Worst-Case Attacks in Reactive Edge-Cloud Systems: Expected Latency and SLA Violation

Jhonatan Tavori, Mehmet Kerem Turkcan, Zoran Kostic, Javad Ghaderi, Gil Zussman
Columbia University, New York, USA
{j.tavori,mkt2126,zk2172,jg3465,gil.zussman}@columbia.edu

***Abstract*—Edge-cloud systems are increasingly deployed to meet the demands of real-time applications by processing latency-sensitive tasks at the edge site, near the end-devices. When load exceeds capacity, offloading of requests to the cloud allows the system to balance the spike. As their popularity increases, edge servers become an attractive target for attackers whose goal is to degrade system performance.**

This paper investigates *worst-case* (i.e., damage-maximizing) attacks on reactive edge-cloud systems, focusing on two key performance metrics: expected end-to-end latency and the probability of violating service-level agreements (SLAs). We use a general modeling framework based on two layers of computation and analyze how adversarial removal of edge servers affects delay under reactive offloading mitigation.

We adopt common queueing models and prove that the system's performance remain *convex* in the attack size, even when accounting for reactive offloading by the edge. This enables a characterization of worst-case policies as *concentrated* attacks, targeting only a few critical sites to maximize damage, yielding greater performance degradation than spreading efforts across the network. Simulation results validate the analysis and demonstrate that the identified worst-case attack strategies can increase damage by up to 40% using the same attack resources.

***Index Terms*—Edge-Cloud Computing, Worst-Case Attacks, Convexity Analysis, Network Security.**

I. Introduction

The use of edge-cloud architectures is rapidly expanding, driven by the increasing demand for low-latency processing in real-time applications such as smart cities and autonomous systems [1]–[3]. In such systems, latency-sensitive tasks are executed on geographically distributed edge servers which are located close to end devices.

Service providers must ensure compliance with strict service-level agreements (SLAs), especially regarding availability and latency deadlines [4], [5]. When edge resources are insufficient or overloaded for some period, tasks may be offloaded to a centralized cloud. While this introduces increased communication latency, offloading can help when edge queues are highly congested by avoiding excessive queueing delays.

The integration of edge and cloud resources has been extensively studied in the literature. Prior work shows that optimal server placement, dynamic task offloading, and coordinated cross-tier scheduling can significantly improve real-time performance, e.g., [6]–[9].

However, this architectural shift from classical cloud models introduces new vulnerabilities. Edge sites are typically smaller, more densely deployed, and less physically secure than cloud data centers. Their reliance on public infrastructure and wireless access exposes them to a wider range of attacks.

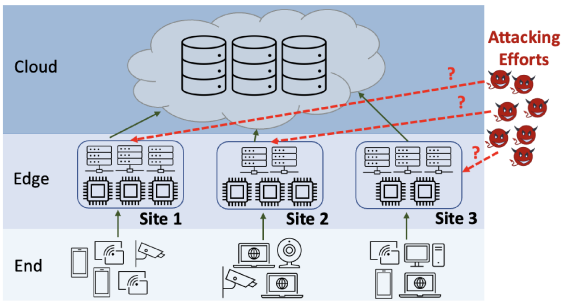


Fig. 1: Worst-case attack in an edge–cloud architecture: Should an attacker spread or concentrate its efforts to maximize overall system disruption?

An attacker may degrade system performance by selectively disabling edge servers via malware injection, targeted hardware disruption, or distributed denial-of-service (DDoS) attacks. This results in longer queues at edge sites and excessive offloading to the cloud, ultimately inflating latency and causing tasks to miss their deadlines.

In this work, we investigate how an intelligent adversary can disrupt edge-cloud systems by selectively targeting edge servers. The edge-cloud interplay raises new structural questions: How should an attacker allocate limited resources to inflict maximal damage? Do effective attacks concentrate on a small number of loaded edge sites, or scatter efforts to trigger global cascades? How does the availability of a cloud fallback influence the worst-case attack pattern?

Recently, [10] studied worst-case attacks on single-layer cloud systems, focusing on request blocking and expected latency in the M/M/1 model. This paper goes beyond that framework by analyzing worst-case attacks in hybrid edge-cloud architectures, comprising two integrated computation layers with reactive offloading, hence capturing the interplay between local edge failures, cloud offloading, and end-to-end latency. We incorporate additional service models and analyze both the expected delay and the SLA violation probability.

While our primary focus is on edge-cloud systems, the two-layer computation-communication model we analyze captures a broader class of systems where local processing hardware is backed by a high-capacity remote tier. Similar architectures emerge in mobile networks [11], [12], vehicular networks [13],

and satellite-ground platforms [14], [15], where performance is jointly dependent on local capacity and remote latency. For clarity of exposition, our analysis in this paper will focus on edge-cloud systems, where local computation and remote communication jointly determine performance under attack.

We study an edge-cloud system with k edge sites, represented by $\boldsymbol{C} = (C_1, \ldots, C_k)$, $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_k)$, $\boldsymbol{x} = (x_1, \ldots, x_k)$. Site i hosts C_i edge servers and its intended tasks arrive as a Poisson process at rate λ_i. A fraction x_i of tasks is processed locally at the edge site i, while the remaining fraction $1 - x_i$ is offloaded to the cloud, incurring additional communication delay. Figure 1 depicts an example of the system with three edge sites ($k = 3$). The adversarial attack is represented by $\boldsymbol{v} = (v_1, \ldots, v_k)$, where v_i servers are disabled at site i. The complete model is given in Section III.

When evaluating the damage inflicted by an attack, we focus on two key performance metrics that capture delay-related service degradation:

- *Expected response time, D:* This quantifies the average latency experienced by tasks after queueing and offloading, defined as:

$$D\big(\boldsymbol{\lambda}, \boldsymbol{x}, \boldsymbol{C} - \boldsymbol{v}\big) = \mathbb{E}[\text{latency of an arbitrary task}]\,. \quad (1)$$

- *SLA violation probability, P:* This is the probability that a task's end-to-end delay exceeds a real-time deadline t, defined as:

$$P^t\big(\boldsymbol{\lambda}, \boldsymbol{x}, \boldsymbol{C} - \boldsymbol{v}\big) \;=\; \Pr\{\text{latency} > t\}\,. \quad (2)$$

We consider an adversary with a total attack budget V, who selects an attack vector $\mathbf{v}$ such that $\|v\|_1 \leq V$, aiming to maximize service degradation as defined by the above metrics. That is, the attacker chooses how many servers to disable at each edge site. In Fig. 1, this corresponds to deciding how many attack efforts to allocate to each site.

To reflect realistic system behavior, we allow for online mitigation by the operator through adaptive offloading decisions $\mathbf{x}$, determining the fraction of tasks to process locally versus offload to the cloud. The goal is to minimize worst-case performance degradation via robust provisioning and offloading strategies.

This naturally leads to a max–min optimization problem, in which the attacker seeks to maximize system damage by selecting an attack vector $\mathbf{v}$, anticipating that the system will optimally adjust the offloading strategy $\mathbf{x}$ in response. At a high level, this can be expressed as

$$\max_{\mathbf{v}:\|v\|_1 \leq V} \; \min_{\mathbf{x}:x\in[0,1]^k} \; D\big(\boldsymbol{\lambda}, \boldsymbol{x}, \boldsymbol{C}-\boldsymbol{v}\big) \text{ and } P^t\big(\boldsymbol{\lambda}, \boldsymbol{x}, \boldsymbol{C}-\boldsymbol{v}\big). \quad (3)$$

A formal definition is provided in Section IV. This attacker–defender interplay lies at the core of our analysis.

Our Contributions

We introduce a convex-analytic framework for evaluating worst-case attacks in edge-cloud systems. A central insight is that when the performance degradation function is convex in the attack vector, optimal attack strategies become concentrated, they inflict the most severe degradation by focusing on a small number of critical sites rather than spreading effort system-wide. In Section IV, we formalize this principle and show, via marginal damage analysis (i.e., the effect of increasing v_i to $v_i + 1$), that convexity in this setting naturally induces concentrated attack behavior.

We then examine per-site behavior to understand how localized degradation propagates system-wide. We begin with the widely used M/M/1 queueing model [16] (Section V). We show that the performance degradation at an individual edge site is convex in the number of disabled servers. We examine both performance metrics: expected latency and SLA violation probability. We analyze their dependence on both the attack size v and the system's offloading decision x, and prove that both metrics are *jointly convex* in x and v.

Within our max–min formulation, where the system reacts to attacks by adjusting the offloading decisions $\mathbf{x}$ to minimize degradation, we show that this joint convexity implies that the resulting objective, latency under optimal offloading, remains convex in the attack vector $\mathbf{v}$. In turn, this ensures that even under optimal reaction by the system, the attacker's best strategy remains concentrated. A similar result is given for the SLA violation metric P^t.

Next, we extend the analysis to the M/D/1 queueing model [17] (Section VI), which is relevant for many edge workloads with low service time variability, such as AI inference tasks. We show that the expected latency remains convex with respect to both the attack size and the offloading fraction. Because the exact formulas for SLA violation are not always analytically tractable [18], [19], we employ approximations and validate them via numerical analysis, through which convexity is empirically demonstrated.

We generalize our results to the M/G/1 setting [20], where service times follow an arbitrary distribution with finite variance (Section VII). Despite the increased generality, we establish convexity of the expected latency in both attack and offloading variables, under any service time distribution. To support the analysis, we evaluate several service-time distributions commonly used in the computer-networks literature, e.g., hyper-exponential, uniform, log-normal, and Pareto, spanning low-variance, high-variance, and long-tailed behaviors. Using simulations we demonstrate convexity in all cases.

We further evaluate multi-site systems in Section VIII, demonstrating that worst-case attacks are consistently concentrated and thus confirming our theoretical results. As observed, concentrating the attack budget on a single site causes increased damage than distributing it across multiple sites, across varying cloud delay settings and service-time distributions.

This work establishes broadly applicable structural results for edge–cloud systems across a wide range of service-time distributions. The convexity of key performance metrics under adversarial capacity loss provides a principled foundation for resilient planning to ensure SLA compliance under attack. Moreover, the concentration property reduces the attacker's effective search space, from exponential in the number of sites to

approximately linear, potentially enabling faster identification of worst-case strategies, even with limited system knowledge.

II. Related Works

Security challenges in edge-cloud systems are receiving increased attention, as edge nodes, unlike centralized clouds, are more vulnerable to physical compromise, malware injections and targeted DDoS attacks [21]–[23].

Models for edge-cloud systems have been extensively studied, particularly to optimize offloading, resource coordination, and infrastructure design [6], [7], [9], [24]. Several models address the optimal placement of edge servers to reduce access latency and balance network load (see, e.g., [6]). Others explore resource coordination between edge and cloud tiers, including wholesale and buyback models, where edge servers offer excess capacity to the cloud for shared processing [7], [8]. Additional formulations consider collaborative workload allocation to minimize end-to-end delay, accounting for bandwidth and service heterogeneity across tiers [9].

While DDoS attacks and mitigation strategies have been extensively studied, *worst-case attacks* have been analyzed only in centralized cloud settings [10], [25], with limited attention to edge-cloud architectures. To our knowledge, this is the first work to propose a max-min framework in the edge-cloud context that jointly considers worst-case attack strategies under system-level mitigation.

Queueing theory provides closed-form expressions for waiting times under various arrival and service time assumptions. Among these, the Poisson process is one of the most widely used stochastic models for task arrival generation in communication networks, as it captures random, memory-less event occurrences while still permitting tractable analysis of key performance metrics [20], [26].

The M/M/1 queueing model, which assumes Poisson arrivals and exponentially distributed service times, has been extensively applied to analyze delay and utilization in edge-cloud systems [16], [27], [28]. It has also been used to model task behavior in mobile devices and wireless endpoints connected to base stations [29], [30], enabling analytical comparisons of latency under different resource allocation strategies.

In scenarios where service times are fixed or deterministic, the M/D/1 queueing model, which assumes Poisson arrivals and constant service times, provides a better fit. It has been applied to edge-cloud models involving deep neural network (DNN) inference tasks [17], and to real-time video analytics workloads such as camera frame processing [31], where processing times exhibit low variance.

For scenarios with variable and bursty processing times, the M/G/1 model provides a flexible framework and has been used to study edge–cloud systems under general service distributions, including heavy-tailed or high-variance workloads [24].

III. System Model and Notation

We consider a distributed edge–cloud computing system with k geographically separated edge sites and a centralized cloud. Each site hosts multiple edge servers and serves a local population of mobile or IoT devices (e.g., surveillance cameras, sensors, or embedded devices) that generate computation tasks. Our framework captures edge and cloud resources balancing dynamics and allow us to study offloading strategies and their interaction with task arrival rates and edge-server availability under adversarial conditions.

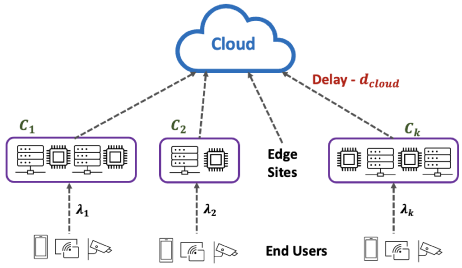


Fig. 2: Model notation for a multi-site edge–cloud system, where tasks are processed locally or offloaded to the cloud with additional communication latency.

At site $i \in \{1, 2, \ldots, k\}$, computation tasks are generated by connected local devices (e.g., with periodic sensing or stochastic activity) and are approximated as arriving according to a Poisson process with rate λ_i. This rate reflects the number of local devices and their configurations, such as sensing frequency or video frame rate. To process these tasks, each site hosts $C_i \geq 0$ identical edge servers. Every edge server has a service rate of μ (i.e., it completes on average μ tasks per unit time, with mean service time $1/\mu$), yielding a total local service capacity of $C_i \cdot \mu$ at site i. The actual service times may follow various distributions (e.g., exponential, deterministic, or more general forms), and in later sections we analyze performance under different service-time models. The notation is illustrated in Fig. 2, which depicts the users, edge servers, and the cloud. In the multi-site system comprising k sites, we denote the vectors of arrival rates and server capacities as $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_k)$ and $\mathbf{C} = (C_1, \ldots, C_k)$, respectively.

Due to resource constraints or traffic surges, not all tasks may be processed locally. Therefore, a fraction of tasks may be offloaded to the centralized cloud infrastructure. We denote by $x_i \in [0, 1]$ the fraction of tasks at site i that are processed locally at the edge; the remaining fraction $1 - x_i$ is outsourced to the remote cloud. For the full system, we define the offloading vector $\boldsymbol{x} = (x_1, \ldots, x_k)$.

As illustrated in Fig. 3, tasks processed locally are distributed across the C_i edge servers at site i, yielding an effective arrival rate of $x_i \lambda_i / C_i$ per server.

While and edge site can offload some or all of its traffic to the cloud, doing so inherently incurs additional communication delay, introducing a fundamental tradeoff. We model the communication delay as a non-negative random variable C_{cloud} and the cloud processing time as T_{cloud}, and let $D_{\text{cloud}} = C_{\text{cloud}} + T_{\text{cloud}}$ denote the total cloud latency, with expectation $d_{\text{cloud}} = \mathbb{E}[D_{\text{cloud}}]$. We also define the tail violation probability $P^t_{\text{cloud}} = \Pr(D_{\text{cloud}} > t)$, i.e., the probability that the cloud latency exceeds t. As cloud services are typically provisioned

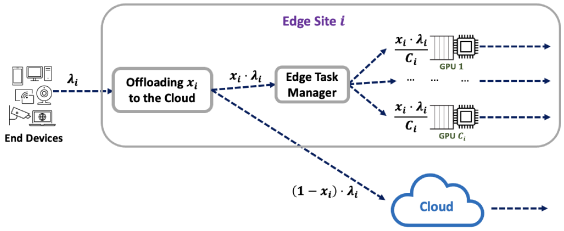


Fig. 3: Zoom-in on a single edge site. Site has C_i local servers to process tasks. A fraction x_i of tasks is served locally, while the remaining fraction $1 - x_i$ is offloaded to the cloud.

with substantially greater computational capacity than edge systems, we assume that D_{cloud} and P^t_{cloud} are independent of the offloaded traffic intensity. While this assumption limits the model's applicability in some scenarios, it enables a focused analysis of offloading decisions and their impact on edge performance, which is the primary focus of this paper.

A. Performance Metrics

To quantify real-time performance in compromised systems, we consider (i) the expected end-to-end latency and (ii) the SLA violation probability, i.e., the probability that response times exceed their deadlines.

1) Expected end-to-end latency: Let $D_{\text{edge}}(\lambda_i, C_i)$ denote the expected sojourn time at edge site i. The expected end-to-end latency of tasks generated at site i, denoted by D_i, is:

$$D_i(x_i) = x_i\, D_{\text{edge}}(x_i\lambda_i, C_i) + \big(1 - x_i\big) d_{\text{cloud}}. \tag{4}$$

This quantity depends on the offloading parameter x_i.[1]

The optimal offloading ratio $x_i^\star$ is obtained as

$$x_i^\star = \arg\min_{0 \le x_i \le 1} \; D_i(x_i) \quad \text{s.t.} \quad x_i \cdot \lambda_i < C_i \cdot \mu. \tag{5}$$

2) SLA (deadline) violation probability: Given a latency deadline t, let

$$P^t_{\text{edge}}(x_i, \lambda_i, C_i) \;:=\; \Pr\big\{\text{edge delay} > t\big\}.$$

The overall latency violation probability is

$$P^t(x_i) = x_i \cdot\; P^t_{\text{edge}}\big(x_i\big) + \big(1 - x_i\big) \cdot P^t_{\text{cloud}}. \tag{6}$$

and the optimal offload $x_i^\star$ is defined as in (5).

From the system operator's point of view, the goal is to optimize performance across all sites. A natural objective is the weighted average of per-site performance, with each site weighted by its arrival rate to reflect user demand. Let $\Lambda = \sum_{i=1}^k \lambda_i$. Then the expected delay of an arbitrary request is

$$D = \sum_{i=1}^{k} \frac{\lambda_i}{\Lambda}\, D_i(x_i^\star), \tag{7}$$

and the corresponding tail (SLA violation) probability at threshold t is

$$P^t = \sum_{i=1}^{k} \frac{\lambda_i}{\Lambda}\, P_i^t(x_i^\star), \tag{8}$$

[1]When the system parameters (λ_i, C_i) are clear from the context, we omit them from the explicit notation D_i.

where D_i and P_i^t are the per-site expected delay and violation probability under the optimal offloading decision $x_i^\star$. Because these objective functions are separable across sites, we focus our analysis on how the attack size at each site impacts its local contribution to the overall degradation.

IV. Worst-Case Attacks

With the service model established, we turn to analyzing how an intelligent adversary can degrade service quality by selectively disabling edge servers.

A. Adaptive Offload under Attack: A Max-Min Game

We denote the attack vector $\mathbf{v} = (v_1, \ldots, v_k)$, where the attacker removes v_i servers from site i, reducing its local processing capacity from C_i to $C_i - v_i$ (i.e., $v_i \le C_i$). The adversary's goal is to inflict maximum degradation, either in terms of expected end-to-end delay (4) or the deadline-violation probability (6). Such attacks are constrained in practice by physical, computational, or logistical limitations, making a budgeted adversary model both realistic and analytically meaningful. Accordingly, the attack is subject to a global budget V:

$$\|\mathbf{v}\|_1 = \sum_{i=1}^{k} v_i \le V. \tag{9}$$

To capture the system's reactive mitigation, we incorporate *reactive* offloading: following an attack, the system can shift a fraction of load to the cloud to mitigate local edge overload. Thus, determining the worst-case degradation requires jointly analyzing the attack vector $\boldsymbol{v}$ and the system's offloading decisions $\boldsymbol{x}$. We denote by $D_i(x_i^\star, v_i)$ the latency under the optimal offload, given an attack of size v_i in site i.

Note that we require system *stability*, and thus only consider offloading values within the following set[2]:

$$\mathcal{F}_i := \{(x, v) \mid x \in [0,1], v \le C_i, \mu \cdot (C_i - v) \ge \lambda_i \cdot x\}\,. \tag{10}$$

We model the attacker-defender interaction and interplay as a max–min game:

$$\max_{\{\mathbf{v}\ \text{s.t.}\ \|\mathbf{v}\|_1 \le V\}} \quad \min_{\{\mathbf{x}\ \text{s.t.}\ \forall i,\ (x_i, v_i) \in \mathcal{F}_i\}} \mathbf{D}_{\mathbf{C},\boldsymbol{\lambda},\mu}(\mathbf{v}, \mathbf{x}). \tag{11}$$

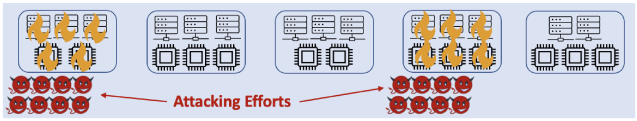
The attacker chooses $\boldsymbol{v}$ to maximize degradation, while the system responds by optimizing $\boldsymbol{x}$ to minimize it. A similar formulation applies when the objective is to maximize the SLA violation probability, replacing with $\max\ \min\ P^t(\mathbf{v}, \mathbf{x})$.

B. Concentrated Structure of Worst-Case Convex Attacks

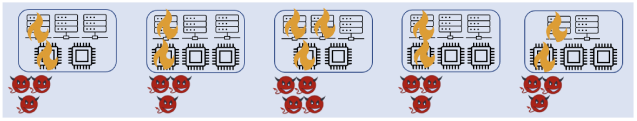
In the following sections, we show the performance metrics under consideration (expected delay and violation probability) exhibit a *convex structure* with respect to the attack vector. Intuitively, convexity implies that marginal damage is non-decreasing in the number of servers disabled in the site.

This convexity enables us to characterize the structure of optimal attacks. In particular, we show that worst-case attacks

[2]Such a constraint is always satisfiable, since $x_i = 0$ trivially belongs to the feasible set. Throughout the paper, we refer to $\mathcal{F}_i$ as the *feasible set*.



(a) Concentrated attack: Attack efforts focused on a small number of sites to reach their critical capacity and maximize service degradation.



(b) Scattered attack: Attack efforts spread across many sites, aiming for a system-wide disruption.

Fig. 4: Attack pattern visualization. As we show, concentrated attacks (top) represent the worst-case strategy and cause greater degradation than scattered attacks (bottom).

are *concentrated*, meaning that the attack budget is allocated to as few sites as possible. This behavior is illustrated in Fig. 4a, where attack resources are concentrated at sites 1 and 4. Formally:

Definition IV.1 (Concentrated Attack)**.** *An attack vector $\boldsymbol{v}$ is* concentrated *if for every site i, $v_i = 0$ or $v_i = C_i$, except possibly one site which may be partially attacked to absorb leftover attack budget.*[3]

This structure simplifies the search for damage-maximizing attacks. Similar convexity properties were observed in cloud-only models [10]. We provide a formal proof that edge-cloud worst-case attacks are concentrated, generalizing prior results and relying on the convexity of the damage function with respect to the attack vector.

Proposition IV.1 (Worst-Case Convex Attacks are Concentrated)**.** *Suppose the attacker wishes to maximize a performance degradation metric $D = \sum_{i=1}^{k} \lambda_i/\Lambda \cdot D_i(x_i^\star, v_i)$*[4]*. If for every site i, $D_i(x_i^\star, v_i)$ is convex in the attack size v_i, and separable across sites, then the attacker maximizes damage by concentrating the budget on a few critical sites rather than spreading it across many.*

Proof of Proposition IV.1. The attacker's goal is to maximize:

$$\max_{\mathbf{v}=(v_1,\ldots,v_k)} \quad \sum_{i=1}^{k} \frac{\lambda_i}{\Lambda} D_i(x_i^\star, v_i) \quad \text{s.t.} \quad \|\mathbf{v}\|_1 = \sum_i v_i \le V,$$

where $x_i^\star$ is the optimal offloading strategy. Recall that $D_i(x_i^\star, v_i)$ is convex in v_i.

Let $\mathbf{v}^\star$ be the worst-case attack (i.e., $\mathbf{v}^\star$ maximizes the delay). Assume by contradiction that the optimal attack $\mathbf{v}^\star$ includes two partially attacked sites i_1, i_2, i.e., $0 < v_{i_1}^\star < C_{i_1}$, $0 < v_{i_2}^\star < C_{i_2}$. Suppose without loss of generality that[5]:

$$\left.\frac{\partial D_{i_1}}{\partial v}\right|_{v_{i_1}^\star} > \left.\frac{\partial D_{i_2}}{\partial v}\right|_{v_{i_2}^\star}. \tag{12}$$

Now consider shifting a small amount of attack budget from site i_2 to site i_1, resulting in $v'_{i_1} = v_{i_1}^\star + 1$, and $v'_{i_2} = v_{i_2}^\star - 1$. Denote this change by $\mathbf{v}'$. By convexity of $D_i(x_i^\star, v_i)$, and by our assumption (12):

$$D_{i_1}(x_{i_1}^\star, v'_{i_1}) - D_{i_1}(x_{i_1}^\star, v_{i_1}^\star) > D_{i_2}(x_{i_2}^\star, v_{i_2}^\star) - D_{i_2}(x_{i_2}^\star, v'_{i_2}),$$

so the total cost increases after the shift: $\sum_{i=1}^{k} \frac{\lambda_i}{\Lambda} D_i(x_i^\star, v'_i) > \sum_{i=1}^{k} \frac{\lambda_i}{\Lambda} D_i(x_i^\star, v_i^\star)$, which contradicts the optimality of $\mathbf{v}^\star$. □

Hence, the optimal attack does not involve multiple partially attacked sites, aligning with the structure we defined as a concentrated attack. Therefore, by establishing that the performance metrics are convex in the attack vector, we conclude that the worst-case attack must be concentrated.

V. The M/M/1 Case: Analysis

To evaluate the impact of attacks on the edge-cloud latency, we begin with the M/M/1 queueing model, widely used in edge-cloud systems due to its analytical tractability. In this model, tasks arrive according to a Poisson process (rate λ) and are served with exponentially distributed service times (rate μ). The expected latency (sojourn time) and the tail probability of exceeding a deadline t are given by:

$$D_{\text{M/M/1}}(\lambda, \mu) = \frac{1}{\mu - \lambda}, \quad P^t_{\text{M/M/1}}(\lambda, \mu) = e^{-(\mu-\lambda)\cdot t}. \tag{13}$$

We focus on a single edge site in the system, and analyze the behavior of D and P^t under the M/M/1 model.

A. Expected Latency Convexity

We establish that the expected latency function is convex in both the attack size v_i and the offloading fraction x_i under the M/M/1 model.

Claim V.1. *Let $D_i(x_i, v_i)$ denote the expected latency of a request when a fraction x_i of tasks is processed at the edge at site i and v_i edge servers are disabled under the M/M/1 model. Then $D_i(x_i, v_i)$ is convex over the domain $\mathcal{F}_i$.*

Proof of Claim V.1. The expected latency at the edge under attack size v_i and offloading fraction x_i is given by: $D_{\text{edge}}(x_i, v_i) = \frac{1}{\mu - \frac{x_i \cdot \lambda_i}{C_i - v_i}}$. Recall that the expected latency for tasks offloaded to the cloud is denoted by d_{cloud}. Therefore, the expected latency of an arbitrary request at site i is:

$$\begin{aligned} D_i(x_i, v_i) &= x_i \cdot D_{\text{edge}}(x_i, v_i) + (1 - x_i) \cdot d_{\text{cloud}} = \\ &= x_i \cdot \frac{1}{\mu - \frac{x_i \cdot \lambda_i}{C_i - v_i}} + (1 - x_i) \cdot d_{\text{cloud}}. \end{aligned} \tag{14}$$

[3]The exception allows a single site to have $0 < v_j < C_j$ when the attack budget cannot be exactly allocated as full removals C_i across sites.

[4]For ease of notation, we denote $\min_{x \text{ s.t. } (x, v_i) \in \mathcal{F}_i} D_i(x, v_i)$ by $D_i(x_i^\star, v_i)$.

[5]We omit the case where both values are equal. In this case, either the same construction leads to a contradiction, due to the convexity, as in the unequal case, or the values remain equal throughout the construction, in which case we construct a fully concentrated attack which is at least as damaging. See [25] for a similar more detailed discussion.

By direct differentiation, we obtain the second-order partial derivatives:

$$\frac{\partial^2 D_i}{\partial x_i^2} = \frac{2\lambda_i \mu (C_i - v_i)^2}{\left(\mu(C_i - v_i) - \lambda_i x_i\right)^3},$$
$$\frac{\partial^2 D_i}{\partial x_i \partial v_i} = \frac{\partial^2 D_i}{\partial v_i \partial x_i} = \frac{2\lambda_i \mu x_i (C_i - v_i)}{\left(\mu(C_i - v_i) - \lambda_i x_i\right)^3}, \quad (15)$$
$$\frac{\partial^2 D_i}{\partial v_i^2} = \frac{2\lambda_i \mu x_i^2}{\left(\mu(C_i - v_i) - \lambda_i x_i\right)^3}.$$

All terms are non-negative in the domain $\mathcal{F}_i$, since the denominators are positive and all numerators involve only non-negative quantities.

We collect these into the Hessian matrix, $\nabla^2 D_i(x_i, v_i)$:

$$\frac{2\lambda_i \mu}{\left(\mu(C_i - v_i) - \lambda_i x_i\right)^3} \cdot \begin{pmatrix} (C_i - v_i)^2 & x_i(C_i - v_i) \\ x_i(C_i - v_i) & x_i^2 \end{pmatrix}.$$

The determinant of the Hessian is: $(C_i - v_i)^2 \cdot x_i^2 - [x_i(C_i - v_i)]^2 = 0$. Since all diagonal elements of the Hessian are nonnegative and the determinant is nonnegative, the Hessian is positive semi-definite, implying[6] that $D_i(x_i, v_i)$ is convex on $\mathcal{F}_i$. □

Having established that $D_i(x_i, v_i)$ is jointly convex in both x_i and v_i, we now apply a standard result from convex analysis: the partial minimization of a jointly convex function over a convex feasible set preserves convexity [32]. Formally:

Lemma V.1 (Minimization Preserves Convexity [32])**.** *Let $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R} \cup \{+\infty\}$ be jointly convex in (x, v). Then $g(v) = \inf_{x \in \mathbb{R}^n} f(x, v)$ is convex on its effective domain.*

In our context, since $D_i(x_i, v_i)$ is jointly convex and the feasible set $\mathcal{F}_i$ (see (10)) is convex in x_i for each fixed v_i, we conclude the following:

Corollary V.1. *$D_i(x_i^\star, v_i)$ is convex in v_i.*

B. SLA Violation Probability Convexity

We now establish convexity of the SLA violation probability with respect to both the attack size v_i and the offloading fraction x_i.

Claim V.2. *Let $P^t(x_i, v_i)$ denote the probability that the latency of a request exceeds a given threshold $t > 0$, under an M/M/1 model with edge offloading fraction x_i and attack size v_i. Then $P^t(x_i, v_i)$ is convex over the domain $\mathcal{F}_i$.*

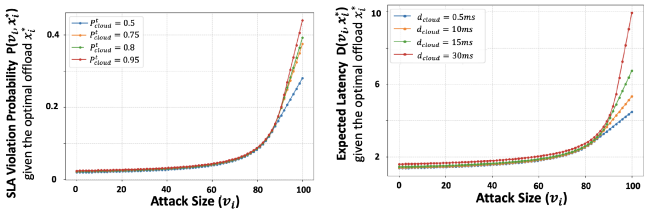
Proof of Claim V.2. The SLA violation probability for requests served at the edge is given by:

$$P^t_{\text{edge}}(x_i, v_i) = e^{-(\mu - \frac{x_i \lambda_i}{C_i - v_i})t}.$$

[6]We use the known Hessian Criterion for Convexity in Two Variables [32]: Let $f : \mathbb{R}^2 \to \mathbb{R}$ be twice continuously differentiable on an open convex set $D \subseteq \mathbb{R}^2$. Let the Hessian matrix of f be given by

$$\nabla^2 f(x, y) = \begin{pmatrix} f_{xx} & f_{xy} \\ f_{yx} & f_{yy} \end{pmatrix}.$$

Then f is convex on D if and only if the Hessian is positive semi-definite in D (i.e., $f_{xx} \geq 0, \quad f_{yy} \geq 0, \quad \det \nabla^2 f = f_{xx} f_{yy} - f_{xy}^2 \geq 0$).



(a) Expected Latency. (b) SLA Violation Probability.

Fig. 5: Numerical validation of analytical results in the M/M/1 setting, for several cloud latency distributions. $C_i = 100, \mu = 1.0, 0 \leq v_i \leq 100$. Both performance metrics are *convex* in the attack size v_i, aligning with Claims V.1 and V.2.

Let P^t_{cloud} denote the violation probability for requests offloaded to the cloud. The overall probability of an arbitrary request having a latency greater than t is given by:

$$P^t(x_i, v_i) = x_i \cdot P^t_{\text{edge}}(x_i, v_i) + (1 - x_i) \cdot P^t_{\text{cloud}} =$$
$$= x_i \cdot e^{-(\mu - \frac{x_i \cdot \lambda_i}{C_i - v_i}) \cdot t} + (1 - x_i) \cdot P^t_{\text{cloud}}.$$

We compute the second-order partial derivatives:

$$\frac{\partial^2 P^t}{\partial x_i^2} = \frac{\lambda_i t}{C_i - v_i} e^{-\left(\mu - \frac{x_i \lambda_i}{C_i - v_i}\right)t} \left(2 + \frac{x_i \lambda_i t}{C_i - v_i}\right),$$
$$\frac{\partial^2 P^t}{\partial x_i \partial v_i} = \frac{x_i \lambda_i t}{(C_i - v_i)^2} e^{-\left(\mu - \frac{x_i \lambda_i}{C_i - v_i}\right)t} \left(2 + \frac{x_i \lambda_i t}{C_i - v_i}\right), \quad (16)$$
$$\frac{\partial^2 P^t}{\partial v_i^2} = \frac{x_i^2 \lambda_i t}{(C_i - v_i)^3} e^{-\left(\mu - \frac{x_i \lambda_i}{C_i - v_i}\right)t} \left(2 + \frac{x_i \lambda_i t}{C_i - v_i}\right).$$

Each expression is non-negative over $\mathcal{F}_i$ (for $C_i > v_i$). The Hessian matrix, $P^t(x_i, v_i)$, is:

$$\left(2 + \frac{x_i t \lambda_i}{C_i - v}\right) e^{-(\mu - \frac{x_i \cdot \lambda_i}{C_i - v_i})t} \frac{t\lambda_i}{C_i - v_i} \begin{pmatrix} 1 & \frac{x_i}{C_i - v_i} \\ \frac{x_i}{C_i - v_i} & \frac{x_i^2}{(C_i - v_i)^2} \end{pmatrix}.$$

The determinant of this matrix is: $\frac{x_i^2}{(C_i - v_i)^2} - \left(\frac{x_i}{C_i - v_i}\right)^2 = 0$. Moreover, all diagonal entries of the Hessian are nonnegative on $\mathcal{F}_i$. Thus, the Hessian is positive semi-definite and $P^t(x_i, v_i)$ is convex on $\mathcal{F}_i$. □

Since $P^t(x_i, v_i)$ is jointly convex in (x_i, v_i), the following corollary follows directly from Lemma V.1.

Corollary V.2. *$P^t(x_i^\star, v_i)$ is convex in v_i.*

The M/M/1 analysis shows that both expected latency and SLA violation probability are jointly convex in the attack and offloading variables. This guarantees that worst-case attacks are concentrated. In Fig. 5, we numerically evaluate both metrics as functions of the attack level, under several cloud latency distributions, to illustrate this convexity.

VI. The M/D/1 Case: Analysis and Numerical Validation

We now analyze the system under the M/D/1 queueing model, in which task arrivals follow a Poisson process (rate λ),

but service times are deterministic with constant rate μ. This model is relevant in scenarios where task execution exhibits low variability, such as inference tasks on pre-loaded models or processing fixed-size sensor frame.

A. Expected Latency Convexity

The expected latency (sojourn time) in an M/D/1 queue is given by a closed-form expression [20]:

$$D_{\text{M/D/1}}(\lambda, \mu) = \frac{1}{\mu} + \frac{\lambda}{2\mu(\mu - \lambda)}. \tag{17}$$

As in the M/M/1 case, we analyze the expected latency as a function of the attack size v_i at site i and the local offloading fraction x_i. Substituting the effective arrival rate $\lambda = \frac{x_i \cdot \lambda_i}{C_i - v_i}$ into (17), we obtain:

$$D_i(x_i, v_i) = x_i \cdot \left(\frac{1}{\mu} + \frac{\frac{x_i \cdot \lambda_i}{C_i - v_i}}{2\mu \left(\mu - \frac{x_i \cdot \lambda_i}{C_i - v_i} \right)} \right) + (1 - x_i) \cdot d_{\text{cloud}}. \tag{18}$$

We now establish the following convexity property:

Claim VI.1. *Let $D_i(x_i, v_i)$ denote the expected latency of a request when a fraction x_i of tasks is processed at the edge at site i and v_i edge servers are disabled under the M/D/1 model. Then $D_i(x_i, v_i)$ is convex over the domain $\mathcal{F}_i$.*

The proof follows by showing that the Hessian matrix of $D_i(x_i, v_i)$ is positive semi-definite throughout $\mathcal{F}_i$. The argument is similar to the convexity proof for the M/M/1 case (Claim VI.1). A derivation of the second-order partial derivatives and the Hessian-based convexity argument is provided in the appendix.

B. SLA Violation Probability Numerical Evaluation

A known expression for the tail probability in the M/D/1 model is given by [18]:

$$P^t_{\text{M/D/1}} = 1 - (1 - \rho)e^{\lambda t} \cdot \sum_{j=0}^{T} \frac{(\rho j - \lambda t)^j}{j!} e^{-(j-1)\rho}. \tag{19}$$

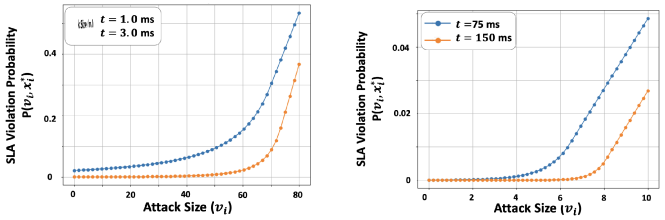
where $T = \lfloor t\mu \rfloor$ denotes the largest integer less than or equal to $t\mu$, and $\rho = \lambda/\mu$ is the system utilization. However, as noted in [18], [19], this expression becomes numerically unsuitable when the load ρ approaches 1 or when computing tail probabilities for large waiting times, due to the presence of large, nearly canceling terms in the summation.

To handle the high-utilization regime ($\rho \to 1$), approximations have been proposed. In particular, [33] provided an approximation under processor-sharing assumptions, relevant to modern workloads such as inference tasks or CPU-bound jobs, which can exhibit parallelism.[7] When $\rho \to 1$ and $t \to \infty$, the tail probability can be approximated as [33]:

$$P^t_{\text{M/D/1}} \approx e^{-\lambda \cdot (1-\rho) \cdot t}, \text{ when } \rho \to 1. \tag{20}$$

This approximation is particularly relevant to our setting, where an attacker may degrade the edge capacity such that the effective utilization becomes high (i.e., $\rho \to 1$). We use both expressions to examine the behavior of the SLA violation probability with respect to attack size and offloading fraction. We numerically evaluate both the exact expression in (19) and the approximation in (20), and in both cases observe that the SLA violation probability appears convex in the attack size v_i.

[7] Known as processor sharing in the queueing literature, see [34].



(a) Exact expression from (19) (low load, $\rho = 0.2$).

(b) Approximation from (20) (heavy load, $\rho \to 1$).

Fig. 6: Numerical illustration of convexity in the SLA violation probability under the M/D/1 model. The figure plots $\min_{x_i} P(x_i, v_i)$ as a function of the attack size v_i for different deadline thresholds t and load regimes.

Fig. 6 presents numerical evaluations of the optimized SLA violation probability $P(x_i^\star, v_i)$ under the M/D/1 model. Fig. 6a show results using the exact tail expression from (19), for relatively low load ($\rho = 0.2$) Fig. 6b uses the heavy-load approximation from (20), which takes a simplified exponential form. We evaluate scenarios where $\rho \to 1$ and t is relatively high. In both cases, the violation probability curves exhibit empirically convex behavior as a function of the attack size v_i, across several deadline thresholds t.

VII. General Service Distribution: M/G/1

To account for general task service time distributions, we now consider the M/G/1 queueing model. This model assumes that task arrivals follow a Poisson process with rate λ, while the service times are drawn from a general distribution S. This setting is relevant for modeling systems with non-deterministic, but non-exponential, execution durations (e.g., I/O-bound workloads or variable-sized inference jobs).

Unlike the M/M/1 or M/D/1 models, the M/G/1 queue lacks closed-form expressions for the latency distribution, which depends on the specific service-time distribution. However, the expected latency has a well-known formula, which relies on the expectation and variance of the service distribution S.

By the Pollaczek-Khinchin formula [20], the expected time in system (sojourn time) is given by:

$$D_{\text{M/G/1}}(\lambda) = \mathbb{E}[S] + \frac{\lambda \cdot \mathbb{E}[S^2]}{2(1 - \lambda \cdot \mathbb{E}[S])}. \tag{21}$$

Claim VII.1 (Convexity of M/G/1 Sojourn Time under Attack)**.** *Let $D_i(x_i, v_i)$ be defined as*

$$D_i(x_i, v_i) = x_i \cdot \left(\frac{1}{\mu} + \frac{\frac{\lambda_i x_i}{C_i - v_i} \cdot \mathbb{E}[S^2]}{2\left(1 - \frac{\lambda_i x_i}{C_i - v_i}\right)} \right) + (1 - x_i) \cdot d_{cloud}.$$

Then $D_i(x_i, v_i)$ is convex over the domain $\mathcal{F}_i$.

The proof follows by arguments similar to those used in the M/M/1 and M/D/1 cases. For completeness, a concise outline of the proof is provided in the appendix.

Corollary VII.1. *$D_i(x_i^\star, v_i)$ is convex in v_i under general service distributions (M/G/1 model).*

This result is particularly significant because it extends the understanding of system behavior beyond highly specific queueing models, showing that the structural properties of worst-case latency attacks persist under general service-time distributions.

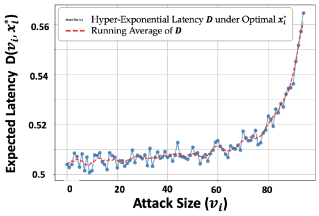
VIII. Simulation and Numerical Evaluation

A. General Service Distributions

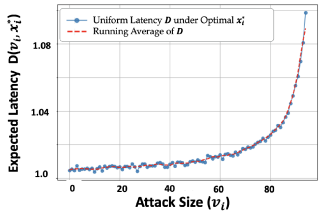
We numerically demonstrate the convexity of the expected latency in the M/G/1 queue under different service-time distributions. These simulations aim to confirm that the convexity behavior observed analytically in the M/M/1 case generalizes to broader settings where service times are not exponentially distributed.

We consider four representative service-time distributions, each capturing a distinct aspect of real-world workloads: (i) *Hyper-exponential distribution* [35]: Models workloads with high variance and heavy tails, typical in network traffic, cloud services, and GPU job scheduling. Captures the presence of infrequent but extremely long tasks. (ii) *Uniform distribution* [36]: Represents bounded and moderately variable workloads, such as fixed-size packet transmissions or compute-bound inference with predictable execution times. (iii) *Pareto distribution* [37]: A classic heavy-tailed model used in stress-testing scenarios such as DDoS modeling, file transfers, or adversarial task sizes. Reflects extreme variability and bursty job patterns. (iv) *Log-normal distribution* [38]: Empirically observed in many real-world systems, including file sizes, job latencies, and GPU kernel execution times. Captures heavy-tailed behavior with finite variance.

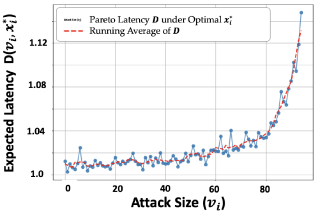
Each plot in Fig. 7 shows the simulated average latency $D_i(x_i, v_i)$ as a function of the offloading fraction x_i and attack size v_i, under different service time distributions. For each configuration, the simulation was repeated 10 times with 100,000 arrivals per run. Service times were drawn from the specified distribution using standard Python `random` and `numpy` libraries. A running average (window size 5, in red) is applied to reduce noise. Across all distributions, the latency exhibits convexity in (x_i, v_i), in agreement with ClaimVII.1. Additionally, Fig. 7e shows the SLA violation count from the simulations, which also displays convex behavior.



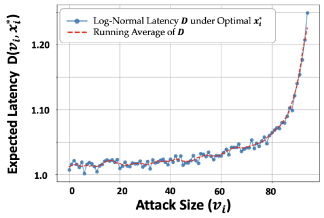
(a) Hyper-exponential service rate (high variability).



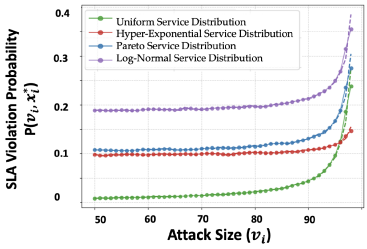
(b) Uniform service rate (bounded variability).



(c) Pareto service rate (heavy-tailed).



(d) Log-normal service rate (long right tail).



(e) SLA violation count across all four distributions.

Fig. 7: Simulations under various service-time distributions in M/G/1 systems. Despite differences in variability and tail behavior, convexity in the attack size v_i is consistently observed, supporting Claim VII.1. Subfigs. (a)-(d) show expected latency $D_i(x_i^\star, v_i)$ as a function of v_i. Subfig, (e) reports the corresponding SLA violation count across all distributions.

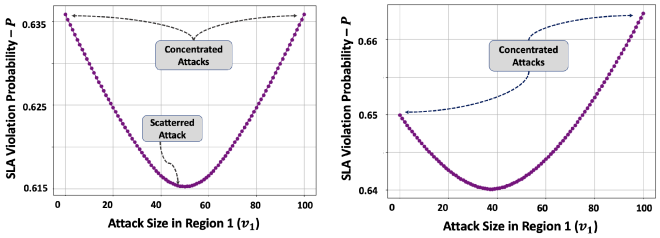
B. Multi-Site Environments

We consider a multi-site system and demonstrate that concentrated attacks consistently result in greater performance degradation compared to dispersed attacks, confirming the structural convexity properties established in prior sections.

We begin by examining a two-site system and evaluate attack strategies ranging from fully concentrated (e.g., $(100\%, 0\%)$) to evenly distributed (e.g., $(50\%, 50\%)$), by varying v_1 from 0 to 100 and setting $v_2 := 100 - v_1$. Fig. 8 shows the resulting system delay under each configuration.

Fig. 8a corresponds to a symmetric setting (equal load and capacity in both sites), while Fig. 8b considers an asymmetric configuration. In both cases, the most severe damage occurs when the attack is fully concentrated on a single site, consistent with the convexity of the performance function. In the symmetric case, either site can be targeted with the same result; in the asymmetric case, the attacker prefers the site with relatively lower capacity compared to its load.

Next, we evaluate a system with five sites and compare attack strategies that distribute a fixed attack budget across one to five sites under different cloud delay values. To study the impact of attacks under realistic network conditions, we evaluate system behavior using latency values reported in prior measurement studies that quantify typical user-to-edge and user-to-cloud delays. Specifically, we adopt edge latencies on the order of 5-10 ms from end devices [39], while cloud



(a) Symmetric case: both sites have equal capacity and demand.

(b) Asymmetric case: one site has lower capacity relative to its load.

Fig. 8: Impact of varying attack concentration in a two-site system, in symmetric and asymmetric settings. Performance degradation is maximized when the attack is fully concentrated on a single site, in accordance with the analytical results.

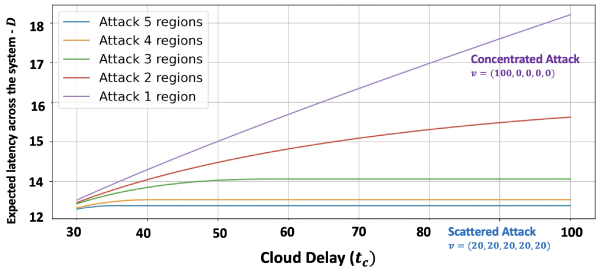


Fig. 9: Impact of attack spread across a five-site system, shown as a function of cloud delay. Concentrated attacks consistently induce the highest performance degradation.

communication incurs delays in the tens of milliseconds, typically around 50-100 ms [39], [40].

The attack vector is varied from a fully concentrated strategy (all in percentage of attack value): $(100, 0, 0, 0, 0)$ (purple curve in Fig. 9), to increasingly dispersed strategies: $(\frac{1}{2}, \frac{1}{2}, 0, 0, 0)$ (red), $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}, 0, 0)$ (green), $(\frac{1}{4}), \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, 0$ (orange), and the fully dispersed case $(\frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5})$ (blue). As depicted in Fig. 9, the fully concentrated attack consistently yields the highest damage, regardless of the cloud delay parameter d_{cloud}. Moreover, as the cloud latency increases, the damage increases as well, due to increased cloud offloading. In particular, when cloud latency is roughly 100 ms [39], for the same total attack budget the concentrated attack results in up to a 38% increase in damage, raising the expected latency from 13.2 ms to 18.3 ms.

IX. Concluding Remarks and Future Work

This paper presented a theoretical framework for analyzing worst-case attacks in reactive edge-cloud systems, modeled as a max-min game between attacker and defender to capture the interplay between the attack and offloading mitigation.

Through a combination of queueing models (M/M/1, M/D/1, M/G/1) analysis and numerical simulations we showed that two key performance metrics, expected latency and SLA violation probability, are convex in the attack vector. This convexity enables a powerful characterization of worst-case attacks, which we show to be *concentrated*, i.e., targeting a small number of critical edge sites rather than distributing effort widely.

These attack structures provide practical "rules of thumb" for identifying edge–cloud vulnerabilities. The resulting structural insights reduce the computational burden of identifying damage-maximizing attack vectors and enable effective heuristics even under partial information. Under typical edge–cloud latency assumptions drawn from prior measurement studies, our numerical results show that such worst-case attack strategies can increase system damage by up to 40% using the same attack budget.

The worst-case characterization developed in this work establishes a rigorous performance baseline that can inform the design and operation of edge–cloud systems under adversarial conditions. Future work may extend the framework to trace-driven arrivals and heterogeneous edge capacities, and investigate how inter-site offloading and coordination among edge sites influence system resilience.

X. Acknowledgments

This work was supported in part by NSF and Center for Smart Streetscapes (CS3) under NSF Cooperative Agreement EEC-2133516, NSF grant CNS-2038984 and corresponding support from the Federal Highway Administration (FHWA), NSF grant CNS-2148128, funds from federal agency and industry partners as specified in the Resilient & Intelligent NextG Systems (RINGS) program, support from MediaTek USA Inc., and by the Zuckerman STEM Leadership Program.

Appendix

Proof of Claim VI.1. By direct differentiation, we obtain the Hessian matrix of $D_i(x_i, v_i)$, $\nabla^2 D_i(x_i, v_i)$:

$$\frac{\lambda_i \cdot \mu}{(\mu(C_i - v_i) - \lambda_i x_i)^3} \cdot \begin{pmatrix} (C_i - v_i)^2 & x_i \cdot (C_i - v_i) \\ x_i \cdot (C_i - v_i) & x_i^2 \end{pmatrix}.$$

All terms in the matrix are non-negative within the domain $\mathcal{F}_i$. Moreover, the determinant is:

$$\det \nabla^2 D_i(x_i, v_i) = (C_i - v_i)^2 \cdot x_i^2 - (x_i \cdot (C_i - v_i))^2 = 0, \quad (22)$$

which implies that the Hessian is positive semi-definite but rank-deficient. Thus, we conclude that $D_i(x_i, v_i)$ is convex over $\mathcal{F}_i$. □

Proof of Claim VII.1. By direct differentiation, we obtain the Hessian matrix of $D_i(x_i, v_i)$, $\nabla^2 D_i(x_i, v_i)$:

$$= \frac{\mathbb{E}[S] \cdot \lambda_i^2 \cdot \mu^2}{(\mu(C_i - v_i) - \lambda_i x_i)^3} \cdot \begin{pmatrix} (C_i - v_i)^2 & x_i \cdot (C_i - v_i) \\ x_i \cdot (C_i - v_i) & x_i^2 \end{pmatrix}.$$

All terms in the matrix are non-negative within the domain $\mathcal{F}_i$. Moreover, the determinant is:

$$\det \nabla^2 D_i(x_i, v_i) = (C_i - v_i)^2 \cdot x_i^2 - (x_i \cdot (C_i - v_i))^2 = 0, \quad (23)$$

which implies that the Hessian is positive semi-definite. Thus, we conclude that $D_i(x_i, v_i)$ is convex over $\mathcal{F}_i$. □

REFERENCES

[1] J. Pan and J. McElhannon, "Future edge cloud and edge computing for internet of things applications," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 439–449, 2017.

[2] M. Trigka and E. Dritsas, "Edge and cloud computing in smart cities," *Future Internet*, vol. 17, no. 3, p. 118, 2025.

[3] M. Ghasemi, Y. Fu, X. Ouyang, P. Wang, M. K. Turkcan, J. Tavori, S. Kleisarchaki, T. Calmant, L. Gürgen, Z. Kostic, X. S. Di, G. Zussman, and J. Ghaderi, "Real-time video analytics for urban safety: Deployment over edge and end devices," in *Proc. ACM/IEEE SEC'25 (Symposium on Edge Computing)*, Dec. 2025.

[4] J. Wang, J. Pan, F. Esposito, P. Calyam, Z. Yang, and P. Mohapatra, "Edge cloud offloading algorithms: Issues, methods, and perspectives," *ACM Computing Surveys*, vol. 52, no. 1, pp. 1–23, 2019.

[5] B. Wang, C. Wang, W. Huang, Y. Song, and X. Qin, "A survey and taxonomy on task offloading for edge-cloud computing," *IEEE Access*, vol. 8, pp. 186 080–186 101, 2020.

[6] S. Wang, Y. Zhao, J. Xu, J. Yuan, and C.-H. Hsu, "Edge server placement in mobile edge computing," *Journal of Parallel and Distributed Computing*, vol. 127, pp. 160–168, 2019.

[7] Y. Zhang, X. Lan, Y. Li, L. Cai, and J. Pan, "Efficient computation resource management in mobile edge-cloud computing," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 3455–3466, 2018.

[8] Y. Zhang, X. Lan, J. Ren, and L. Cai, "Efficient computing resource sharing for mobile edge-cloud computing networks," *IEEE/ACM Trans. Netw.*, vol. 28, no. 3, pp. 1227–1240, 2020.

[9] J. Ren, G. Yu, Y. He, and G. Y. Li, "Collaborative cloud and edge computing for latency minimization," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 5, pp. 5031–5044, 2019.

[10] J. Tavori and H. Levy, "How to attack and congest delay-sensitive applications on the cloud," in *Proc. IEEE INFOCOM'23*, May 2023.

[11] A. Ebrahimzadeh and M. Maier, "Cooperative computation offloading in fiwi enhanced 4g hetnets using self-organizing mec," *IEEE Transactions on Wireless Communications*, vol. 19, no. 7, pp. 4480–4493, 2020.

[12] S. Xin, L. Zhuo, and C. Xin, "Online node cooperation strategy design for hierarchical federated learning," in *Proc. IEEE INFOCOM'22 Workshops*, May 2022.

[13] B. Ma, Z. Ren, and W. Cheng, "Traffic routing-based computation offloading in cybertwin-driven internet of vehicles for v2x applications," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 5, pp. 4551–4560, 2021.

[14] S. Yao, Y. Lin, M. Wang, K. Xu, M. Xu, C. Xu, and H. Zhang, "Leoedge: A satellite-ground cooperation platform for the ai inference in large leo constellation," *IEEE Journal on Selected Areas in Communications*, vol. 43, no. 1, pp. 36–50, 2024.

[15] P. Du, F. Pang, T. Braun, M. Gerla, C. Hoffmann, and J. H. Kim, "Traffic optimization in software defined naval network for satellite communications," in *Proc. IEEE MILCOM'17*, Oct. 2017.

[16] X. Ma, A. Zhou, S. Zhang, and S. Wang, "Cooperative service caching and workload scheduling in mobile edge computing," in *Proc. IEEE INFOCOM'20*, May 2020.

[17] W. Fan, L. Gao, Y. Su, F. Wu, and Y. Liu, "Joint dnn partition and resource allocation for task offloading in edge–cloud-assisted iot environments," *IEEE Internet of Things Journal*, vol. 10, no. 12, pp. 10 146–10 159, 2023.

[18] V. B. Iversen and L. Staalhagen, "Waiting time distribution in m/d/1 queueing systems," *Electronics Letters*, vol. 35, no. 25, pp. 2184–2185, 1999.

[19] F. Ciucu, "Network calculus delay bounds in queueing networks with exact solutions," in *Proc. International Teletraffic Congress*, June 2007.

[20] L. Kleinrock, "Queueing systems, volume 2: Computer applications," 1976.

[21] Y. Ma, L. Liu, Z. Liu, F. Li, Q. Xie, K. Chen, C. Lv, Y. He, and F. Li, "A survey of ddos attack and defense technologies in multi-access edge computing," *IEEE Internet of Things Journal*, 2024.

[22] C. Peng and Q. Zhu, "Trust-aware resource management for secure and optimal network slicing in 5g mobile edge networks," in *Proc. IEEE INFOCOM'23 Workshops*, May 2023.

[23] X. Xia, F. Chen, Q. He, R. Luo, B. Liu, C. Chua, R. Buyya, and Y. Yang, "Edgeshield: Enabling collaborative ddos mitigation at the edge," *IEEE Transactions on Mobile Computing*, vol. 23, no. 12, pp. 14 502–14 513, 2024.

[24] Z. Zhou, S. Yu, W. Chen, and X. Chen, "Ce-iot: Cost-effective cloud-edge resource provisioning for heterogeneous iot applications," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 8600–8614, 2020.

[25] J. Tavori and H. Levy, "Tornadoes in the cloud: Worst-case attacks on distributed resources systems," in *Proc. IEEE INFOCOM'21*, May 2021.

[26] M. Becchi, "From poisson processes to self-similarity: a survey of network traffic models," *Washington University in St. Louis, Tech. Rep*, 2008.

[27] A. Ali-Eldin, B. Wang, and P. Shenoy, "The hidden cost of the edge: a performance comparison of edge and cloud latencies," in *Proc. ACM SC'21*, Nov. 2021.

[28] K. Khalafi and N. Lu, "Network slicing for edge-cloud orchestrated networks via online convex optimization," in *Proc. IEEE INFOCOM'24 Workshops*, May 2024.

[29] A. Chagdali, S. E. Elayoubi, A. M. Masucci, and A. Simonian, "Performance of URLLC traffic scheduling policies with redundancy," in *Proc. IEEE ITC (International Teletraffic Congress)*, Sep. 2020.

[30] M. Abdullah, S. E. Elayoubi, and T. Chahed, "Efficient queue control policies for latency-critical traffic in mobile networks," *IEEE Transactions on Network and Service Management*, vol. 21, no. 5, pp. 5076–5090, 2024.

[31] J. A. Peris and V. Fodor, "Resource dimensioning for single-cell edge video analytics," in *Proc. IEEE ICC'23*, May 2023.

[32] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[33] R. Egorova, B. Zwart, and O. Boxma, "Sojourn time tails in the m/d/1 processor sharing queue," *Probability in the engineering and informational sciences*, vol. 20, no. 3, pp. 429–446, 2006.

[34] S. F. Yashkov and A. Yashkova, "Processor sharing: A survey of the mathematical theory," *Automation and Remote Control*, vol. 68, no. 9, pp. 1662–1731, 2007.

[35] J. F. Shortle, "An equivalent random method with hyper-exponential service," *Performance Evaluation*, vol. 57, no. 3, pp. 409–422, 2004.

[36] W. Wang, B. Liang, and B. Li, "Multi-resource generalized processor sharing for packet processing," in *Proc. IEEE/ACM IWQoS'13*, June 2013.

[37] X. Zhou, J. Wei, and C.-Z. Xu, "Processing rate allocation for proportional slowdown differentiation on internet servers," in *Proc. IEEE IPDPS'04*, Apr. 2004.

[38] H. Qian, C. S. Surapaneni, S. Dispensa, and D. Medhi, "Service management architecture and system capacity design for phonefactor™—a two-factor authentication service," in *Proc. IFIP/IEEE IM'09*, June 2009.

[39] B. Charyyev, E. Arslan, and M. H. Gunes, "Latency comparison of cloud datacenters and edge servers," in *Proc. IEEE GLOBECOM'20*, Dec. 2020.

[40] S. Choy, B. Wong, G. Simon, and C. Rosenberg, "The brewing storm in cloud gaming: A measurement study on cloud to end-user latency," in *Proc. IEEE Netgames'12 (Annual Workshop on Network and Systems Support for Games)*, Nov. 2012.