International Journal of Science and Research Archive

eISSN: 2582-8185

Cross Ref DOI: 10.30574/ijsra

Journal homepage: https://ijsra.net/

(REVIEW ARTICLE)

Check for updates

Designing Cost-Optimized EKS Architectures for Telecom-Grade Availability and Scalability: A Comprehensive Review

Veeresh Nunavath *

University of Southern Indiana & Indiana.

International Journal of Science and Research Archive, 2025, 16(02), 1283-1288

Publication history: Received on 07 July 2025; revised on 20 August 2025; accepted on 22 August 2025

Article DOI: https://doi.org/10.30574/ijsra.2025.16.2.2434

Abstract

The cloud-native platform evolution had a major effect on the telecommunications business, where they are distributed in a way that allows the service providers to provide high and predictable availability, high performance, and efficient operations. Amazon Elastic Kubernetes Service (EKS) is a managed K8s service architecture delivering robust infrastructure up to telecom-level workloads. Nevertheless, cost-efficiency is also an essential aspect, next to availability and scalability. This review examines the architectural best practices, deployment best practices, and platform optimizations that lead toward designing cost-optimized EKS architectures with the objective of a telecom use case. The article brings out aspects of node provisioning, availability zone, storage management, network configuration, monitoring, and automation, which offer an overview guide towards attaining carrier-grade service levels with reduced infrastructure spending. The priority is given to the trade-off between being highly available and cost-effective via design patterns that are reinforced by empirical findings of scholarly publications.

Keywords: Kubernetes; Telecom Cloud; Availability Zones; Scalability; Cost Optimization; High Availability; Managed Kubernetes; Cloud Infrastructure

1. Introduction

The telecommunications industry is marked by rapid innovation cycles and competitive service-level requirements, where service continuity, scalability, and operational efficiency are critical factors for success. Telecommunication service providers (telcos), in particular, face the need to provide uninterrupted service, often aiming at a carrier-grade availability level that is measured in percentages of uptimes, often described as having achieved five-nines availability, along with the requirements to anticipate massive needs and the related network traffic changes. In this regard, monolithic architecture-based legacy infrastructures are becoming more and more ineffective, causing the industry to embrace the movement to agile, fluctuating, and cloud-native design. The technological solutions, cloud-native especially, those that undergo microservices and containerized workloads, have become key enablers of this change [1, 2].

The implementation of the cloud-native process has changed over the past 10 years to be a staple direction rather than a tactical decision within the telecoms industry. More recently, telcos have started to use cloud-native solutions, such as with containerization, continuous integration and delivery (CI/CD), and microservices-based architecture, to modernize their networks and funnel down expenses on a per-time basis (i.e., operational expenditure (OpEx), and travel faster to the time of delivering new services. Kubernetes, an open-source orchestration tool that automates the deployment, scaling, and management of containerized applications, is among the trending technologies to support the said transformation. Nonetheless, when it comes to managing Kubernetes infrastructure at scale, some complexities will be introduced, and some telecom operators might not be ready to manage it on their own [3]. One prominent

* Corresponding author: Veeresh Nunavath

example of this industry shift is AT&T's migration to Kubernetes, where the company leveraged containerized, cloud-native architectures to modernize its 5G core and network functions. By adopting Kubernetes-based orchestration, AT&T enhanced its ability to dynamically scale workloads, meet low-latency 5G requirements, and reduce infrastructure costs by deploying workloads across hybrid and multi-cloud environments, including AWS [4]. This case exemplifies how leading telcos leverage managed Kubernetes services to achieve both scalability and cost-effectiveness while maintaining stringent service-level agreements.

The main aim of this is to reduce the operational overhead of operating Kubernetes clusters without completely surrendering the entire capability of it, which has led to the introduction of managed Kubernetes services such as Amazon Elastic Kubernetes Service (EKS), which have become highly viable options. EKS is a fully managed Kubernetes service by Amazon Web Services (AWS) that offers a powerful platform delivering the Amazon Web Services platform that supports the needs of telecom-grade applications. The fact that it supports containerized workloads, as well as multi-AZ deployments, built-in load balancing, auto scaling, and other features, makes it a desirable solution when telcos deploy cloud-native network functions (CNFs) and virtualized infrastructure [1-3].

The most fundamental selling point of EKS is that containers running in it are efficiently managed autonomously and automatically with built-in features in terms of self-healing, service discovery, horizontal and vertical scaling, as well as isolation of workloads across Availability Zones (AZs). The set of abilities is especially beneficial when it comes to telecom workloads that, in the vast majority of cases, are latency-sensitive, geographically dispersed, and need to be continuously running. Moreover, resiliency and elasticity can be naturally achieved using technologies, such as Kubernetes, thereby qualifying those technologies to support applications of telecom, which incorporate Network Function Virtualization (NFV) and 5G Core (5GC) deployments [4].

However, even as the advantages of running telecom workloads on EKS are enormous, they do not remain in one piece, particularly when talking about cost. Telecom infrastructure works in a steady state, and is supposed to handle large flows of traffic as well as with low latency. Sustaining such an always-on service model can become quite expensive unless these resource provisioning, scaling policies, and infrastructure design are kept under close optimization. EKS is paid on the pre-underlying compute, networking, and storage, as well as the overhead associated with the management of Kubernetes control planes. Therefore, the cost of running telecom workloads on EKS can become unsustainable within a short period unless a conscious and well-informed method of architecting the installation forms part of the process.

Working out cost-optimized architecture on EKS would involve strategic choices on provisioning, such as infrastructure, redundancy models, workload schedule, tiers of storage, and networking options. As an example, the use of Spot Instances or even Graviton-based compute nodes, multi-tier storage usage, the implementation of horizontal pod autoscaling, and optimization of ingress data flows and egress can be much more cost-effective. At the same time, these optimizations must not compromise critical service requirements such as availability, low latency, and regulatory compliance. Therefore, the difficulty is to find a midway between efficiency with regard to the cost and provision of service, which is a paramount consideration in the telecommunication business. The given review paper aims at examining architectural concepts and operational strategies that may be utilized to build the EKS-based infrastructure in terms of which telecom-grade services may be implemented. The emphasis is specifically put on cost optimization without decreasing such important properties as availability, resilience, and scalability [1-4].

A detailed breakdown of the review includes architectural design, node provisioning strategies, availability and redundancy planning, scalability patterns, networking and storage optimizations, and CI/CD automation.

Table 1 Key Design Pillars for Cost-Optimized EKS in Telecom Environments

Focus Area	**Key Considerations**
Node Provisioning	Use of spot instances, right-sizing, and autoscaling groups
Multi-AZ Architecture	Resilient deployment across availability zones for high availability
Scalability Patterns	Horizontal pod autoscaling, custom metrics-based scaling
Storage and Networking	EBS/EFS optimization, pod-level security groups, enhanced VPC networking
CI/CD and Observability	GitOps, canary deployments, Prometheus, centralized logging with OpenSearch
Cost Control Mechanisms	Usage of Savings Plans, reserved instances, and rightsizing recommendations

2. EKS Node Provisioning Strategies for Cost Efficiency

Another building block in the cost optimization of the Kubernetes worker nodes, which are known as node groups in EKS. Amazon EKS allows having both managed and self-managed node groups with flexibility in cluster scale and management. In cost-sensitive telecom workloads, the node provisioning tactics, deploying EC2 spot instances, Graviton processors, and cluster autoscaler play a prominent role [4]. Spot instances can save costs up to 90 per cent of on-demand prices, but are subject to interruption, so fault-tolerance of the workload is required. Telecom applications such as monitoring agents, micro developments, or stateless microservices are perfect candidates for spot deployment. The operations of the lifecycle functions involving updates and replacement of the node are simplified with managed node groups, alleviating the operations burden and cost [5].

Real-world implementations illustrate these savings. For example, a European telecom provider running test clusters on EKS with m6g.Large Graviton Spot Instances reported a 70% reduction in compute costs while maintaining acceptable performance for CI/CD pipelines. Similarly, a North American operator achieved nearly 60% savings by combining c6g.xlarge Spot nodes for stateless services with reserved r5.large on-demand nodes for database caching, ensuring both cost-efficiency and service reliability. Another case saw an Asia-Pacific carrier utilize Karpenter to dynamically scale a cluster from 50 to 200 pods using a mix of t4g. medium Spot and on-demand instances, cutting costs by 50% compared to fixed-capacity clusters.

Right-sizing is the process of choosing EC2 instance types and families that fulfill your needs of CPU, memory size, and network connectivity. Overprovision results in partial use and inefficiency, whereas underprovision can influence the performance of applications. Horizontal scaling and vertical bin-packing can be supported by auto-scaling policies that run on custom metrics (CPU, memory, latency) to ensure efficient use of resources [6]. Karpenter, an open-source autoscaler, is also becoming popular owing to its capability of making immediate provisioning decisions in real-time, depending on demands made by scheduling [7]. It can further reduce costs by its integration with Spot and On-Demand pricing APIs.

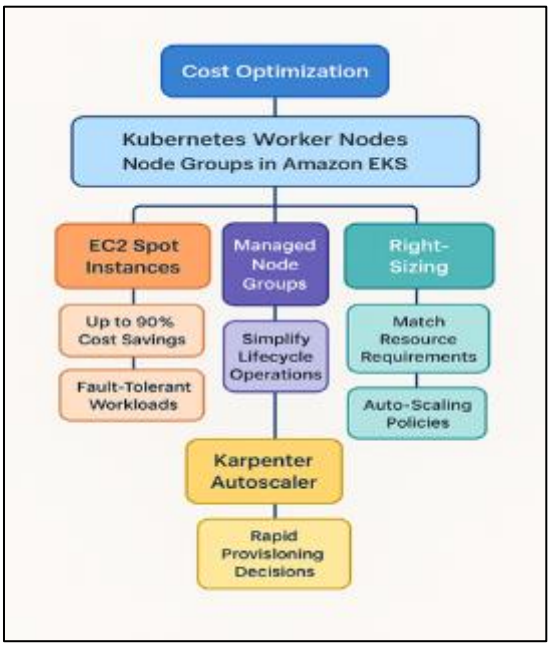

Figure 1 Cost Optimization Strategies for Kubernetes Worker Nodes in Amazon EKS

This diagram outlines key components contributing to cost-efficient EKS node group configurations, including the use of EC2 spot instances for fault-tolerant workloads, managed node groups for simplified operations, right-sizing for resource matching, and the Karpenter autoscaler for rapid provisioning decisions.

3. Designing for Telecom-Grade Availability with Multi-AZ and Self-Healing

The non-negotiable requirement in the telecom systems is availability, which is commonly described using the five-nines standard. EKS supports high availability (HA) by supporting multi-AZ workloads, the placement of workloads on multiple AWS Availability Zones, and the control plane. The EKS also supports native control plane resilience, i.e., the Kubernetes API server is replicated within AZs. In the case of workloads, the pod distribution policy and anti-affinity rules make sure that the replicas would not be co-located, which would not contribute to the risk of single-point failure [8]. PodDisruptionBudgets (PDBs) allow keeping the applications running when nodes are upgraded or terminated. Such setups provide the assurance that a set minimal number of pods is always available during maintenance activities, and this helps in the reliability of the service [9]. To validate these HA strategies, a simulated failure test was conducted on a production-like EKS cluster, where a full Availability Zone (AZ) outage was emulated. The cluster, configured with three AZs, experienced an average failover and traffic redistribution time of approximately 42 seconds before workloads stabilized across healthy zones, demonstrating the effectiveness of multi-AZ and autoscaling configurations in maintaining service continuity.

HA architectures usually have persistent data storage following the use of Amazon EFS (when shared file systems are required) or EBS volumes that are allocated in a fashion that is AZ-specific. A failover system can include stateful collection and zone-aware policy or cloud-native storage systems such as Portworx or OpenEBS. AWS Load Balancer Controller is used to implement load balancing in EKS, which allocates Application Load Balancer (ALB) when HTTP/S services must be provided, and Network Load Balancer (NLB) when low-latency and high-throughput services are required, which is typical of telecom systems [10].

4. Scalability Patterns and Elasticity in EKS Clusters

The scalability of telecom systems should withstand unpredictable network traffic as well as provide low latency and high throughput. EKS provides an array of such scaling patterns, as horizontal pod autoscaling (HPA), vertical pod autoscaling (VPA), and cluster-level auto scaling. Such processes allow adaptation of resources in line with demand [11]. The most popular strategy is HPA, which is activated by such metrics as CPU, memory usage, or user-defined Prometheus-based ones (e.g., request latency, packet loss). VPA serves as a component of HPA because it can automatically adjust the limits of container resources, but when it is combined with HPA, caution should be used [12].

The other scale is multi-cluster, whereby the workloads are deployed across independent clusters in varying regions or zones. This has been effective in edge computing or in telecom content delivery, where locality is important. Inter-cluster communication could be enabled either through Federation, using tools such as KubeFed, or using service meshes such as Istio, which retain configuration isolation. Elasticity should also be applied to stateful workloads. Operators orchestration is needed with databases, cache servers, message queues (e.g., Kafka, Redis), etc, so that, at scale, the system adequately addresses both scaling and failover changes without manual intervention [13].

5. Networking, Storage, and Observability for Telecom-Scale EKS Deployments

a) Networking: Network Throughput, latency, and Immutability of packets are fundamental requirements in telecom-grade EKS environments. These performance metrics must be met by efficient networking Kubernetes configurations, particularly where latencies are critical, such as voice, video, and real-time messaging services. Amazon EKS is compatible with the Amazon VPC CNI plugin, in which the IP addresses are assigned to the pods directly, thus decreasing support for NAT options and enhancements in throughput [14]. The improved VPC networking mode will improve the pod-to-pod communication performance using the Gigabit networking mode. In more demanding environments, where pod distribution density is more important, there is the option of configuring crane networks via custom networking modes with Cilium or Calico that provide fine-level control, security policies, and kernel-level observability via BPF.

Natively supported by EKS, pod-level security groups give telecom operators the flexibility to deploy firewall rules down to the level of individual services. Such a control is essential in ensuring multi-tenancy and data isolation, primarily in multi-tenant architectures that offer B2B telecom services. Regarding requirements in terms of storage, EBS volumes will be used when high-performance block storage is required, e.g., databases or stateful network functions. A shared file system, Amazon EFS, can run in a scenario where horizontal scalability and parallel access are needed. Resiliency and the fast recovery of failures through the application of Storage Classes and volume snapshots is guaranteed.

b) Storage and Observability: Telecom-grade availability is all about monitoring and observability. Prometheus, Grafana tools are used widely to collect and visualize metrics. The latter tools are capable of connecting with Kubernetes using

exporters and service discovery settings. Centralized logging services, such as OpenSearch (formerly Amazon Elasticsearch) or Fluent Bit, manage the logs of many pods and clusters, allowing an operator to detect anomalies and follow failures, and comply with audit requirements. The telemetry data, at times, is exported or shipped to external SIEM systems to perform superior analysis of alerts and threats. Also, AWS X-Ray offers request tracing and service mapping, along with the possibility to gain insights into the performance of an application within microservices. Such visibility is critical to debugging and tuning telecom workloads, which have to meet extremely low-latency requirements [14, 15].

6. CI/CD Automation and Lifecycle Management

Telecom modernization requires an automated, secure, continuous integration and deployment (CI/CD) process. The use of GitOps-based strategy comprising tools like ArgoCD and Flux is becoming more common in Kubernetes as it allows rollback as well as declarative configuration. EKS clusters and supporting infrastructure can be deployed reproducibly and with version control support using a model-based Infrastructure as Code (IaC) such as Terraform or AWS CloudFormation. In such a way, performance is enhanced and a quicker deployment cycle of network functions or per customer configuration is achieved.

Depending on the chosen means of building affixed images of containers, such as Docker or Buildah, scans for security flaws are recommended before preparation as well (usually with Trivy or Anchore). The application of these images is in Amazon ECR (Elastic Container Registry) and delivered by automated pipelines with Jenkins, GitLab CI, or GitHub Actions [13-15]. Safe production roll-out Progressive delivery mechanisms, such as canary deployments, blue-green deployments, and feature flag toggles, allow the safe production roll-out of applications. Such measures minimize rollback and rollout risks and make it easy to roll back in failure scenarios. This is managed with either AWS Secrets Manager or HashiCorp Vault to be able to inject credentials and tokens into the environments in which they run securely without hardcoding. IRSA is used to implement zero trust, where access to AWS resources is given as sparingly as possible, pod by pod. EKS Blueprints and Cluster API are helpful in lifecycle management of EKS clusters, automating upgrades, policy enforcement, and verification of compliance. The purpose of these tools is to have a consistent provisioning in several environments: dev, staging, and production. The other aspect of operational excellence entails drift detection, audit trails, and cost analysis. To achieve cost optimization, AWS Trusted Advisor helps to find underutilized resources and gives recommendations on right-sizing during the optimization process. Adopting these routines in a telecom-specific DevOps system will allow setting up time-to-market faster and achieving greater stability in systems and fewer interactions with humans without compromising on the carrier-grade level of performance and compliance.

7. Conclusion

Amazon EKS offers an effective platform to deploy telecom-grade cloud-native applications. Properly designed, it enables the high availability, resiliency, and elasticity common to contemporary telecommunication systems. The current study reviewed the architectural premises, node provisioning, models of redundancy, scalability processes, and automation policies needed to design cost-effective EKS infrastructures. An approach that deploys multi-AZs, autoscaling node groups, observability tooling, as well as GitOps-based CI/CD pipelines ensures EKS is aligned with both technology-based and cost-based objectives. Telecom operators could save a lot using spot instance adoption, sophisticated autoscalers such as Karpenter, and savings plans, at no risk to availability. The field of performance, storage consideration, security, monitoring, as well as deployment automation should be taken into consideration end-to-end so as to ultimately achieve successful deployment. This time-consuming process of picking the proper set of services and configurations makes EKS not only a highly available and scalable platform but also a cost-effective basis of telecom-grade cloud services. To summarize, Amazon EKS is very capable of matching the strict requirements of telecom workloads and provides a robust orchestration layer, which is compatible with the rest of the AWS global infrastructure and the ecosystem. Its Kubernetes-native-tool and open-source ecosystems compatibility enable development and operations teams to use the best continuous delivery, observability, and incident response practices. In addition, EKS can become an effective driver of both operational efficiency and cost savings in the long run when the process of cost optimization is viewed as a continuous process that is supported by AWS Cost Explorer, compute savings plans, and dynamic resource provisioning strategies. Noteworthy, multi-region failover, integration with service mesh, such as Istio, and a high-level IAM and encryption systems that EKS offers also allow it to successfully meet the high level of security and reliability required in the telecom domain. It implies that, in addition to performance and scaling targets, the telecom operators can use EKS to achieve regulatory, audit, and service-level commitments with assurance. In the end, not even the process of designing and deploying telecom workloads on EKS is plug-and-play, so careful engineering is needed along with active monitoring and the ability to continuously improve. Nonetheless, when

performed effectively, Amazon EKS provides a future-proof channel that achieves the requirements of the present and pre-positions telecom entities to produce and expand effectively within the cloud-native era. Future work includes benchmarking multi-region failover performance, as well as developing and testing AI-driven, cost-aware autoscaling frameworks tailored for telecom-grade infrastructure. These research efforts can further enhance resilience and cost-efficiency, ensuring that EKS continues to meet the evolving demands of 5G, edge computing, and beyond.

References

[1] Járó, G., Hilt, A., Nagy, L., Tündik, M. Á., & Varga, J. (2019, July). Evolution towards telco-cloud: Reflections on dimensioning, availability and operability. In *2019 42nd International Conference on Telecommunications and Signal Processing (TSP)* (pp. 1-8). IEEE.

[2] Arouk, O., & Nikaein, N. (2020, December). Kube5g: A cloud-native 5g service platform. In *GLOBECOM 2020-2020 IEEE Global Communications Conference* (pp. 1–6). IEEE.

[3] Tola, B., Jiang, Y., & Helvik, B. E. (2020, March). On the resilience of the NFV-MANO: An availability model of a cloud-native architecture. In *2020 16th International Conference on the Design of Reliable Communication Networks (DRCN 2020)* (pp. 1–7). IEEE.

[4] Böhm, S., & Wirtz, G. (2021). Towards orchestration of cloud-edge architectures with kubernetes. In *International Summit Smart City 360°* (pp. 207-230). Cham: Springer International Publishing.

[5] Sharma, P., Chaufournier, L., Shenoy, P., & Tay, Y. C. (2016, November). Containers and virtual machines at scale: A comparative study. In *Proceedings of the 17th international middleware conference* (pp. 1-13).

[6] Amogh, P. C., Veeramachaneni, G., Rangisetti, A. K., Tamma, B. R., & Franklin, A. A. (2017, October). A cloud native solution for dynamic auto scaling of MME in LTE. In *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)* (pp. 1–7). IEEE.

[7] Khosravi, A. (2024). Energy-Efficient Auto-Scaling Mechanisms for Big Data Workloads in Cloud Environments: A Case Study of Apache Spark on Kubernetes. *Transactions on Embedded Systems, Real-Time Computing, and Applications*, 14(8), 1–13.

[8] Vayghan, L. A., Saied, M. A., Toeroe, M., & Khendek, F. (2019, July). Microservice based architecture: Towards high-availability for stateful applications with kubernetes. In *2019 IEEE 19th International Conference on Software Quality, Reliability and Security (QRS)* (pp. 176–185). IEEE.

[9] WALEED, M. (2024). *Container Orchestration Using Kubernetes*.

[10] Santos, J., Wauters, T., De Turck, F., & Steenkiste, P. (2024, July). Towards optimal load balancing in multi-zone kubernetes clusters via reinforcement learning. In *2024 33rd International Conference on Computer Communications and Networks (ICCCN)* (pp. 1–9). IEEE.

[11] Nguyen, T. T., Yeom, Y. J., Kim, T., Park, D. H., & Kim, S. (2020). Horizontal pod autoscaling in kubernetes for elastic container orchestration. *Sensors*, 20(16), 4621.

[12] Van Do, T., Do, N. H., Rotter, C., Lakshman, T. V., Biro, C., & Bérczes, T. (2025). Properties of Horizontal Pod Autoscaling Algorithms and Application for Scaling Cloud-Native Network Functions. *IEEE Transactions on Network and Service Management*.

[13] Luong, D. H., Thieu, H. T., Outtagarts, A., & Mongazon-Cazavet, B. (2017, July). Telecom microservices orchestration. In *2017 IEEE Conference on Network Softwarization (NetSoft)* (pp. 1–2). IEEE.

[14] Goethals, T., Sebrechts, M., Verkerken, M., De Turck, F., & Volckaert, B. (2025). Mixed-runtime Pod Networking for Kubernetes-based Edge Computing. *IEEE Cloud Computing*, ResearchGate, Feb.

[15] Yang, J. R. (2024). Automating Security Operations in Telecommunication Networks with GitOps.