Explaining the Write Performance of Kafka under Hardware-Related Chaos Scenarios

Mika Petersen
stu245087@mail.uni-kiel.de
Christian-Albrechts-University, Kiel, DE

Hendrik Reiter
hendrik.reiter@email.uni-kiel.de
Christian-Albrechts-University, Kiel, DE

Wilhelm Hasselbring
hasselbring@email.uni-kiel.de
Christian-Albrechts-University, Kiel, DE

Abstract

Apache Kafka is a widely adopted distributed event streaming platform in both industry and academia. Edge computing is a rapidly evolving research field characterized by hardware constraints and an increased risk of failures. This paper evaluates the impact of hardware-related chaos scenarios on Kafka's write-path performance between producers and brokers in a simulated edge-computing environment. This study aims to characterize the performance impacts of different chaos scenarios injected using Ecoscape, an edge reliability benchmark framework. The experiment applied different levels of CPU stress. Our results show that CPU stress has a substantial impact on the data throughput processed by the producers and brokers. Furthermore, this effect propagates differently across Kafka's producers and brokers, resulting in both performance improvements and degradations, depending on the affected component. This highlights the need for fine-grained monitoring of Kafka's components is necessary to assess the interaction between hardware conditions and system performance when deploying stream-processing systems at the edge.

1 Introduction

The transition from centralized systems for multi-source image analysis into partly decentralized architectures can reduce bandwidth consumption and enhance data privacy [3]. Edge computing enables collected data to be processed closer to its source, allowing images or sensor data to be (partly) analysed on resource-constrained hardware components [4]. Afterwards, relevant results are transmitted to central server. However, the decentralization and limited resources of edge environments introduce additional reliability challenges. Edge devices are more vulnerable to hardware-related failures, and network disruptions, due to their limited resources, which can affect the performance and availability [1]. Event streaming platforms play an important role in such systems by enabling the efficient and reliable exchange of data between distributed components. Apache Kafka, for example, can be used to transport and process continuous streams of collected data. Understanding how such systems behave under edge-specific hardware faults is therefore essential for developing and applying edge-environment systems.

Research on edge computing is highly active, particularly in the context of the Industrial Internet of Things part (IIoT), as it aims to transfer the benefits from IoT technologies to industrial applications. Apache Kafka, as a widely adopted distributed event streaming platform, also continuous to be extensively investigated. However, the intersection of these two research areas remains comparatively underexplored, as Kafka is primarily designed for large scale server systems. This paper addresses this research gap by providing empirical data and corroborating other research about Kafka's ability to handle edge specific failure scenarios.

We combine a simulated edge environment with induced hardware-related fault scenarios to investigate Kafka's behaviour under changing environmental conditions. This empirical research measures the degradation of Kafka's writing performance under CPU stress. CPU stress is a typical edge related problem due to thermal throttling, task co-location on CPU-limited hardware, or fluctuating power supply [1].

The remainder of this paper is structured as follows: Section 2 introduces the foundations of Kafka and Ecoscape, while Ssection 3 presents the experiment design and presents the results. Section 4 discusses the results and related work, Section 5 concludes the paper.

2 Foundation

Kafka. Apache Kafka[1] is a distributed event-streaming platform based on the publish-subscribe pattern, where brokers handle data processing and

[1] https://developer.confluent.io/courses/architecture/broker/

Metric	50%	70%	90%
NetworkIOThread	-29.9%	-42.4%	-55.1%
BytesInPerSec	-30.8%	-38.5%	-53.8%
LocalTime-Produce	-31.4%	-48.5%	-60.0%
RemoteTime-Produce	32.0%	45.7%	54.8%
RequestQT-Produce	357.1%	940.0%	700.0%
ResponseQT-Produce	500.0%	850.0%	920.0%
LocalTime-FetchF.	50.0%	38.5%	60.0%
RemoteTime-FetchF.	-2.8%	-1.1%	-2.2%
RequestQT-FetchF.	352.9%	600.0%	847.1%
ResponseQT-FetchF.	515.0%	758.8%	912.5%

Table 1: Median difference in percent between the baseline and the CPU stress phase.

storage. When a client sends a produce request, it lands in a network buffer before a network thread places it into the request queue. An I/O thread then writes the messages to the operating system page cache for high-throughput persistence. To satisfy replication requirements, the produce request is temporarily parked in a delayed operation structure called the purgatory, where it waits asynchronously for follower replicas to fetch and acknowledge the data. Once acknowledged by the required in-sync replicas, a response is routed through the response queue back to the client. Under hardware faults such as CPU stress, disk I/O contention, or network latency, processing bottlenecks emerge as measurable queueing delays across these internal processing stage.

Ecoscape. Ecoscape [2] is a Kubernetes-based benchmark designed to evaluate performance resilience in fault-prone edge computing environments. It simulates heterogeneous edge topologies by injecting controlled infrastructure faults (e.g CPU stress) using Chaos Engineering via Chaos Mesh. CPU stress is injected on *cgroup*-level to effectively target the workload to simulate the effects of thermal throttling, task co-location, or limited power supply. Ecoscape automates experiment orchestration to quantify the impact of these faults on domain-specific Service Level Objectives (SLOs) through a normalized violation score. Each benchmark run systematically progresses through a warm-up, evaluation, chaos injection, and tear-down phase. Consequently, Ecoscape provides a standardized and reproducible framework for evaluating performance resilience strategies, making it ideal for benchmarking dynamic edge systems.

3 Experiment Design and Results

Experiment Design. The experiment were conducted on simulated edge devices hosted on a server at the CAU Kiel. The CPU stress scenario was applied after 50 seconds and removed after 110s (CPU stress phase is indicated red in Figures 1 and 2) to all three Kafka brokers, and measurements were aggregated over 10-seconds intervals. CPU Stress was applied at utilization rates 90% using two stress workers. Each utilization rate was executed five times, and the resulting measurements are reported using the minimum, maximum and median values.

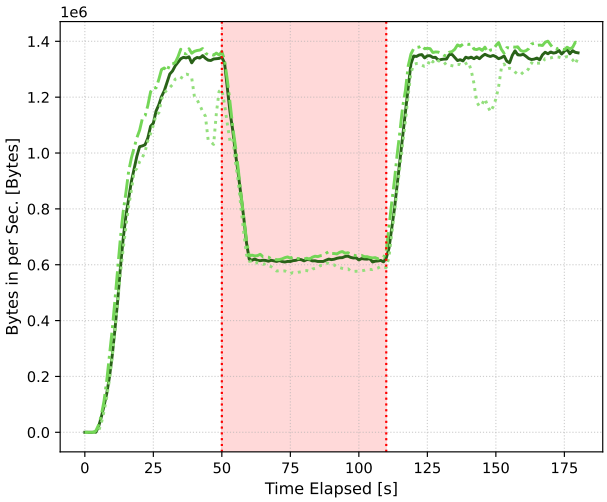

Figure 1: Median of bytes in per second at 90% CPU stress. The CPU stress phase is indicated in red.

Results. We show a selection of metrics which aim to represent our empirical evaluation. The collected data and the resulting diagrams are provided at Zenodo.[2] The time series plots contain three types of values: the dotted line represents the minimum value, the continuos line represents the median, and the dash dotted line represents the maximum value. Figure 1 shows that the bytes per second decrease up to 50% during the chaos phase. The effect of induced chaos increases the response queue time by up to 900% as shown in Figure 2. Table 1 presents the median values measured during the chaos phase relative to the median values measured during the non-chaos phase. The relative values are calculated as follows:

$$\left(\frac{\text{median}_{\text{chaos}}}{\text{median}_{\text{non-chaos}}} * 100\right) - 100$$

4 Discussion & Related Work

Discussion. The decrease of Network I/O threads, or conversely, the increase in idle network threads can be attributed to the reduction of processing capacity caused by the applied CPU stress. As the systems processes less data, buffers can become saturated, resulting in fewer incoming bytes as observed.

For the produce-request, local time decreases, which is consistent with fewer incoming bytes requiring less processing and disk activity. In contrast, the increase in remote time may result from less CPU time available in the brokers acknowledging the requests. Request and response queue time increases

[2] https://zenodo.org/records/22084676

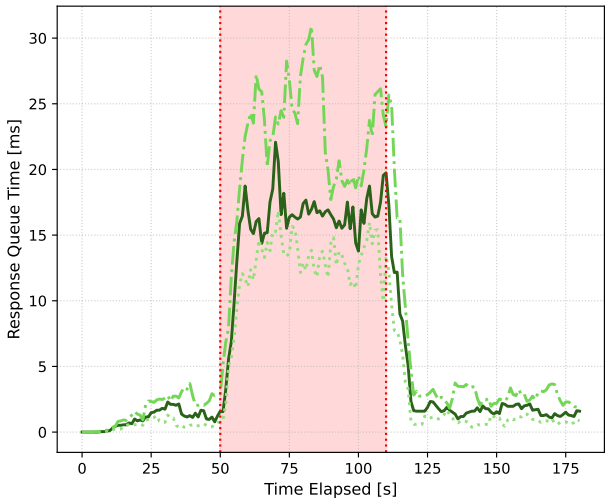

Figure 2: Median of response queue time of the fetch follower request at 90% CPU stress.

significantly, indicating increased saturation of the I/O threads.

For the fetch-follower metrics, the local time increases because larger batches require more time to be read from disk, thereby increasing disk pressure. Remote time decreases only marginally which may be explained by increased batch sizes while the number of requests remains approximately constant. Request and response queue times again increases significantly, indicating that the I/O threads become a bottleneck.

Threats To Validity. The conducted experiments have internal and external treads to validity:

Internal Validity: The simulation environment was hosted on a shared server, which could introduce resource contention or background noise affecting timing metrics. To mitigate this risk, all experiments were repeated five times, and median values were reported alongside minimum and maximum bounds.

External Validity: The fault injection scenarios relied on a simulated edge environment deployed via Kubernetes rather than bare-metal constrained edge devices. While Chaos Mesh provides accurate resource limits, actual IoT hardware may exhibit additional performance bottlenecks (e.g., network latency, slower I/O).

Related Work. Szydlo et al. [3] proposed a benchmark for evaluating CPU rates in relation to CPU efficiency under hardware constrains typical of edge devices. Their results show that CPU optimization can lead to significant improvements in efficiency. Although their research motivation differs from ours, their findings indicate a strong relationship between CPU performance and the overall performance of edge devices.

Codemo and Pahl [4] conducted a Kafka benchmark on Raspberry Pi devices. Their results show that Kafka provides a high degree of resilience to crashes under induced stress. They further identify CPU saturation as a primary performance limitation in their experimental scenario. While their work evaluates Kafka on real edge hardware, our study focuses on the impact of specific hardware-related fault scenarios on Kafka's write-path performance.

5 Conclusion

Hardware faults must be considered when deploying on resource-constrained hardware components. In our experiments, Kafka was substantially affected by the induced chaos, including unexpected changes across multiple metrics. These results demonstrate that fine-grained monitoring is essential for understanding how a system responds to changing hardware conditions. With our test scenario, CPU stress has a significant impact on the total throughput and the queue time.

Future work will deploy Kafka on real-world IoT hardware to validate the results obtained in simulation. Further research could also test different broker configurations or compare Kafka with other distributed event streaming platforms, such as Apache Pulsar,[3] under similar fault scenarios. Such a comparison may help identify strengths and limitations, and determine their suitability for specific IIoT use cases.

Acknowledgments

This work received funding from the Deutsche Forschungsgemeinschaft (DFG), grant 496119880.

References

[1] M. Pourreza and P. Narasimhan. "A Survey of Faults and Fault-Injection Techniques in Edge Computing Systems". In: *2023 IEEE International Conference on Edge Computing and Communications (EDGE)*. 2023, pp. 63–71.

[2] H. Reiter et al. "Ecoscape: Fault Tolerance Benchmark for Adaptive Remediation Strategies in Real-Time Edge ML". In: *2025 IEEE International Conference on Quantum Software (QSW)*. IEEE, 2025, pp. 275–280.

[3] T. Szydlo et al. *Benchmarking of CPU-intensive Stream Data Processing in The Edge Computing Systems*. 2025.

[4] M. Codemo and C. Pahl. "Performance Benchmarking for Data Stream Processing on Constrained Edge Devices". In: *Proceedings of the 16th International Conference on Cloud Computing and Services Science*. SCITEPRESS - Science and Technology Publications, 2026, pp. 352–361.

[3] https://pulsar.apache.org/