectively contribute to the workload as a single aggregated source, complex load balancing is not implemented (assuming identical Fog nodes), and network congestion or hardware failures are not considered. These simplifications emphasize the core aspects of performance analysis—processing power and latency—while making the model manageable. Future research may refine these assumptions to incorporate more nuanced factors and heterogeneous configurations.

4. Proposed Modeling Approach

4.1. SPN Model Description

Figure 2 presents the SPN model for the proposed architecture. This model is structured into five layers corresponding to the system's operational flow: data ingress (cameras), Edge layer, transmission, Fog layer, and output.

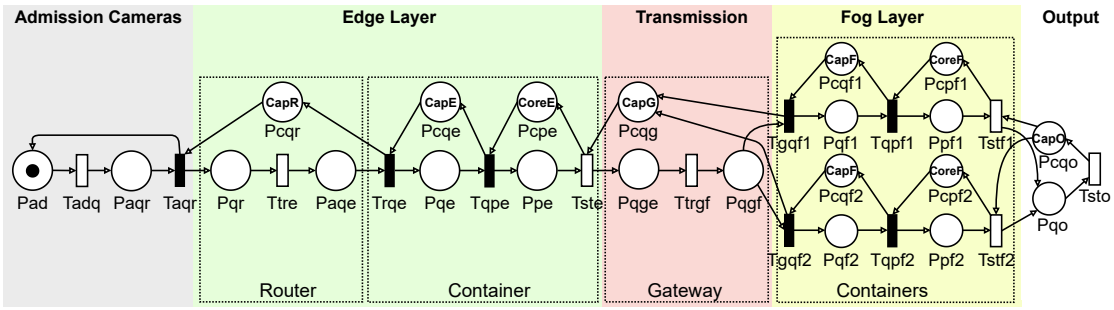


Figura 2. Proposed SPN Model

The naming conventions for places (`P`) and transitions (`T`) remain systematic, with markers (`C`) indicating maximum queue capacities. Acronyms signify their roles, e.g., "ad" for admission, "a" for arrival, "q" for queue, "r" for router, "p" for processing, "g" for gateway, "e" for Edge, "f" for Fog, "o" for output, "tr" for transmission, "st" for service time, "Cap" for capacity, and "Core" for processing.

The model flow is as follows:

- **Admission Cameras:** New data requests arrive at place `Pad`, with inter-arrival times governed by `Tadq`. Each token represents a camera frame or image requiring processing.
- **Router (Edge Layer):** Requests proceed to `Paqr`, then to the router's queue `Pqr`, constrained by capacity `CapR`. The transition `Ttre` represents the time needed to move frames from the router to the container in the Edge layer.

- **Container (Edge Layer):** After leaving `Pqr`, tokens move to `Paqe` and eventually enter `Pqe`, the queue for the Edge container. The marker `Pcqe` and the timed transition `Tste` reflect how many tokens the container can process concurrently and how long each processing takes. Additionally, `Pcpe` represents the processing stage of the Edge container, where tasks are actively handled before being forwarded to the next stage. The number of cores (`CoreE`) in the Edge dictates parallelism.
- **Gateway (Transmission Layer):** Once processed at the Edge, frames pass through the gateway queue `Pqge` and `Pqgf`, governed by `CapG`. The transition `Ttrgf` captures the time to forward data from Edge to Fog.
- **Containers (Fog Layer):** In the Fog layer, tokens enter `Pqf1` or `Pqf2`, each with its own capacity (`CapF`) and corresponding transitions `Tstf1` and `Tstf2` for processing. The places `Ppf1`, `Ppf2` represent processing in the Fog's containers. The parameter `CoreF` controls parallelism within Fog.
- **Output:** Finally, the processed frames proceed to place `Pqo`, and the transition `Tsto` models the last step of sending alerts or final processed data, concluding the entire workflow.

By systematically configuring capacities (`CapR, CapE, CapF, CapO, CapG`) and service times (`Tste, Tstf1, Tstf2, Tsto`), this SPN model supports a wide range of experimental scenarios, providing insights into potential bottlenecks, discard rates, and mean response times under diverse workload conditions.

4.2. Metrics

In the realm of queueing theory and distributed systems analysis, four performance metrics frequently serve as vital indicators of system health and efficiency: *Drop Probability*, *Throughput*, *Mean Response Time*, and *Utilization*. Generally, these can be derived from established theoretical foundations:

- **Drop Probability (DP):** Often interpreted as the fraction of requests not admitted into the system due to lack of capacity. In many queueing formulations, if p is the probability of encountering a full queue state, the system drops new arrivals when no buffer space remains.
- **Throughput (TP):** Defined as the rate at which requests are served. From the perspective of birth-death processes in queueing theory, if λ is the arrival rate and μ is the service rate per server, the throughput can approach λ in stable systems, but is often capped by the service capacity $\mu \times$ (number of servers).
- **Mean Response Time (MRT):** By Little's Law, $L = \lambda \times W$, where L is the average number of tasks in the system, λ is the arrival rate, and W is the average waiting time. Extending this to response time $T = W + S$ (where S is the average service time), we often have $T = \frac{L}{\lambda}$.
- **Utilization (UN):** Represents the fraction of time servers or resources are actively in use. If ρ denotes utilization, in a single-server queue with arrival rate λ and service rate μ, $\rho = \frac{\lambda}{\mu}$. In multi-server or multi-core scenarios, this generalizes to $\rho = \frac{\lambda}{m \times \mu}$, where m is the number of servers or cores.

These theoretical constructs provide a groundwork for understanding how incoming requests are processed, queued, and served in a distributed architecture. In our SPN

approach, the token movements and timed transitions map naturally to these metrics, enabling precise calculations of the system's operational state under varying workload conditions.

Drop Probability (DP): Equation 1 calculates the DP, indicating the likelihood of requests being discarded due to full queues or insufficient resources. In this system, DP specifically evaluates the probability of dropping at the router stage.

$$DP = P((Paqr > 0)AND(Pcqr = 0)) \times 100 \tag{1}$$

Throughput (TP): Equation 2 defines TP as the rate at which requests are successfully processed by the system. It is computed from the expected number of tokens at the output queue divided by the service time.

$$TP = \frac{Esp(Pqo)}{TempoEmTstb} \tag{2}$$

Mean Response Time (MRT): Equation 3 employs Little's Law to determine MRT, linking the average number of tasks in the system with arrival rate and average response time.

$$\begin{aligned} MRT = \frac{1}{TP} \times (&Esp(Paqr) + Esp(Pqr) + Esp(Paqe) + Esp(Pqe) + Esp(Ppe) \\ &+ Esp(Pqge) + Esp(Pqgf) + Esp(Pqf1) + Esp(Pqf2) + Esp(Ppf1) \\ &+ Esp(Ppf2) + Esp(Pqo)) \end{aligned} \tag{3}$$

Utilization (UN): Equation 4 calculates utilization as the proportion of available resources actively in use.

$$UN = \frac{Esp(Ppf1)}{CoreF} \times 100 \tag{4}$$

5. SPN Model Validation

This section details the validation of the SPN model through a real experiment, confirming the model's predictive accuracy and applicability. The validation methodology involves comparing Mean Response Time (MRT) results from both the model and a controlled physical setup.

5.1. Experiment Architecture

Figure 3 illustrates the experimental environment, consisting of three interconnected machines representing the data source, Edge layer, and Fog layer. Measurements of MRT across various configurations enable a direct comparison between model predictions and empirical outcomes.

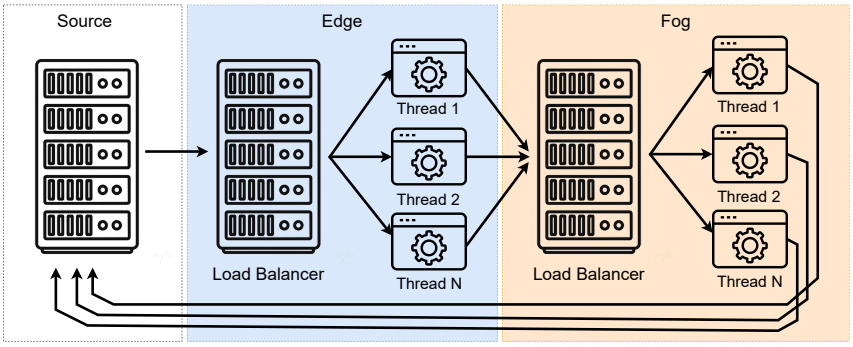


Figura 3. Validation Architecture

All machines share identical hardware (Intel Core i3 3.7GHz, 8GB RAM, Linux OS). The data source generates incoming requests, the Edge machine routes and performs initial processing, and the Fog machine handles more resource-intensive tasks. To simulate a surveillance system, the data source component generates streaming video traffic by continuously producing image frames at predefined intervals. Each frame is represented as a message that is transmitted to the Edge layer, mimicking the real-time flow of video feeds in a smart surveillance system.

5.2. Model Used in Validation

A simplified SPN model (Figure 4) omits parallel components and final output layers for clarity. This model focuses on the essential pipeline of request generation, queuing, and Fog processing.

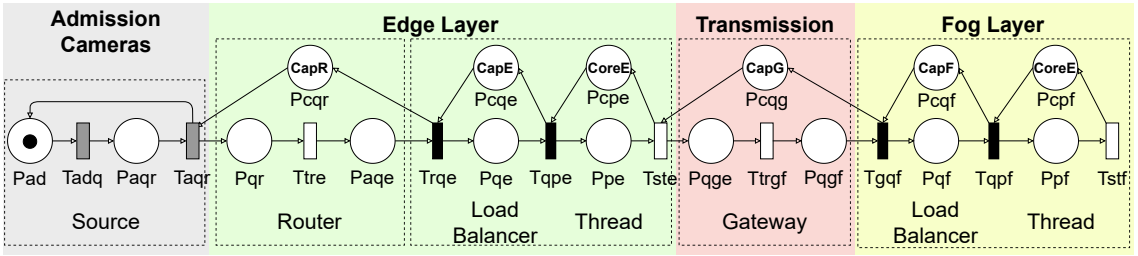


Figura 4. Simplified Validation Model

5.3. Validation Results

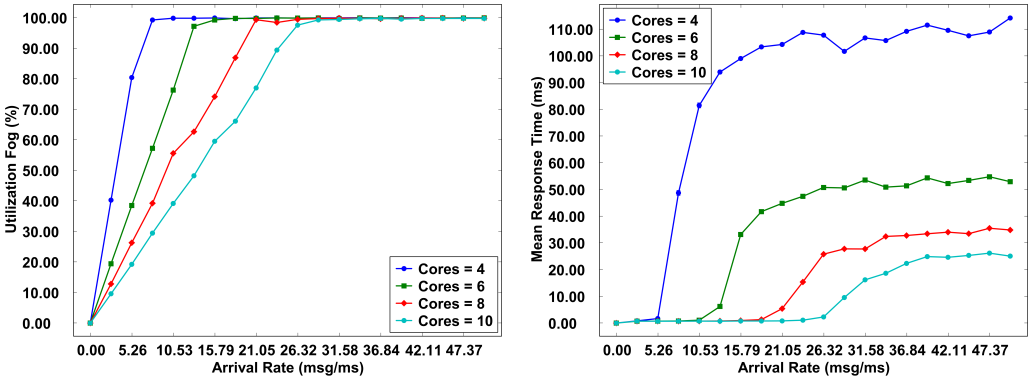
The MRT from the model is contrasted with measurements taken under equivalent arrival rates on the physical setup. Tests with 2, 3, and 4 Fog cores show the model's MRT remains within the confidence intervals of the experimental MRT, affirming its accuracy in capturing real-system behavior.

6. Case Study

Table 2 consolidates the main parameters, and Figure 5 presents outcomes obtained by modifying the number of Fog cores. The system was tested with 4, 6, 8, and 10 Fog cores under various arrival rates, highlighting differences in utilization, MRT, drop probability, and throughput.

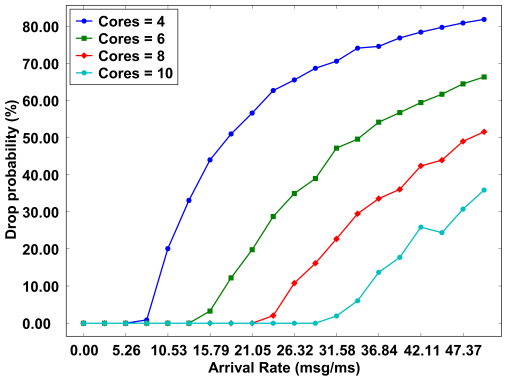
Tabela 2. Key Model Parameters

Type	Components	Value
Timed Transitions	AD	0.02 ms
	Tadq	0.02 ms
	Ttre, Ttrgf	0.01 ms
	Tste	0.04 ms
	Tstf1, Tstf2	0.6 ms
	Tsto	0.05 ms
Place Markings	CapR, CapO	50
	CapE, CapF	200
	CoreE, CoreF	4
	CapG	100

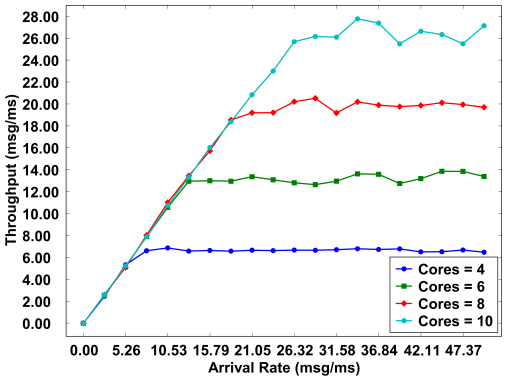


(a) Usage Percentage of Fog Layer

(b) Average System Response Time



(c) System Discard Probability



(d) System Throughput

Figura 5. Fog Core Variation

6.1. Analysis of Results

Fog Utilization The Fog layer's utilization ramps up quickly with higher arrival rates, reaching 100% more rapidly in lower-core configurations. With 4 cores, full utilization occurs around 6 msg/ms, whereas 10 cores can handle arrival rates exceeding 25 msg/ms before saturating. This underscores how parallel processing resources can mitigate bottlenecks under heavier loads.

Mean Response Time Response times (MRT) soar once the Fog layer hits full capacity. At lower core counts, this saturation occurs early, causing a sharp climb in MRT. In contrast, configurations with 8 or 10 cores allow significantly lower MRT values at mid-range arrival rates, offering latency advantages critical in surveillance scenarios.

Discard Probability As arrival rates exceed the system's processing threshold, discard probabilities climb. With 4 cores, drop probabilities can surpass 70–80%. However, at 10 cores, discard probability remains under 40% even at high arrival rates (e.g., 47.37 msg/ms), illustrating the importance of scaling to handle data peaks.

Throughput Throughput initially increases with arrival rate, then plateaus once the system's resources are fully utilized. The 4-core configuration reaches its throughput limit (5–6 msg/ms) sooner. Meanwhile, 10 cores achieve higher throughput (26 msg/ms) before hitting saturation, demonstrating improved capacity for demanding workloads.

6.2. Implications

Although these results already point to the direct advantages of adding more processing cores to the Fog layer, the broader implications for intelligent surveillance in smart buildings are multifaceted and particularly impactful:

- **Enhanced Real-Time Responsiveness:** By substantially lowering average response times under moderate-to-high arrival rates, configurations with a greater number of Fog cores ensure faster detection of security-relevant events. This heightened responsiveness is crucial for scenarios such as real-time anomaly detection or crowd management, where delays can lead to escalated risks or missed opportunities to intervene.
- **Scalability and Future-Proofing:** The ability to scale cores within Fog containers means that system architects can plan for a growing number of cameras or increasing data resolution without overhauling the architecture. As the number of cameras or the resolution of captured video grows, so does the volume of data. Having a Fog layer that can be incrementally scaled ensures the system can handle future demands while preserving performance targets.
- **Resource Utilization and Cost Efficiency:** While adding cores clearly benefits throughput and response time, each additional core represents a cost—both in hardware investment and operational overhead (energy consumption, maintenance, etc.). The results indicate that moderate levels of utilization can be maintained up to relatively high arrival rates with sufficient Fog resources. However, system designers must weigh the cost savings of fewer cores against the risks of increased discard probability or degraded response times. This trade-off becomes particularly relevant for budget-constrained or energy-sensitive deployments, where overprovisioning might be impractical.
- **System Robustness to Traffic Spikes:** In many real-world deployments, the arrival rate of camera frames can exhibit significant bursts (e.g., numerous triggers from multiple camera feeds simultaneously). Configurations with higher Fog cores demonstrate a delayed onset of saturation, thus reducing the likelihood of losing critical footage or incurring prohibitive response times during sudden spikes.

This resilience ensures the system continues to operate effectively even under transient surges in workload, which is vital for security-critical applications.

- **Operational Continuity and Reliability:** By maintaining lower discard probabilities even at high loads, the system supports more continuous video streams. In security contexts, dropped frames or missed detections can result in incomplete situational awareness. Consequently, a sufficiently scaled Fog layer not only improves performance metrics but also contributes to overall reliability—reducing blind spots in monitoring and enhancing trust in automated surveillance.
- **Design Guidelines for Edge-Fog Surveillance:** Finally, these findings provide concrete guidelines for designing Edge-Fog architectures. For instance, when the target is to keep the average response time below 10 ms under typical building traffic, a Fog container with at least 8 cores may be recommended. If minimizing discard rates is paramount, scaling to 10 cores becomes more appealing, especially in scenarios with heavy peak loads. Such guidelines help practitioners align resource allocations with defined service-level objectives (SLOs), guaranteeing that the chosen configuration meets real-world operational requirements.

Overall, the capacity to handle heavier workloads with reduced discard rates and lower latency translates into more reliable, timely, and effective surveillance operations. These improvements are significant for environments that require continuous real-time monitoring—such as corporate campuses, hospitals, or high-security government facilities—where the cost of missed events could be very high. Consequently, while the system designer must balance scalability with cost, the insights gained from this SPN-based analysis underscore how a well-provisioned Fog layer can drastically enhance both the performance and resilience of intelligent surveillance in smart buildings.

7. Conclusion

This work proposed an SPN model to analyze the performance of a camera-based surveillance system in intelligent buildings, leveraging Edge and Fog computing. A simplified version of the model was validated against real-world experiments, demonstrating its reliability. Our findings highlight the pivotal role of Fog layer processing capacity. Increasing the number of cores per container reduces drop probabilities, shortens mean response times, and improves throughput. For instance, with 10 Fog cores, the drop probability remained at approximately 35% at an arrival rate of 47.37 msg/ms, while the MRT stayed below 10 ms for arrival rates up to around 29 msg/ms. These insights guide more informed planning and dimensioning of resources, ensuring cost-effective and efficient surveillance solutions in smart buildings. Future work may extend the analysis to more complex configurations, incorporate additional factors (e.g., load balancing, network congestion), explore cloud computing integration as an alternative when Fog resources are unavailable, and conduct real-world tests to further validate and refine the model.

Acknowledgements

This research is part of the INCT of Intelligent Communications Networks and the Internet of Things (ICoNIoT) funded by CNPq (proc. 405940/2022-0) and Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) Finance Code 88887.954253/2024-00.

Referências

Borges, I., Callou, G., and Silva, F. A. (2023). Performance evaluation and energy consumption of a video surveillance system with distributed storage. In *2023 18th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–6. IEEE.

Brito, C., Barbosa, V., Lima, L. N., Rocha, J. W., Araújo, J. M., Lopes, L., Rego, P. A., Sales, M., Callou, G., Fé, I., et al. (2024). Performance evaluation of a video surveillance system using stochastic petri nets for license plate detection on highways. *Journal of Reliable Intelligent Environments*, pages 1–12.

Chen, Z., Lai, Z., Song, C., Zhang, X., and Cheng, J. C. (2023). Smart camera placement for building surveillance using openbim and an efficient bi-level optimization approach. *Journal of Building Engineering*, 77:107257.

Cob-Parro, A. C., Losada-Gutiérrez, C., Marrón-Romera, M., Gardel-Vicente, A., and Bravo-Muñoz, I. (2021). Smart video surveillance system based on edge computing. *Sensors*, 21(9):2958.

Cui, L., Su, D., Zhou, Y., Zhang, L., Wu, Y., and Chen, S. (2020). Edge learning for surveillance video uploading sharing in public transport systems. *IEEE Transactions on Intelligent Transportation Systems*, 22(4):2274–2285.

Fé, I., Nguyen, T. A., Mauro, M. D., Postiglione, F., Ramos, A., Soares, A., Choi, E., Min, D., Lee, J. W., and Silva, F. A. (2024). Energy-aware dynamic response and efficient consolidation strategies for disaster survivability of cloud microservices architecture. *Computing*, pages 1–47.

Hurbungs, V., Bassoo, V., and Fowdur, T. (2021). Fog and edge computing: concepts, tools and focus areas. *International Journal of Information Technology*, 13(2):511–522.

Ibrahim, S. W. (2016). A comprehensive review on intelligent surveillance systems. *Communications in science and technology*, 1(1).

Khairuddin, M., Shahbudin, S., and Kassim, M. (2021). A smart building security system with intelligent face detection and recognition. In *Iop conference series: Materials science and engineering*, volume 1176, page 012030. IOP Publishing.

Khan, W. Z., Ahmed, E., Hakak, S., Yaqoob, I., and Ahmed, A. (2019). Edge computing: A survey. *Future Generation Computer Systems*, 97:219–235.

Kim, W., Kwak, A., Yoo, B., and Ko, H. (2024). Ipfs viewer: Iot surveillance camera system using ipfs and mqtt. In *2024 IEEE International Conference on Consumer Electronics (ICCE)*, pages 1–6. IEEE.

Kim, Y. and Jeong, J. (2023). A simulation-based approach to evaluate the performance of automated surveillance camera systems for smart cities. *Applied Sciences*, 13(19):10682.

Kong, L., Tan, J., Huang, J., Chen, G., Wang, S., Jin, X., Zeng, P., Khan, M., and Das, S. K. (2022). Edge-computing-driven internet of things: A survey. *ACM Computing Surveys*, 55(8):1–41.

Laricchia, F. (2024). Video surveillance camera market size worldwide from 2019 to 2027 (in billion u.s. dollars). Accessed on 08/25/2024.

Laroui, M., Nour, B., Moungla, H., Cherif, M. A., Afifi, H., and Guizani, M. (2021). Edge and fog computing for iot: A survey on current research activities & future directions. *Computer Communications*, 180:210–231.

Patwal, A., Diwakar, M., Tripathi, V., and Singh, P. (2023). Crowd counting analysis using deep learning: A critical review. *Procedia Computer Science*, 218:2448–2458.

Piza, E. L., Welsh, B. C., Farrington, D. P., and Thomas, A. L. (2019). Cctv surveillance for crime prevention: A 40-year systematic review with meta-analysis. *Criminology & public policy*, 18(1):135–159.

Rodrigues, L., Neto, F., Gonçalves, G., Soares, A., and Silva, F. A. (2021). Performance evaluation of smart cooperative traffic lights in vanets. *International Journal of Computational Science and Engineering*, 24(3):276–289.

Sabino, A., Lima, L. N., Brito, C., Feitosa, L., Caetano, M. F., Barreto, P. S., and Silva, F. A. (2024). Forest fire monitoring system supported by unmanned aerial vehicles and edge computing: a performance evaluation using petri nets. *Cluster Computing*, pages 1–21.

Saini, D. K., Ahir, D., and Ganatra, A. (2016). Techniques and challenges in building intelligent systems: anomaly detection in camera surveillance. In *Proceedings of First International Conference on Information and Communication Technology for Intelligent Systems: Volume 2*, pages 11–21. Springer.

Sharifi, M., Abhari, A., and Taghipour, S. (2021). A queueing model for video analytics applications of smart cities. In *2021 Winter Simulation Conference (WSC)*, pages 1–10. IEEE.

Silva, L. G., Brito, C., Cardoso, I., Sabino, A., Lima, L. N., Gonçalves, G., Rocha Filho, G. P., Fé, I., and Silva, F. A. (2024). Desvendando a elasticidade de máquinas virtuais em vanets: Uma estratégia para aperfeiçoar o planejamento de capacidade em rsus. In *Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)*, pages 169–182. SBC.

Singh, P. and Singh, K. D. (2023). Fog-centric intelligent surveillance system: A novel approach for effective and efficient surveillance. In *2023 International Conference on Advancement in Computation & Computer Technologies (InCACCT)*, pages 762–766. IEEE.

Srirama, S. N. (2024). A decade of research in fog computing: Relevance, challenges, and future directions. *Software: Practice and Experience*, 54(1):3–23.

Usmanova, N., Mirzayev, D., Ergashev, F., and Yunusova, D. (2023). Field monitoring application based on video surveillance: Evaluation of system performance. In *E3S Web of Conferences*, volume 443, page 06016. EDP Sciences.

Zhou, H., Hu, F., Juras, M., Mehta, A. B., and Deng, Y. (2021). Real-time video streaming and control of cellular-connected uav system: Prototype and performance evaluation. *IEEE Wireless Communications Letters*, 10(8):1657–1661.