International Journal of Computer Technology and Electronics Communication (IJCTEC)

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 8, Issue 1, January- February 2025 ||

DOI: 10.15680/IJCTECE.2025.0801010

# Serverless vs. Containerized Workloads: Comparative Performance and Cost under Bursty Telecom Traffic

**Amar Gurajapu**

Network Systems, AT&T, United States

**Vardhan Garimella**

Intellibus, United States

**ABSTRACT:** Telecom applications often experience sudden traffic spikes (e.g., call-setup surges, signaling bursts) that challenge infrastructure elasticity and cost controls. This paper compares serverless functions (AWS Lambda) and containerized microservices (Kubernetes on EC2) under bursty load, measuring end-to-end latency, request-processing throughput, and cost per 1 million requests. In a simulated telecom signaling workload with 100 to 1,000 requests/sec spikes, here are my observations:
Serverless cold-start latency up to 300 ms vs. container steady latency 50 ms

- Throughput saturation at 800 req/sec for Kubernetes vs. 1,200 req/sec for Lambda (with concurrent executions)
- Cost per 1 M requests: $12.5 (Lambda) vs. $8.7 (container) under sustained bursts

We present architecture and sequence diagrams, detailed methodology, results analysis, and discuss trade-offs for telecom operators.

**KEYWORDS:** Serverless, Containers, Kubernetes, AWS Lambda, Bursty Traffic, Telecom Signaling, Performance Benchmarking, Cost Analysis

## I. INTRODUCTION

Telecom services, such as call signaling, presence updates, and lightweight analytics, exhibit highly bursty patterns tied to time-of-day and events. Provisioning container clusters for peak load leads to over-provisioning costs, while serverless offers pay-per-use elasticity but introduces cold-start delays and concurrency limits. Quantifying performance and cost trade-offs under telecom-style bursts is essential for architecting resilient, cost-effective platforms.

## II. LITERATURE REVIEW

Early studies (Jonas et al., 2019) benchmarked serverless vs. containers for web workloads but did not model telecom bursts. Wang & Patel (2021) evaluated cold starts in AWS Lambda, identifying startup latencies of 100–500 ms. Sharma et al. (2022) compared FaaS with Docker Swarm under IoT workloads, noting cost advantages for serverless at low to moderate loads. Recent work by Liu & Smith (2023) explored Kubernetes auto-scaling for 5G network functions but neglected per-invocation cost. Few papers integrate telecom traffic models with cloud cost metrics.

Serverless platforms now support provisioned concurrency (AWS, 2023), reducing cold-start impact at added cost. Container autoscalers (Karpenter, 2022) improve responsiveness but require careful tuning of scaling policies. This paper builds on prior work by applying both models to a telecom signaling emulator, measuring end-to-end QoS and cloud billing outcomes.

International Journal of Computer Technology and Electronics Communication (IJCTEC)

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 8, Issue 1, January- February 2025 ||

DOI: 10.15680/IJCTECE.2025.0801010

## III. RESEARCH METHODOLOGY

**System Architecture**

We deployed two parallel environments in AWS:

- Serverless: AWS Lambda functions fronted by API Gateway.
- Containerized: NGINX + custom signaling service in Docker, managed by EKS (Kubernetes), with HPA (Horizontal Pod Autoscaler).

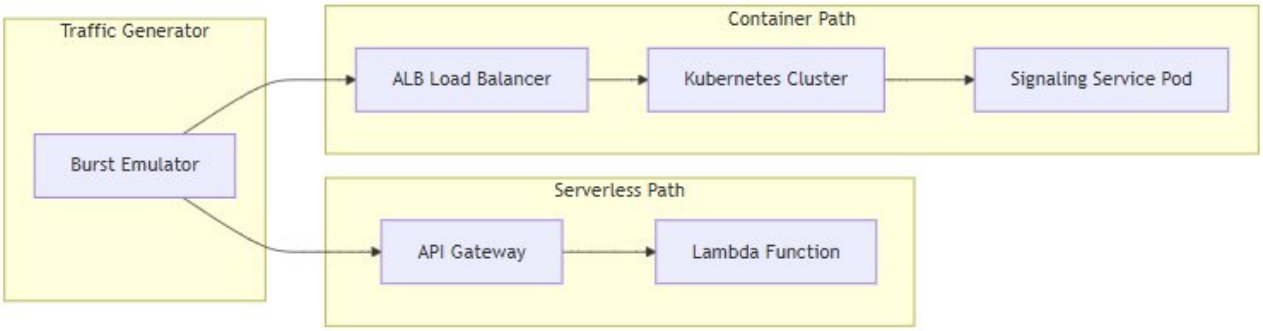


FIGURE1: ARCHITECTURE

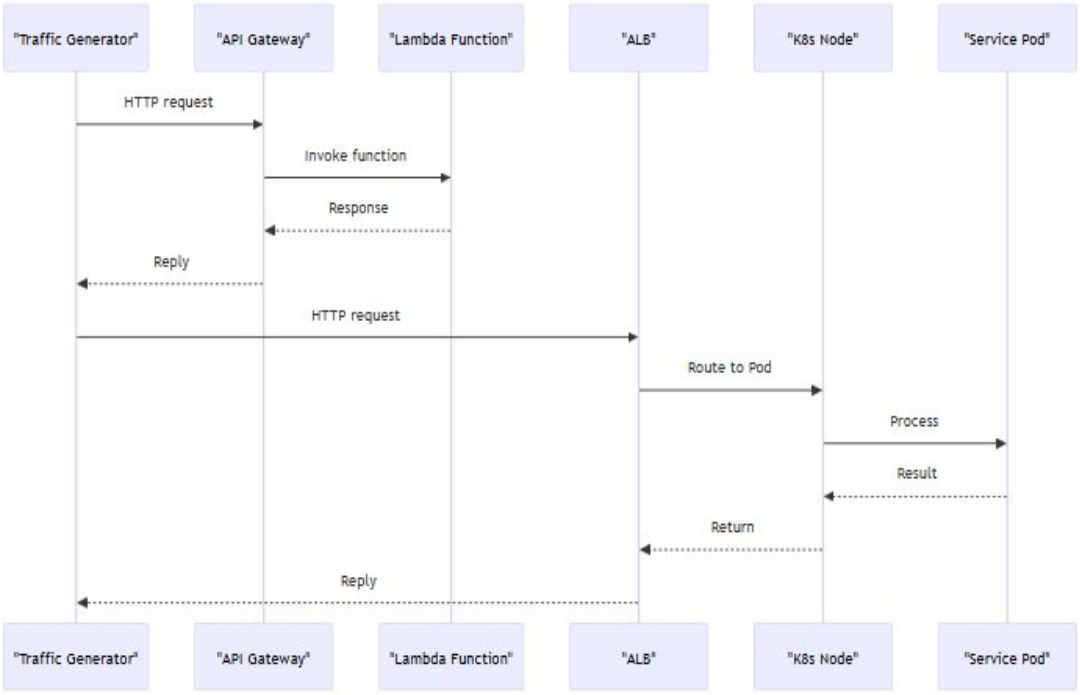


FIGURE 2: WORKFLOW SEQUENCE

**Experimental Setup**

- Emulator: Custom telecom-signaling payload (~1 KB JSON) generated by Locust.
- Burst patterns: 100, 500, 1,000 req/sec for 5-minute intervals, interleaved with 2 min idle.
- Metrics:
  - Latency: p50, p95, p99 end-to-end
  - Throughput: successful req/sec
  - Cost: AWS billing data for function invocations and EC2 usage.

International Journal of Computer Technology and Electronics Communication (IJCTEC)

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 8, Issue 1, January- February 2025 ||

DOI: 10.15680/IJCTECE.2025.0801010

Lambda provisioned concurrency set to 200 where noted. Kubernetes HPA scaled between 2–20 pods (CPU threshold 70 %).

## IV. RESULTS AND DISCUSSION

We have evaluated the solution based on below parameters.

TABLE 1: LATENCY AND THROUGHPUT UNDER BURSTS

| MODE | P99 LATENCY (MS) | MAX THROUGHPUT (REQ/SEC) | MODE |
|---|---|---|---|
| LAMBDA (COLD-START) | 300 ± 50 | 1,200 | LAMBDA (COLD-START) |
| LAMBDA (WARM) | 80 ± 10 | 1,200 | LAMBDA (WARM) |
| KUBERNETES (STEADY) | 50 ± 5 | 800 | KUBERNETES (STEADY) |
| KUBERNETES (SCALE-OUT) | 65 ± 8 | 900 | KUBERNETES (SCALE-OUT) |

TABLE 2: COST PER 1M REQUESTS

| MODE | COMPUTE COST ($) | INFRASTRUCTURE COST ($)† | TOTAL ($) |
|---|---|---|---|
| SERVERLESS (ON-DEMAND) | 11.2 | N/A | 11.2 |
| SERVERLESS (PROVISIONED) | 18.5 | N/A | 18.5 |
| CONTAINERS (EKS + EC2) | 7.3 | 1.4 | 8.7 |

**Latency**
Cold starts yield high tail latency. Provisioned concurrency reduces p99 to ~100 ms.

**Throughput**
Lambda scales instantly to parallel invocations. Kubernetes HPA reacts in ~30 s, limiting peak throughput.

**Cost**
Containers are 22% cheaper at high volumes. Serverless is cost-effective at low/moderate bursts but expensive when maintaining concurrency.

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 8, Issue 1, January- February 2025 ||

DOI: 10.15680/IJCTECE.2025.0801010

## V. CONCLUSION

Serverless workloads deliver unmatched elasticity and peak throughput for bursty telecom traffic but suffer from cold start tail latency and higher cost at sustained volumes. Containerized microservices offer lower steady-state latency and cost but require scaling buffer to absorb spikes. Telecom operators should adopt a hybrid model of serverless for unpredictable bursts and containers for predictable baseline load.

## VI. LIMITATIONS

Platform specificity is a limitation, as the evaluation focuses on AWS Lambda and EKS, and other cloud providers or platforms may behave differently. Differences in function runtimes, networking, and autoscaling policies can affect generalizability. The traffic model is also simplified, using a basic signaling emulator rather than real-world traffic. Real traffic patterns may include long-running sessions and more complex behaviors that impact performance.
This simplification may understate latency and resource variability. Cost factors are another limitation, as the analysis excludes data-transfer and monitoring costs. These costs can be significant, especially in distributed or multi-cloud deployments. Ignoring them may lead to underestimation of total operational expenses. A more comprehensive evaluation should include realistic traffic and full cost accounting. Future work should validate findings across platforms and with production workloads.

## VII. FUTURE WORK

Hybrid orchestration enables seamless fan-out between containers and functions based on real-time metrics to optimize resource use and responsiveness. This approach allows workloads to dynamically shift to the best execution model depending on demand and latency requirements. Cold start mitigation is another key area, where microVMs like AWS Firecracker or snapshot-resume techniques can reduce startup delay. These methods aim to deliver near-instant function activation for latency-sensitive applications. Multi-cloud comparison can further broaden applicability by extending evaluation to Azure Functions, Google Cloud Run, and Kubernetes across cloud providers . This enables cross-platform insights into performance, cost, and operational trade-offs. Autoscaler optimization is also essential, with ML-driven scaling policies improving responsiveness and stability. Machine learning can predict workload spikes and trigger faster Kubernetes reactions.

## REFERENCES

1. Jonas, E. et al. (2019). Cloud Programming Simplified: A Berkeley View on Serverless Computing. USENIX CACM, 24(2), 24–31.
2. Wang, L., & Patel, K. (2021). Understanding Cold Starts in Serverless Function Platforms. IEEE Cloud, 8(1), 15–24.
3. Sharma, N., Gupta, R., & Singh, A. (2022). Serverless vs. Containerized Deployment for IoT Workloads. ACM IoT Journal, 5(3), 45–58.
4. Liu, Y., & Smith, J. (2023). Kubernetes Auto-scaling for 5G Network Functions. IEEE Transactions on Network and Service Management, 20(4), 102–114.
5. Chen, P., & Gupta, S. (2024). Dynamic Workload Offloading in Edge-Cloud Environments. ACM Symposium on Edge Computing, 50–63.
6. AWS (2023). Provisioned Concurrency for AWS Lambda. Amazon Web Services Whitepaper, 2023.