



Performance Evaluation of a Camera Surveillance System in Smart Buildings Using Queuing Models

Lucas Silva Lopes  [Federal University of Piauí | lucaslopes092020@ufpi.edu.br]

José Miqueias Araújo  [Federal University of Piauí | jmiqueias@ufpi.edu.br]


Luiz Nelson Lima  [Federal University of Piauí | luiznelson@ufpi.edu.br]

Vandirleya Barbosa  [Federal University of Piauí | vandirleya.barbosa@ufpi.edu.br]

Arthur Sabino  [Federal University of Piauí | arthursabino@ufpi.edu.br]

Geraldo P. Rocha Filho  [State University of Southwest Bahia | geraldo.rocha@uesb.edu.br]

Francisco Airtton Silva   [Federal University of Piauí | faps@ufpi.edu.br]

 Laboratory of Applied Research to Distributed Systems (PASID), Federal University of Piauí (UFPI), Picos, Piauí, Brazil; R. Cicero Duarte, No. 905 - Junco, 64607-670

Received: 12 October 2024 • **Accepted:** 12 May 2025 • **Published:** 11 August 2025

Abstract Security is increasingly prioritized, driving the use of camera surveillance in various settings such as companies, schools, and hospitals. Cameras deter crime and enable continuous monitoring. Integrating Edge and Fog Computing into these systems decentralizes data processing, allowing for faster responses to critical events. Challenges in deploying such systems include high costs, complex technology integration, and precise sizing. Costs cover cameras, Edge devices, cabling, and software, while integration requires technical expertise and time. Accurate sizing is essential to prevent resource under- or over-utilization. Analytical modeling helps simulate scenarios and calculate needed resources. This work proposes an M/M/c/K queuing model to assess surveillance system performance in smart buildings, considering data arrival rates and Edge and Fog container capacities. The model allows parameter customization to analyze various scenarios. Results show that increasing the number of containers more significantly improves system performance than increasing the number of cores per container.

Keywords: Surveillance Cameras, Smart Building, Edge-Fog computing, Queue Theory

1 Introduction

Security is an increasing priority in a rapidly evolving world. Camera surveillance has become an essential tool across various sectors of modern society [Gracias *et al.*, 2023]. These systems enhance security by enabling real-time monitoring, deterring criminal activities such as theft and vandalism, and facilitating rapid response to incidents [Myagmar-Ochir and Kim, 2023]. However, despite their advantages, traditional surveillance architectures face critical challenges related to scalability, latency, and resource management.

As surveillance systems expand, ensuring their efficiency and scalability becomes increasingly important. The increasing number of cameras deployed in smart buildings and urban areas generates massive amounts of data, requiring robust infrastructures to process and analyze information in real-time. Without efficient resource management, surveillance systems may suffer from delays, data loss, and high operational costs.

The rise of Edge and Fog Computing has introduced new possibilities for improving surveillance efficiency. By decentralizing data processing, these paradigms bring computation closer to the source, reducing latency and enabling real-time analytics. Edge Computing processes data on devices near the cameras [Hossain *et al.*, 2023], while Fog Computing leverages nearby data centers for higher computational capacity. The integration of AI and machine learning in the Fog layer further enhances surveillance capabilities, en-

abling anomaly detection, facial recognition, and behavior analysis [Alsadie, 2024].

Nevertheless, designing and scaling an effective surveillance infrastructure based on Edge and Fog Computing remains a complex challenge. The high costs of installing cameras, processing units, cabling, and software, combined with the technical intricacies of integrating these technologies, create significant barriers to adoption [Moorthy H. *et al.*, 2020; Srirama, 2024]. One of the most pressing issues is resource allocation: improper distribution—whether overprovisioning or underprovisioning—can lead to performance bottlenecks and inflated operational costs. Additionally, there is a notable lack of practical tools to assist system designers in evaluating performance and optimizing resource allocation before deployment.

To address these challenges, this work proposes a parameterizable analytical model specifically designed to support decision making in smart surveillance systems. By incorporating an M/M/c/K queuing model, the approach allows designers to analyze key performance metrics including average response time, drop rate, and resource utilization at the Edge and Fog layers. Utilization refers to the percentage of time that nodes/servers are busy processing requests. The model allows for tuning of critical parameters such as arrival rate, number of processing cores per container, number of containers in each processing layer, and serving policies, providing valuable insights for system optimization.

Recent research has explored performance evaluation in

video surveillance systems through modeling, simulation, and experimental testbeds. Prior studies have examined Edge and Cloud Computing architectures [Sharifi *et al.*, 2021], traffic patterns in IP cameras [Usmanova *et al.*, 2023], and multi-camera modeling techniques [Kim and Jeong, 2023]. Others have investigated drone-based surveillance with Edge processing [Sabino *et al.*, 2024] and video content delays in Cloud environments [Cui *et al.*, 2020]. Additionally, simulation approaches, such as the Fog-Centered Intelligent Surveillance System (FISS) [Singh and Singh, 2023], have been proposed. Experimental testbed studies have assessed the energy consumption of surveillance storage systems [Borges *et al.*, 2023], introduced IoT-based surveillance solutions with IPFS and MQTT [Kim *et al.*, 2024], and evaluated UAV-based surveillance deployments [Zhou *et al.*, 2021].

This work introduces an analytical approach to complement existing methodologies. By providing a systematic method for evaluating infrastructure tradeoffs prior to deployment, our approach fills a critical gap in the design of surveillance systems using containers. Simulation results indicate that increasing the number of containers significantly improves response time and drop rate compared to merely increasing processing cores. Moreover, balanced resource allocation between Edge and Fog layers proves more effective in mitigating bottlenecks and enhancing system efficiency.

The main contributions of this work are summarized as follows:

- **Queue Model:** A parameterizable queuing model is proposed for surveillance system architectures in smart buildings leveraging decentralized processing. It serves as a practical tool for system designers to evaluate the impact of architectural changes on performance—such as resource allocation and workload distribution—before actual deployment.
- **Resource Capacity Analysis:** The proposed model provides insights into performance evaluation in a camera surveillance scenario within an intelligent building. We analyze two key scenarios: mean image processing time and drop rate.

The remainder of this paper is organized as follows: Section 2 presents fundamental concepts related to queueing theory. Section 3 highlights related work. Section 4 presents the evaluated architecture and some assumptions. Section 5 details the proposed queue model. Section 6 analyzes the simulation results. Finally, Section 7 discusses the conclusions of this work.

2 Background

Queueing theory provides a mathematical framework for analyzing systems where jobs or requests are waiting to be processed. It allows for the evaluation of resource allocation and performance optimization by modeling arrival processes, service conditions, and processing times [Dudin and Dudina, 2023]. Such models are widely applied in fields such as telecommunications, computer networks, and customer ser-

vice operations to assess efficiency and identify potential bottlenecks.

A call center, for example, can be represented as a queuing system where incoming calls are managed by a set of service agents. These service nodes handle tasks such as customer inquiries, technical support, and transaction processing. The efficiency of these processing units directly impacts the overall performance of the system. Figure 1 illustrates this representation of a call center in a queuing model.

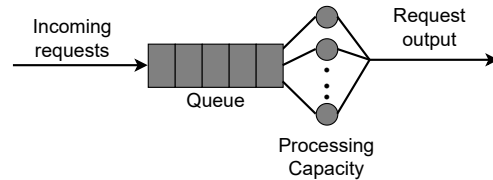


Figure 1. Call center (waiting line + one or more centers).

A queuing network model is a mathematical framework used to represent systems where multiple queues and service nodes are interconnected. This approach is useful in distributed computing environments, cloud infrastructure, and large-scale customer service operations [Hasan *et al.*, 2024]. Each queue represents a waiting process, while service nodes cover the processing units responsible for handling incoming jobs. A modern queuing network can be open, closed, or mixed systems, depending on how jobs enter and exit the system [Narmadha and Rajendran, 2024].

The analysis of these models involves important performance statistics, including queue length, wait time, service utilization, and system throughput. For example, in a cloud computing environment, a queuing network can help determine how efficiently computing resources are allocated and identify potential bottlenecks that interfere with system performance [Singh *et al.*, 2025]. Mathematical formulations such as Little’s Law are used to obtain system performance information [Ogumeyo and Omole, 2023].

Queueing models analyze system performance by evaluating metrics such as the number of customers, throughput, and service center utilization, which measures server efficiency as a percentage of busy time [Ding, 2023]. By simulating different scenarios, these models help identify bottlenecks, optimize resource allocation, and improve efficiency. Advanced approaches incorporate factors like priority levels, service time distributions, and dynamic routing, making them essential for both theoretical research and practical system design.

2.1 Performance Metrics

This subsection outlines the performance metrics employed to evaluate queuing models. The study utilizes three metrics to gauge system performance: mean response time (MRT), resource utilization, and drop rate—equation 1 defines the utilization of system components. ρ represents the utilization of the component, which is the fraction of time that the component is occupied. λ denotes the average arrival rate of jobs or customers to the system, indicating the number of jobs or customers arriving per unit of time. c is the number

of servers or components in parallel within the system. μ stands for the average service rate of each server or component, reflecting the number of jobs or customers that can be serviced per unit of time by each server.

$$\rho = \frac{\lambda}{c\mu} \quad (1)$$

Equation 2 defines the mean response time. MRT represents the mean response time, which is the average time a job or customer spends in the system, including both waiting and service times. μ denotes the average service rate of each server or component. ρ is the utilization of the component.

$$MRT = \frac{1}{\mu(1 - \rho)} \quad (2)$$

3 Related Works

This section presents related work on performance evaluation in the context of video surveillance, considering works published in the last five years. Recent works in machine learning or image processing were disregarded as they focus on image-related performance metrics rather than system performance, which could lead to an unfair comparison. Papers were selected based on five criteria: evaluation method, components used, metrics, processing capacity analysis, and representation of the number of containers per layer. The papers were divided into three groups according to the evaluation method: analytical queueing model, simulation, and Experimental Testbed. Table 1 presents related works and their comparison criteria.

Performance evaluation with Model The first group of works includes models for performance evaluation. [Sharifi et al., 2021] propose a queueing model for scheduling tasks in video analysis applications in smart cities, with Edge and Cloud Computing, showing lower computational time and accuracy in processor operation. [Usmanova et al., 2023] develop a simulation model for IP video surveillance systems, analyzing camera traffic parameters to choose suitable equipment and ensure quality and reliability. [Kim and Jeong, 2023] presents a simulation model that evaluates the coverage of multiple camera surveillance systems, highlighting the economic viability of sophisticated controls. [Sabino et al., 2024] propose Stochastic Petri Net models for systems with drones, cameras, and Edge Computing, showing that adjustments in the Edge layer increase efficiency. [Cui et al., 2020] suggest a WFP algorithm to balance bandwidth and predict workload, demonstrating superiority in experiments with Shenzhen data.

Performance evaluation with simulation The second grouping includes the study that used simulation to evaluate the performance of a video surveillance system. The study by [Singh and Singh, 2023] proposes a FISS System, which combines Fog Computing and intelligent algorithms to improve the efficiency and effectiveness of surveillance. The iFogSim tool was employed to simulate different network configurations and evaluate parameters such as

latency, bandwidth usage, and power consumption. The results demonstrate that FISS features low latency, lower power consumption, and a significant reduction in network traffic compared to approaches that rely exclusively on cloud-based infrastructures.

Performance evaluation with mensuration The third grouping includes studies that used Experimental Testbed to evaluate performance. [Borges et al., 2023] investigate the impact of the quantity and storage method of surveillance cameras on electrical consumption and system efficiency. [Kim et al., 2024] propose an IoT-based surveillance camera system using IPFS and MQTT, evaluating performance in terms of CPU utilization, memory, and response time. [Zhou et al., 2021] developed an experimental laboratory with UAVs connected to LTE cells to improve communications reliability, coverage, throughput, and security.

Contributions of this Work This work stands out in the field of surveillance systems for intelligent buildings by using an M/M/c/K queueing model, a less common approach among related works, which often rely on other types of models, simulations, or Experimental Testbeds. The modeling captures the dynamics of processing and computational times in distributed computing systems, and the use of queueing models is found in the literature, reinforcing the relevance and applicability of the results obtained. The choice of this model is crucial as it allows for a rigorous mathematical analysis of the system's behavior under different load conditions, providing more accurate predictions about the system's performance. SPN modeling offers a detailed probabilistic approach; however, it is more suitable for more complex systems than the one addressed in this study. Simulation allows for a detailed analysis of system behavior, such as cameras and networks, without the need for physical experimentation, but it may need more mathematical precision of queueing models. Experimental Testbed provides accurate empirical data, essential for practical validation, but it requires controlled experimental environments.

The components analyzed include Edge, Cloud, and Fog Computing, as well as cameras, UAVs, and buses. The integration of cameras with both Edge and Fog layers represents a key innovation, as many prior studies—such as [Usmanova et al., 2023; Kim and Jeong, 2023] focus on isolated components, which may lead to less comprehensive evaluations. In contrast, architectures that combine these technologies aim to optimize processing efficiency and network usage in surveillance and monitoring scenarios. The mapped metrics include MRT, utilization, drop rate, energy consumption, throughput, drop probability, and others such as delay, jitter, and success rate in detection. MRT, utilization, and drop rate are fundamental to evaluating the overall performance of distributed systems, especially in high-demand contexts. Unlike studies such as [Cui et al., 2020] and [Zhou et al., 2021], which do not explore processing capacity, our research addresses this analysis, which is essential to optimizing performance and avoiding bottlenecks that could compromise system operation. This analysis in some studies is necessary for the understanding of the total impact of the workload on the

system, especially in contexts where processing is crucial for the application's success. Additionally, the explicit representation of the number of containers per layer offers a level of flexibility in resource allocation that facilitates system adaptation to different scales of operation, promoting efficiency in resource usage.

4 Architecture

Figure 2 presents the architecture used for modeling and performance analysis of an infrastructure designed for intelligent building surveillance by cameras. This architecture integrates Edge and Fog layers to ensure efficient and scalable processing of the captured images. The architecture consists of four distinct layers: surveillance cameras, surveillance area, Edge, and Fog.

The surveillance cameras are strategically installed at specific points in the smart building, as planned by the security team. These locations are chosen to maximize coverage and ensure all critical areas are monitored. Each camera captures images at a specific frame rate, determined by the security requirements and bandwidth availability, and sends them to the Edge layer. The cameras are equipped with network connectivity to transmit real-time video feeds to the processing layers.

The surveillance area refers to the physical spaces within the smart building that are under constant observation. This includes entrances, corridors, stairwells, and other vulnerable points. The cameras in this area are continuously operational, providing a stream of data that is crucial for maintaining security and monitoring activity patterns. The surveillance area forms the foundation of the architecture, feeding raw data into the processing pipeline.

The Edge layer is responsible for initial data processing and acts as an intermediary between the cameras and the deeper processing layers. The Edge layer forwards the images to the Edge-Fog gateway, which acts as the entry point for data distribution and load balancing to the Fog nodes. The Edge layer performs lightweight processing tasks such as image compression, noise reduction, and preliminary feature extraction (e.g., detecting motion or identifying large objects within the frame). Minor processing tasks, such as image compression or preliminary feature extraction, can be performed at the Edge layer to optimize data transmission to the Fog layer [Amshavalli and Kalaivani, 2023]. This reduces the amount of data that needs to be sent over the network, lowering bandwidth usage and improving system responsiveness. The Edge layer is treated as a device within the smart building, meaning it is physically close to the cameras and can quickly handle initial processing tasks.

The Fog layer is where more intensive data processing occurs. This layer utilizes advanced algorithms and techniques to analyze the images for various purposes. The Fog nodes execute computationally tasks such as deep learning-based anomaly detection [Al-Naday et al., 2023], object tracking across multiple frames, fine-grained crowd counting [Patwal et al., 2023], and facial recognition [Khairuddin et al., 2021]. These tasks require more computational power than what is available at the Edge, making the Fog layer crucial for intel-

ligent surveillance applications.

The proposed solution seeks to improve efficiency in contrast to conventional surveillance architectures. The constant transmission of massive video streams causes high latency and excessive bandwidth consumption in cloud-based solutions [Kumar et al., 2024]. Our design is well suited to the surveillance requirements of contemporary smart buildings as it reduces latency and ensures real-time anomaly detection by dividing processing tasks between layers.

Data flows from the cameras to the Edge layer and then to the Fog nodes via the Edge-Fog gateway. This gateway is crucial for load balancing and ensuring that data is efficiently distributed among available resources. After processing, the results are transmitted to the building security personnel through appropriate means, such as email or a web-based monitoring system. This ensures that security staff receive timely alerts and insights necessary for decision-making.

To simplify the modeling, certain assumptions about the architecture were made:

- *Image Capture Modeling:* The model assumes that all active surveillance cameras in the smart building are connected to a device installed within the building. This means every camera contributes to the overall system workload, providing a comprehensive view of the building's security environment.
- *General Scenario Coverage:* The architecture is designed to cover a general scenario where all cameras are operational and feed data into the system. This approach allows the model to simulate a fully active surveillance setup, which is critical for understanding potential bottlenecks and capacity limits.
- *Communication Latency Considerations:* Communication latency between the cameras and the Edge node, as well as between the Edge node and the Fog nodes, was not considered in this model. The latency between the cameras and the Edge layer was deemed insignificant since they are in the same physical environment, minimizing delay. Similarly, the latency between the Edge and Fog layers was omitted to reduce the model's complexity and facilitate analysis. However, this simplification may underestimate the system's mean response time, which is an important consideration for real-time applications.

These assumptions were made to simplify the modeling and performance analysis of the building surveillance infrastructure using cameras. They allow an initial understanding of the system and facilitate the evaluation of its operation in different scenarios. This approach provides a baseline for further studies and potential enhancements that could incorporate more detailed latency models and adaptive resource management strategies.

5 Queue Network Model

Figure 3 depicts a queueing theory-based model for the proposed architecture. This model consists of an input point and an output point. Table 2 describes all elements of the model.

Table 1. Comparative Table of Related Works

Work	Evaluation Method	Used Components	Metrics	Processing Capacity Analysis
[Sharifi <i>et al.</i> , 2021]	Analytical Queueing Model	Edge and Cloud Computing	Processing Rate, Performance Levels of the Processors, Average Computational Times	✓
[Usmanova <i>et al.</i> , 2023]	Event-based Simulation	Cameras	Average Delay, Jitter, Packet Transit Time	×
[Kim and Jeong, 2023]	Event-based Simulation	Cameras	Successful Detection Time, Failure Detection Time, Surveillance Resolution	×
[Singh and Singh, 2023]	Tool-based Simulation	Cameras and Fog Computing	Delay, Network Usage, Energy Consumption	×
[Borges <i>et al.</i> , 2023]	Experimental Testbed	Cameras	Energy Consumption, Utilization, Network Traffic, Storage	✓
[Kim <i>et al.</i> , 2024]	Experimental Testbed	Cameras	MRT, Utilization	✓
[Sabino <i>et al.</i> , 2024]	Stochastic Petri Net (SPN) Simulation	UAVs, Cameras, Edge Computing	MRT, Utilization, Throughput, Drop Probability, CDF, MTTA	✓
[Zhou <i>et al.</i> , 2021]	Experimental Testbed	UAVs	Delay, Throughput	×
[Cui <i>et al.</i> , 2020]	Analytical Queueing Model	Buses, Cloud Computing	Average Arrival Rate, CDF, Delay	×
This Work	Analytical Queueing Model	Cameras, Edge, and Fog Computing	MRT, Utilization, Drop Rate	✓

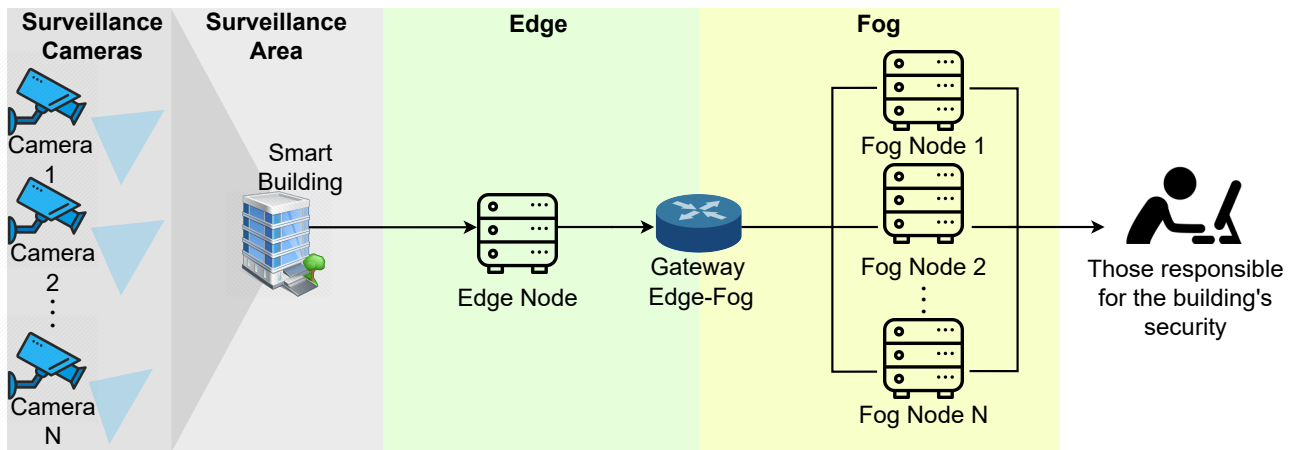


Figure 2. Architecture that Integrates Edge and Fog for Surveillance in Smart Building Using Cameras.

The data flow in the model proceeds from left to right, originating from surveillance cameras installed in the smart building. Each camera sends images at an arrival rate λ to a container represented as a queue with processing cores. Image capture from each camera is treated as an independent process from other cameras. Therefore, image arrivals are exponentially distributed. The drop rule used was “always drop”, where all images exceeding the finite capacity of the queue at the service station are discarded upon arrival.

The queue model illustrated in Figure 3 is central to understanding the data processing workflow within the smart building surveillance system. The model starts with multiple input points corresponding to the cameras deployed throughout the smart building. These cameras are responsible for capturing images continuously and sending these images to the processing system at a defined arrival rate λ . Each camera operates independently, which allows the system to treat each image’s arrival as an independent event. This independence is modeled using an exponential distribution for the

image arrivals, which is typical in queuing systems where events happen continuously and independently over time.

Once the cameras capture images, they enter a queue system where each queue represents a container with processing cores allocated to handle the incoming image data. These containers act as the initial processing nodes, performing basic operations such as filtering and compression. The queue has a finite capacity, meaning it can hold a limited number of images at any given time. Suppose the number of incoming images exceeds this capacity. In that case, the overflow images are dropped, adhering to a drop-tail policy where the newest arrivals are discarded to make space for incoming ones.

The transition from the Edge to the Fog layers is managed by a gateway that acts as the entry point to the Fog Computing resources. This gateway is responsible for implementing a load-balancing strategy to distribute the processing load efficiently across the Fog nodes. The chosen strategy in this work is the least utilization routing, which sends incoming

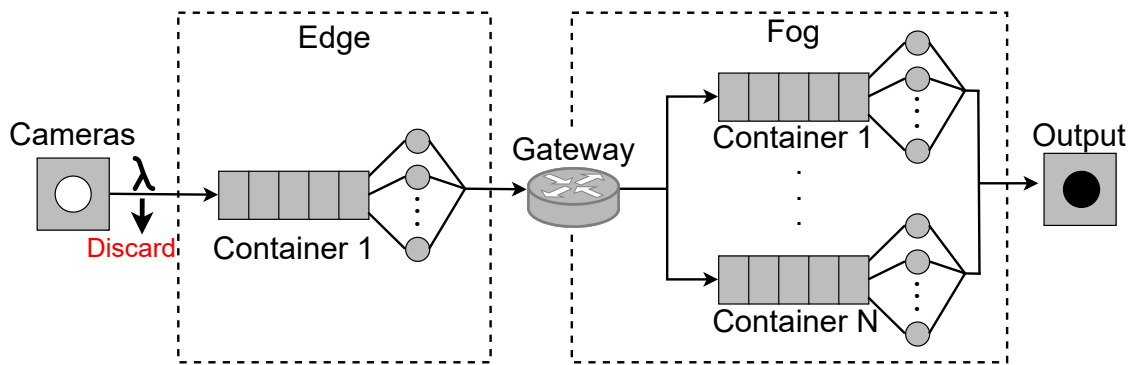


Figure 3. Queue Model of a Surveillance Architecture in a Smart Building Using Cameras.

image data to the container with the lowest current utilization. This strategy ensures that the processing load is evenly distributed and helps prevent any single node from becoming a bottleneck.

The service stations within the model are configured to operate under the Load Independent strategy, with a First Come First Served (FCFS) policy. This configuration means that the service stations process images in the exact order they arrive, without prioritization. The queueing model used is the M/M/c/K model, where M indicates that both arrival and service rates follow an exponential distribution. The parameter c denotes the number of service stations available to handle incoming tasks, and K specifies the maximum number of images the queue can hold.

At the end of the processing pipeline, a sink station, referred to as “Building Security Personnel”, represents the interface through which building security staff access and review the processed data. This sink serves as the endpoint for processed images, providing real-time results that enable security personnel to make informed decisions based on the surveillance data.

In the analysis presented, the architecture’s processing units are modeled as containers, each configured with an equal number of cores across the system’s computing layers. This uniform setup offers design flexibility, allowing containers to be deployed either on a single physical machine or distributed across multiple hosts to optimize resource utilization and expand processing capacity. Moreover, the model supports customization, enabling evaluators to adapt the container configuration according to the specific needs and constraints of the surveillance system being analyzed.

6 Simulation Results

This section presents the simulation results and their analysis. The primary objective of the simulations is to investigate the impact of two distinct scenarios on the performance of surveillance systems in smart buildings: (I) varying the number of containers in the Fog layer and (II) varying the number of processing cores in containers across both the Edge and Fog layers. The number of containers refers to the number of instances of a service or application running in the sys-

tem. This parameter directly affects system performance, as a higher number of containers can better distribute the workload, reducing per-instance utilization, decreasing response times, and mitigating the drop rate by preventing server overload.

The results were measured using three performance metrics: mean response time, utilization rate, and drop rate. The system was stress-tested by adjusting the arrival rate from 2 to 90 frames per second. The Java Modeling Tools (JMT), an open-source tool, were used to model and evaluate the proposed scenarios. JMT is widely recognized for its effectiveness in analyzing and assessing system performance based on queuing theory [Ghandour *et al.*, 2023].

6.1 Methodology Used

Table 3 summarizes the configurations of the base model. Cells with an X indicate that the corresponding component does not have that configuration. The Time (s) column represents the service time for each container in the Edge and Fog layers. The arrival rate indicates the time between the arrival of each image, varying from 0.5 seconds between each image to 0.011 seconds. Table 4 presents the parameters used in both scenarios. The results are detailed in the following subsections.

The simulation parameters were defined based on [Santos *et al.*, 2021], which is grounded in queuing theory and aligns with the methodological approach of this work. The adopted values for processing time and computational resources reflect fog computing environments and enable fair comparisons. The number of simulated cameras is indirectly represented by varying the image arrival rate, covering a range of monitoring scenarios. Routing strategies such as Random and Least Utilization were selected to evaluate trade-offs between load balancing and performance optimization. The results indicate that the Least Utilization strategy provided the best performance in terms of latency and queue management. Although a full factorial design was not applied, the variation of parameters across scenarios approximates such an analysis, and future work will explore this more systematically.

Table 2. Description of Model Components.

Element	Description
Cameras	Responsible for generating images (frames per second)
Edge Node	Responsible for transmitting images to the Edge-Fog Gateway. Besides transmission, it can preprocess images.
Edge-Fog Gateway	Responsible for forwarding images from the Edge to the Fog
Fog Nodes	Responsible for processing images and transmitting the result.
Building Security Handlers	Represents the endpoint of image processing and availability to building security handlers.

Table 3. Base Model Configuration.

Component	Time (s)	Queue Size	Service Policy	Discard Rule	Routing Strategy
Arrival Rate (λ)	[0.5-0.011]	X	X	X	X
Edge Container	0.04s	15	First Come First Served (FCFS)	Drop	Random
Fog Containers	0.111s	70	First Come First Served (FCFS)	Drop	Random
Edge-Fog Gateway	X	X	X	X	Least Utilization

Table 4. Simulation Parameters.

Scenario	Layer	Number of Containers	Number of Cores
Scenario 1	Edge	1	2
	Fog	1/2/3/4/5	2
Scenario 2	Edge	1	1/2/3/4/5
	Fog	2	1/2/3/4/5

6.2 Scenario I - Varying the number of containers in the Fog

In this scenario, we conducted a series of experiments to evaluate the impact of varying the number of containers in the Fog layer while maintaining a fixed configuration in the Edge layer. Specifically, we adjusted the number of containers in the Fog layer from one to five, whereas the number of containers in the Edge layer remained constant at one. This setup was crucial for analyzing how the distribution of computational resources affects system performance metrics such as MRT, utilization, and drop rate.

Figure 4a presents the results for the mean response time (MRT) of the system across different configurations. As anticipated, an increase in resource allocation through additional containers in the Fog layer resulted in a noticeable decrease in MRT, indicating enhanced processing efficiency. Interestingly, the scenarios with three, four, and five containers yielded similar MRT results, suggesting diminishing returns on performance beyond three containers. In contrast, configurations with only one or two containers experienced a significant rise in MRT, with values exceeding three seconds, highlighting the limitations of inadequate resource provisioning.

The close similarity in MRT for scenarios with three, four, and five containers indicates that deploying just three containers might offer a more advantageous and cost-effective solution without sacrificing performance. Figure 4c illustrates the utilization of containers in the Edge layer. The data

shows that variations in the Fog layer did not significantly influence the utilization levels of the Edge layer, which consistently reached 100% utilization across all configurations when the arrival rate neared eighty-one frames per second. This suggests that the Edge layer's capacity was fully utilized, serving as a bottleneck in the data processing pipeline.

Figure 4d provides insights into the utilization patterns within the Fog layer. Higher utilization rates were observed in scenarios with fewer resources, particularly with one and two containers, which reached full utilization (100%). In contrast, configurations with four and five containers did not surpass 80% utilization, demonstrating that excess resources remained underutilized. This observation underscores the relationship between available resources and their efficient utilization. As the arrival rate of frames increased, the impact of resource limitations became more pronounced in scenarios with fewer containers (one and two). In contrast, utilization in configurations with three, four, and five containers stabilized around 90%, even as they processed up to ninety frames per second. This stability suggests that a configuration with three containers can effectively manage workloads up to ninety frames/s, maintaining a utilization rate of approximately 90%. It is worth noting that utilization growth plateaued after sixty frames/s across all configurations, indicating a threshold beyond which additional resources did not translate into improved performance.

Figure 4b depicts the system's drop rate, which measures the frequency of frames being discarded due to insufficient processing capacity. Configurations with one and two containers experienced a higher drop rate, underscoring the challenges of limited resources in handling peak workloads. In contrast, scenarios with three, four, and five containers exhibited a similar drop rate, which was significantly lower than that of the one and two container configurations. This consistency across higher container counts can be attributed to the Edge layer becoming a bottleneck, limiting the overall system's ability to process and transmit data efficiently once the Fog layer's capacity exceeds a certain threshold.

Overall, the analysis reveals that a strategic allocation of resources, specifically with one container on the Edge and three containers in the Fog layer, strikes an optimal balance between performance and cost-effectiveness for handling workloads up to ninety frames per second. This configuration maximizes resource utilization, ensuring lower latency and improved system efficiency, which are critical factors in real-world camera surveillance applications within smart buildings. The reduced MRT achieved with this setup allows for faster identification and response to critical events, such as intrusions or hazardous situations.

Analyzing the results, we conclude that having one container on the Edge and three containers in the Fog offers the best balance of performance and cost for workloads up to ninety frames per second. This configuration optimally distributes processing power, leading to lower latency and higher system efficiency. In real-world camera surveillance applications in smart buildings, the reduced MRT achieved with this setup allows for faster identification and response to events such as intrusions or dangerous situations.

6.3 Scenario II - Varying the number of cores in the Edge and Fog

In this scenario, the number of cores in each container was varied from one to five, both in the Edge layer and in the Fog layer, while keeping the number of containers fixed at one on the Edge and two in the Fog. This setup aimed to evaluate how varying computational resources, specifically the number of cores, impact the overall system performance.

Figure 5a depicts the system's MRT, which serves as a crucial metric for assessing the efficiency of the surveillance system. As the number of cores increased, there was a noticeable decrease in MRT, underscoring the importance of resource allocation in enhancing system responsiveness. Notably, in this scenario, the MRT was reduced to below one second with the allocation of just five cores. This is in contrast to Scenario one, where achieving an MRT below one second necessitated the use of three, four, and five containers in the Fog layer. This finding highlights that while both increasing the number of cores and containers can improve performance, adding more containers exerts a more substantial impact on reducing MRT.

Additionally, the analysis revealed a significant effect of arrival rate on MRT, especially when only one core per container was utilized. The MRT increased markedly with higher workloads, demonstrating a difference of approximately one second between consecutive cases. However, the gap was more pronounced between the one-core and two-core scenarios, where the MRT difference was approximately four seconds. This indicates a non-linear relationship between core allocation and workload capacity, emphasizing the importance of strategic resource distribution based on expected workloads.

Figure 5c illustrates the utilization of the Edge layer, shedding light on resource consumption patterns across different configurations. In scenarios with fewer allocated resources, there was a greater utilization of available resources, with the Edge utilization rate reaching one hundred percent in cases with one, two, and three cores. However, when the num-

ber of cores was increased to four and five, the utilization rates were similar. This suggests that opting for four cores may strike a balance between resource efficiency and performance, offering a cost-effective alternative compared to using five cores without significantly compromising on performance.

Similarly, Figure 5d presents the utilization of the Fog layer, further illustrating how resource constraints influence system performance. In all tested configurations, utilization reached one hundred percent, indicating full capacity usage across the board. Analyzing the impact of increased arrival rates revealed that resource-limited scenarios experienced a more pronounced effect. Notably, with five cores, the system reached one hundred percent utilization at an arrival rate of only ninety frames per second, signifying that five cores are sufficient to manage demands of up to ninety frames per second.

Finally, Figure 5b demonstrates the system's drop rate, a critical metric reflecting the system's ability to handle incoming data without loss. The analysis shows a consistent reduction in drop rate with each increment in core count, with a decrease of approximately fifteen frames per second in drop rate for each additional core. With just five cores, the drop rate was nearly eliminated, approaching zero. This significant reduction in drop rate underscores the efficacy of increasing core count as a means to enhance system reliability and data integrity.

Overall, these findings underscore the nuanced interplay between resource allocation, workload management, and system performance in the context of Edge-Fog Computing for smart building surveillance. By strategically optimizing the number of cores and containers, a balanced, efficient, and robust surveillance system capable of meeting diverse operational demands can be achieved.

The results show that the best performance was achieved with five cores in one container at the Edge layer and two containers in the Fog layer. This configuration led to the lowest MRT, lowest drop rate, and highest utilization of the Edge and Fog layers. Increasing the number of containers in the Fog is advantageous due to its substantial computing resources, while at the Edge, increasing the number of cores is more cost-effective because simpler processing provides shorter service times.

6.4 Analysis of Routing Strategies Variation

This subsection analyzes the impact of different routing strategies on the Edge-Fog Gateway. The Edge-Fog Gateway is responsible for transmitting images and other data from the Edge layer to the Fog layer. The study involves varying the arrival rates to assess their influence on the MRT metric. The routing strategies employed in this analysis are supplied by the JMT simulation tool [Ghandour *et al.*, 2024]. A summary of the strategies used is provided below:

- *Random*: Jobs are assigned to one of the stations linked to the output of the specified station at random. Each outgoing link is chosen with equal likelihood. In a system where there are three outgoing links, the probability for each link is 1/3.

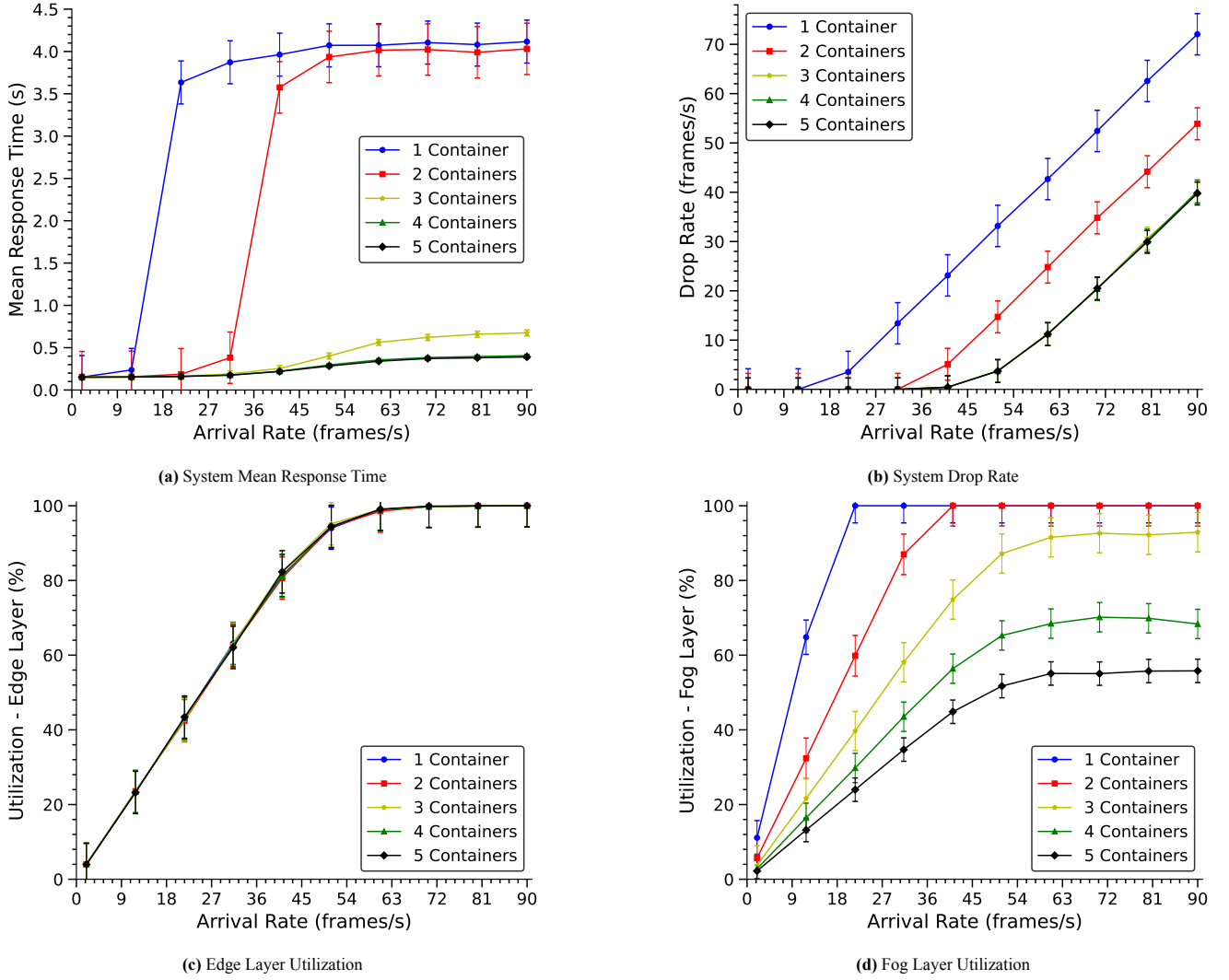


Figure 4. Experimental Results of Scenario I - Varying Number of Containers in the Fog.

- *Round Robin*: Jobs are distributed cyclically to the outgoing links following a circular routing pattern.
- *Least Utilization*: The destination station is selected based on having the least utilization at the time of routing.
- *Join the Shortest Queue (JSQ)*: Each job is directed to the station linked at the output that has the lowest total number of jobs in both the queue and being serviced at the time the job exits the routing station.

Table 5 presents the configuration used and the evaluated routing strategies. The adopted configuration includes one container at the Edge and three containers in the Fog, with five cores in each container. This configuration was chosen because, in Scenario two, the best results across all metrics were achieved with five cores in the Edge and Fog containers. In Scenario one, the Fog with three containers produced results similar to those with four and five containers. Given that increasing the number of containers incurs additional costs, the three container configuration was determined to be the most efficient, offering lower costs compared to the four and five container configurations.

Figure 6 presents the results obtained through simulations. The metric selected to evaluate the routing strategies was

MRT, as MRT is the metric that defines how long the system takes to respond to the user, i.e., how long the system takes to process the images and, if necessary, notify the security personnel. The frame rate per second that the system receives from all the building's cameras varies from zero to 180 frames/s. Unlike in scenarios one and two, the arrival rate was doubled. This was done to demonstrate the flexibility of the queuing models when conducting an analysis. Furthermore, with only 90 frames per second, the behavior of the system's MRT with each routing strategy is barely noticeable.

Up to approximately 54 frames/s, all four strategies yield similar results, with a response time of around 0.15 seconds or 150 milliseconds. After 72 frames/s, the random routing strategy begins to diverge from the other routing strategies. After 108 frames/s, the random strategy proves to have the highest MRT in the system—the round-robin strategy results in the second-highest MRT. However, the least utilization and join the shortest queue strategies have similar MRTs. After 162 frames/s, the least utilization routing strategy becomes slower than the join the shortest queue strategy. This allows us to rank the routing strategies for the configuration with one container at the Edge, three containers in the Fog, and five cores in each container.

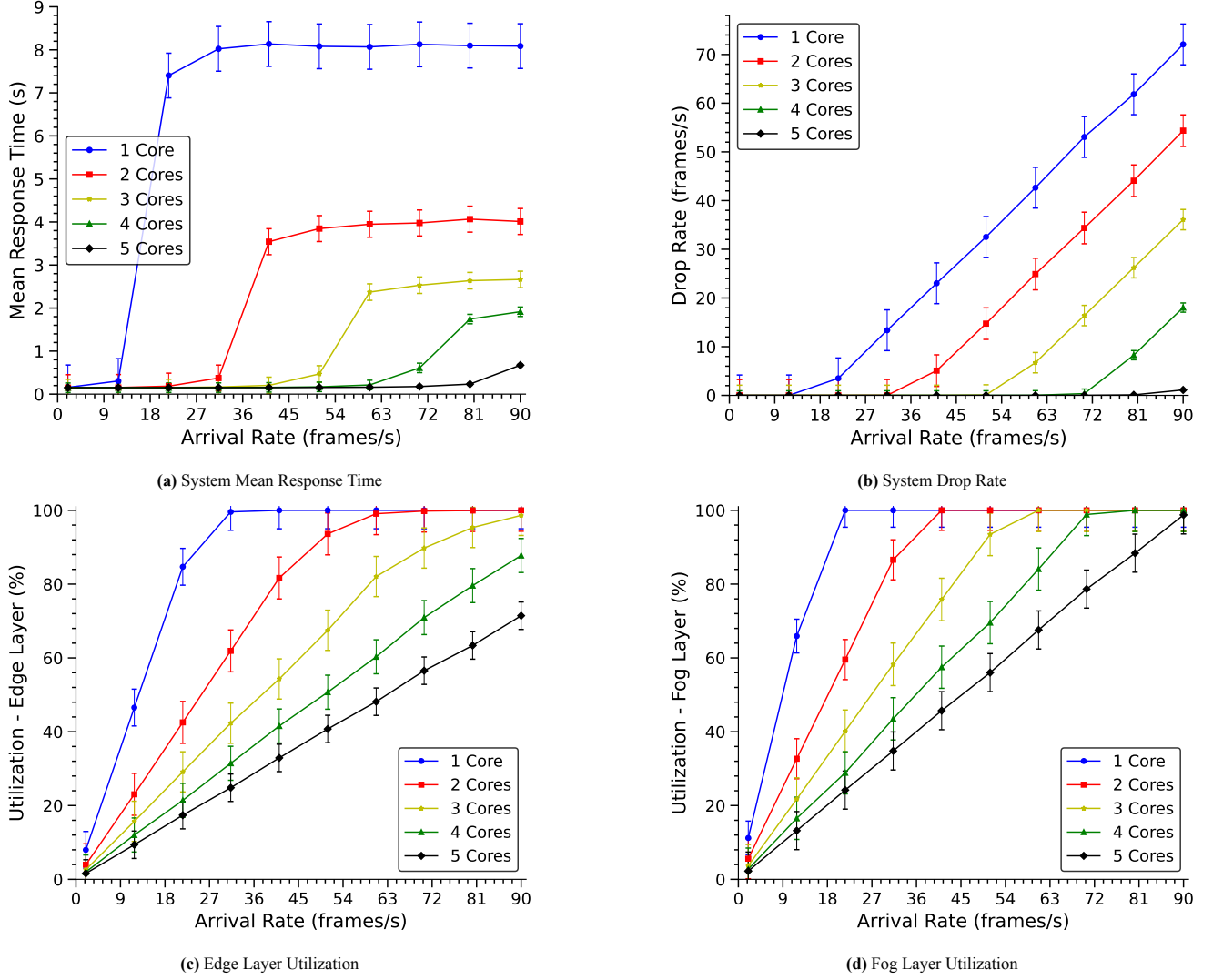


Figure 5. Experimental Results of Scenario II - Varying the Number of Cores in the Edge and Fog.

Table 5. The evaluated routing strategies and configuration used.

Routing Strategy	Layer	Number of Containers	Number of Cores
Random, Least Utilization, Round Robin, Join the Shortest Queue	Edge	1	5
	Fog	3	5

The join the shortest queue routing strategy proved to be the most effective for this configuration, resulting in the lowest MRT values, outperforming the least utilization, round robin, and random routing strategies. On the other hand, the random strategy leads to higher MRT times. The graph suggests that as the number of frames per second increases, the impact of each routing strategy on the MRT becomes more pronounced. Analyzing each routing strategy is crucial, especially for systems with high workloads.

6.5 Discussion on Edge-to-Fog Latency

The proposed model abstracts communication latency between the Edge and Fog layers to reduce analytical complexity and emphasize processing behavior. This simplification is justifiable in scenarios where both layers operate within

the same local network infrastructure, which is common in smart building applications.

However, in larger-scale or geographically distributed deployments, the latency between these layers may become significant—especially under heavy load or limited network bandwidth. In such cases, latency could contribute noticeably to the overall Mean Response Time (MRT), potentially shifting the performance balance between configurations.

Although the current findings remain valid for the intended use case, future work will consider integrating edge-to-fog latency as a configurable parameter, either fixed or modeled stochastically. This enhancement would broaden the applicability of the model and allow for a more accurate simulation of systems deployed across heterogeneous or less predictable networks.

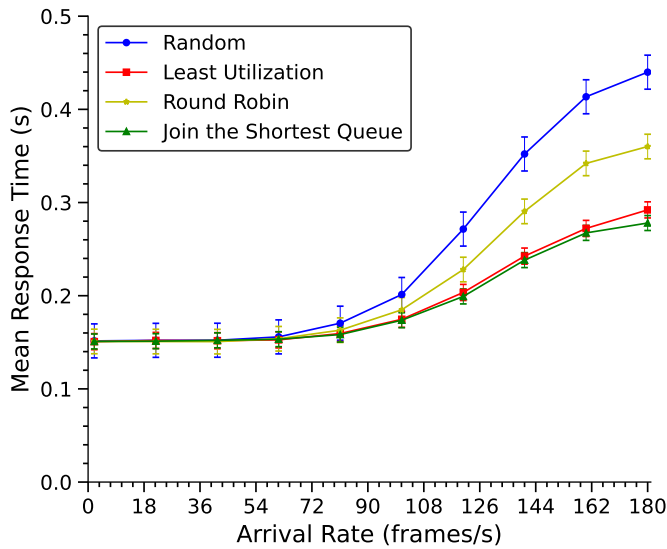


Figure 6. Comparison of the Four Routing Strategies Under Study.

7 Conclusion

This paper presents a queuing network model to evaluate the performance of a surveillance system in a smart building, which uses cameras, Edge, and Fog layers as part of its infrastructure. The model is capable of calculating the system's average response time, system drop rate, utilization of the Edge and Fog layers, and investigating the variation in the number of containers in the Fog layer and the number of cores in both layers. This approach can assist system analysts in defining the optimal configuration for a surveillance infrastructure in buildings using cameras. The best overall performance for scenario one was observed with one container at the Edge and three containers in the Fog. For scenario two, the best performance occurred with a combination of one container at the Edge and two containers in the Fog, each container having five cores. Among the routing policies evaluated, Join the Shortest Queue exhibited the lowest Mean Response Time (MRT) values, while Random showed the highest.

Additionally, the impact of routing policies increases with the arrival rate, highlighting greater performance differences as the workload grows. Although the data transmission time between the Edge and Fog was not considered in this study, the model can be easily adapted to include it. This work extends existing studies by providing a detailed performance evaluation of containerized Fog computing. It analyzes how varying the number of containers impacts response time, utilization, and drop rate, while also highlighting the growing influence of routing policies under higher workloads. The queuing model used offers valuable insights for optimizing resource allocation and system performance. Future work is expected to observe the system's throughput behavior and vary multiple components simultaneously, such as increasing both the number of cores and containers.

Declarations

Funding

No funding was received to conduct this study.

Authors' Contributions

All authors contributed equally to the work.

Competing interests

The authors have no relevant financial or non-financial interests to disclose.

Availability of data and materials

Data sharing is not applicable.

References

- Al-Naday, M., Reed, M., Dobre, V., Toor, S., Volckaert, B., and De Turck, F. (2023). Service-based federated deep reinforcement learning for anomaly detection in fog ecosystems. In *2023 26th Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, pages 121–128. IEEE. DOI: 10.1109/icin56760.2023.10073495.
- Alsadie, D. (2024). Artificial intelligence techniques for securing fog computing environments: trends, challenges, and future directions. *IEEE Access*. DOI: 10.1109/access.2024.3463791.
- Amshavalli, R. and Kalaivani, J. (2023). Real-time institution video data analysis using fog computing and adaptive background subtraction. *Journal of Real-Time Image Processing*, 20(5):96. DOI: 10.1007/s11554-023-01350-3.
- 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. DOI: 10.23919/cisti58278.2023.10211770.
- 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. DOI: 10.1109/tits.2020.3008420.
- Ding, S. (2023). Analysis and application of computer queueing theory. In *2023 International Conference on Image, Algorithms and Artificial Intelligence (ICIAAI 2023)*, pages 447–453. Atlantis Press. DOI: 10.2991/978-94-6463-300-9_45.
- Dudin, S. and Dudina, O. (2023). Analysis of a multi-server queue with group service and service time dependent on the size of a group as a model of a delivery system. *Mathematics*, 11(22):4587. DOI: 10.3390/math11224587.
- Ghandour, O., El Kafhali, S., and Hanini, M. (2023). Computing resources scalability performance analysis in cloud computing data center. *Journal of Grid Computing*, 21(4):61. DOI: 10.1007/s10723-023-09696-5.
- Ghandour, O., El Kafhali, S., and Hanini, M. (2024). Adaptive workload management in cloud computing for service level agreements compliance and resource optimization. *Computers and Electrical Engineering*, 120:109712. DOI: 10.1016/j.compeleceng.2024.109712.

- Gracias, J. S., Parnell, G. S., Specking, E., Pohl, E. A., and Buchanan, R. (2023). Smart cities—a structured literature review. *Smart Cities*, 6(4):1719–1743. DOI: 10.3390/smartsities6040080.
- Hasan, R. A., Obaid, O. I., AlQassab, A. I. M., Jasim, H., and Dheyab, S. A. (2024). Mapping web service characteristics to queueing theory models for performance analysis. *Babylonian Journal of Networking*, 2024:98–110. DOI: 10.58496/bjn/2024/011.
- Hossain, M. E., Tarafder, M. T. R., Ahmed, N., Al Noman, A., Sarkar, M. I., and Hossain, Z. (2023). Integrating ai with edge computing and cloud services for real-time data processing and decision making. *International Journal of Multidisciplinary Sciences and Arts*, 2(4):252–261. DOI: 10.47709/ijmdsa.v2i1.2559.
- 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. DOI: 10.1088/1757-899x/1176/1/012030.
- 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. DOI: 10.1109/icce59016.2024.10444244.
- 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. DOI: 10.3390/app131910682.
- Kumar, T., Sharma, P., Tanwar, J., Alsg hier, H., Bhushan, S., Alhumyani, H., Sharma, V., and Alutaibi, A. I. (2024). Cloud-based video streaming services: Trends, challenges, and opportunities. *CAAI Transactions on Intelligence Technology*, 9(2):265–285. DOI: 10.1049/cit2.12299.
- Moorthy H., R., Upadhy, V., Holla, V. V., Shetty, S. S., and Tantry, V. V. (2020). Challenges encountered in building a fast and efficient surveillance system: An overview. In *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, pages 731–737. DOI: 10.1109/I-SMAC49090.2020.9243563.
- Myagmar-Ochir, Y. and Kim, W. (2023). A survey of video surveillance systems in smart city. *Electronics*, 12(17):3567. DOI: 10.3390/electronics12173567.
- Narmadha, V. and Rajendran, P. (2024). A literature review on development of queueing networks. *Reliability: Theory & Applications*, 19(1 (77)):696–702. Available at: https://gnedenko.net/Journal/2024/012024/RTA_1_2024-54.pdf.
- Ogumeyo, S. and Omole, C. (2023). Mathematical analysis of performance measures of m/m/1 queue models. *The Transactions of the Nigerian Association of Mathematical Physics*, 19:125–134. DOI: 10.60787/tnamp-19-125-134.
- 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. DOI: 10.1016/j.procs.2023.01.220.
- 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. DOI: 10.21203/rs.3.rs-3635851/v1.
- Santos, B., Soares, A., Nguyen, T.-A., Min, D.-K., Lee, J.-W., and Silva, F.-A. (2021). Iot sensor networks in smart buildings: A performance assessment using queueing models. *Sensors*, 21(16):5660. DOI: 10.3390/s21165660.
- 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. DOI: 10.1109/wsc52266.2021.9715373.
- 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. DOI: 10.1109/in-cacct57535.2023.10141802.
- Singh, P. et al. (2025). Queueing theory in cloud computing: Analyzing m/m/1 and m/m/c/n models with aws. *International Journal of Science and Social Science Research*, 2(4):122–134. DOI: 10.5281/zenodo.14942226.
- Srirama, S. N. (2024). A decade of research in fog computing: relevance, challenges, and future directions. *Software: Practice and Experience*, 54(1):3–23. DOI: 10.1002/spe.3243.
- 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. DOI: 10.1051/e3sconf/202344306016.
- 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. DOI: 10.1109/lwc.2021.3076415.