AKS or EKS, which cloud platform should I use for my Kubernetes cluster

KRIAT Yassine*, TAMI Anas*, BOUAM Adam*, EL-KHARRAZ Issam *, EDDAHBI Abderrahmane *,and TEABE Boris†
*École Nationale Supérieure d'Électrotechnique, d'Électronique, d'Informatique, d'Hydraulique et des Télécommunications, (ENSEEIHT), Toulouse INP, 31000 Toulouse, FRANCE
Email: {yassine.kriat, anas.tami, adam.bouam, issam.elkharraz, abderrahmane.eddahbi}@etu.inp-n7.fr
†École Nationale Supérieure d'Électrotechnique, d'Électronique, d'Informatique, d'Hydraulique et des Télécommunications, (ENSEEIHT), Toulouse INP, 31000 Toulouse, FRANCE
Email: boris.teabedjomgwe@enseeiht.fr

***Abstract*—The advent of containerization technologies, led by Kubernetes, has revolutionized the deployment and management of applications in the cloud, offering unprecedented scalability, portability, and resource efficiency. This project explores the deployment of the Google Online Boutique e-commerce application on two of the leading container management services in the cloud: Amazon Elastic Kubernetes Service (EKS) and Azure Kubernetes Service (AKS). The aim is to compare these platforms in terms of performance, reliability, and cost in the context of an e-commerce application. By deploying the application, extracting metrics and execution traces, and developing a simulator based on this data, we aim to thoroughly evaluate each service. Additionally, this study includes the implementation of real-world usage scenarios to simulate actual operational conditions, enhancing the relevance of our findings. This work aims to provide a detailed comparative analysis, offering valuable insights into the advantages and limitations of EKS and AKS for companies considering using Kubernetes for their cloud applications, ultimately guiding strategic decisions in cloud infrastructure selection.**

***Keywords*—Container Orchestrators (EKS, AKS), Cloud Platform Evaluation, Simulator, Performance Comparison, Cost Analysis.**

I. Introduction

Navigating the cloud computing terrain demands strategic decisions, none more pivotal than selecting the right orchestration service for containerized applications. In this report, we delve into a comparative analysis of Amazon Elastic Kubernetes Service (EKS) and Azure Kubernetes Service (AKS), two prominent players in the field. Our exploration spans critical performance metrics such as availability, scalability, latency, and resource utilization. Through meticulous load testing, real-world simulations, and an in-depth cost examination, we aim to unravel the intricacies of EKS and AKS. By distilling nuanced insights, we empower decision-makers to chart a course through the dynamic landscape of cloud-based container orchestration. Join us on this journey to unveil the strengths, weaknesses, and thoughtful considerations that shape the EKS vs. AKS landscape.

As we dissect the performance and cost metrics, we highlight that the apparent superiority of AKS over EKS prompts a closer examination. Beyond the numbers, the optimal choice hinges on specific project needs and a judicious balance of features and services.

II. Context

A. Amazon Elastic Kubernetes Service (EKS)

In June 2018, Amazon Elastic Kubernetes Service (EKS) marked a significant milestone, offering a managed Kubernetes service on AWS to the public. Positioned as a foundation for container orchestration, EKS seamlessly integrates services, applications, and protocols within the Kubernetes environment. Boasting advanced scalability features and effortless connectivity with third-party tools for logs and performance analytics, EKS presents an all-encompassing solution. Whether for migrating workloads across multiple clouds or experimenting with Kubernetes within an existing AWS architecture, EKS stands as a robust choice. [1]

B. Azure Kubernetes Service (AKS)

Introduced in 2018 by Microsoft, Azure Kubernetes Service (AKS) is a comprehensively managed Kubernetes service aimed at streamlining the deployment and management of applications in containers. Offering versatility, AKS operates both on-site and on Azure Public Cloud, catering to clients with critical applications. Notably, AKS provides seamless integration with Microsoft's tools, including Visual Studio and Active Directory, as well as other SaaS services. Particularly advantageous for businesses with established agreements with Microsoft, AKS proves to be a tailored solution. [3]

C. Project's Objectives

- **Deployment of the application:** Establish the required setup on EKS and AKS for the Google Online Boutique application, ensuring a secure, scalable, and test-ready environment.
- **Gather metrics and execution logs:** Employ tools for monitoring and logging to gather insights on the behavior of the application and the utilization of resources across both cloud platforms.
- **Development of a simulator:** Construct a tool capable of emulating real-world usage scenarios using the collected

execution traces, allowing for the evaluation of cluster performance across diverse conditions.
- **Comparative analysis:** Analyze and compile the gathered information to provide a detailed overview of the advantages and limitations of EKS and AKS, with an emphasis on their performance, reliability, and cost implications in relation to the tested application.

D. *Performance Metrics and Recovery Tools*

- Availability and Reliability (uptime, error rate.)
- Scalability (Ability to respond at scale.)
- Latency and Response Time
- Resource utilization efficiency (CPU, memory, storage)

E. *Cost Metrics and Recovery Tools [5]*

- Pricing model (Usage costs, tariffs.)
- Operating costs (Management and maintenance costs.)
- Resource costs (Price of instances, storage, data transfer.)
- Tools: Azure [7] and AWS [6] cost calculators.

III. Problem statement

Despite the swift evolution and widespread adoption of containerization technologies, businesses, especially those in the e-commerce sector, grapple with significant challenges when selecting the optimal cloud service platform for deploying and managing applications. The decision-making process between Amazon Elastic Kubernetes Service (EKS) and Azure Kubernetes Service (AKS) becomes particularly intricate, given the complexities in comparing performance, reliability, cost, and ease of management across these platforms.

While AWS and Azure each offer compelling features for container management through EKS and AKS, the lack of a comprehensive, objective comparison hinders businesses from making informed decisions. This challenge is exacerbated by various factors such as scalability, availability, operational costs, and integration capabilities with existing cloud infrastructures. Additionally, the absence of clear, real-world usage scenario simulations poses a barrier to predicting the performance of each service under specific e-commerce operational conditions.

The core objective of this project is to address these challenges systematically. By deploying the Google Online Boutique application on both EKS and AKS, extracting and analyzing performance metrics, and simulating real-life usage scenarios, the project aims to provide a meticulous comparative analysis of each service's strengths and weaknesses. Through this analysis, businesses in the e-commerce sector will be equipped with valuable insights to guide their decision-making process in selecting the most suitable cloud service platform for their specific needs.

IV. Experimentation

A. *Environment Setup*

Cluster Creation on Amazon Elastic Kubernetes Service (EKS): EKS clusters were efficiently created using the eksctl command-line tool. The streamlined process involved a direct command without the need for detailed configuration files, ensuring a rapid setup of worker nodes and cluster settings. [1]

Cluster Creation on Azure Kubernetes Service (AKS): The AKS cluster was successfully created through the Azure portal, configuring node pools, Kubernetes version, and networking settings. Azure Console was used to ensure proper integration with the Azure cloud environment. [3]

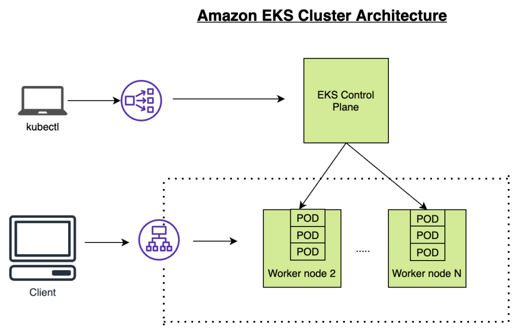


Fig. 1. EKS Cluster Architecture

B. *Application Deployment [4]*

Google Online Boutique Application Deployment on EKS: The Google Online Boutique application was successfully deployed on the EKS cluster using Kubernetes manifests tailored for EKS specifications. This deployment ensured compatibility with the eksctl-created clusters.

Google Online Boutique Application Deployment on AKS: Similar Kubernetes manifests were successfully employed for AKS, ensuring the application's successful deployment on clusters created through the Azure Console.

C. *Monitoring Tools Deployment*

Prometheus and Grafana Deployment on EKS: Prometheus and Grafana were successfully installed on the EKS cluster using Helm charts through the AWS CLI. Helm simplified the deployment process and ensured consistency across different environments.

Prometheus and Grafana Deployment on AKS: Parallelly, Helm charts were successfully used to deploy Prometheus and Grafana on the AKS cluster through the Azure CloudShell. This approach ensured uniformity in monitoring tool deployment across both cloud platforms.

D. *Load Testing Setup*

k6 Installation: k6, the load testing tool, was successfully installed in a dedicated environment, enabling load testing for both EKS and AKS clusters. The installation process involved downloading the tool and configuring test scripts.

Load Testing Scenarios: Three distinct load testing stages (low, medium, and high) were successfully executed using k6. These scenarios simulated various user behaviors and transaction volumes, providing insights into the scalability and performance of both EKS and AKS under different load conditions.

E. Data Collection and Analysis

Performance Metrics Collection: Prometheus and Grafana, installed using Helm, continuously and successfully collected performance metrics from both EKS and AKS clusters during the load testing scenarios.

Load Testing Results Analysis: k6 results were thoroughly and successfully analyzed to assess the impact on response times, resource utilization, and overall system performance. Comparative insights between EKS and AKS were derived, with a focus on how each platform handles varying levels of workload.

V. Results and Comparison

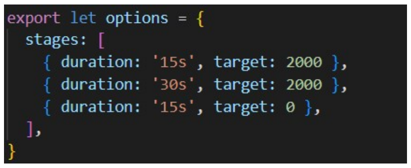
A. Performance annalysis and comparison

To comprehensively evaluate the performance of the Google Online Boutique application on both Amazon Elastic Kubernetes Service (EKS) and Azure Kubernetes Service (AKS), a meticulous approach to load testing was employed. The process involved the execution of multiple load testing scenarios, each designed to simulate varying user loads and operational conditions.

Load Testing Scenarios:

- *Multiple Scenarios:* Several load testing scenarios were crafted to mimic real-world usage patterns and stress the systems under different conditions.
- *Variety of Load Levels:* Load testing was conducted at various levels, ranging from nominal to high user loads, to assess how well the systems could adapt and scale.
- *Diverse Operational Conditions:* The scenarios covered a spectrum of operational conditions, including scenarios of increasing load, sustained load, and decreasing load.

k6 Load Testing Stages:
The load on the system was orchestrated using the k6 tool, employing a staged approach that offered a nuanced understanding of system behavior. One notable example includes a scenario with a very high load, employing 2000 virtual users over specific durations:



```
export let options = {
  stages: [
    { duration: '15s', target: 2000 },
    { duration: '30s', target: 2000 },
    { duration: '15s', target: 0 },
  ],
}
```

Fig. 2. K6 Load Stages

This setup allowed for a gradual increase in load, sustained high load for 30 seconds, and a subsequent decrease, providing valuable insights into how the systems performed under escalating, sustained, and diminishing loads. These load testing stages were crucial for assessing scalability, response times, and overall system performance.

Metric Measurements and Averaging
For each load testing scenario, a comprehensive set of performance metrics was measured and recorded. These metrics included, but were not limited to, uptime, error rates, throughput, response times, latency percentiles, CPU and memory utilization, and storage throughput.
To ensure robustness and reliability in the results, multiple measurements were taken during each scenario, and the averages of these metrics were calculated. This approach provided a holistic understanding of the performance characteristics of both EKS and AKS under diverse load conditions.
By adopting this meticulous approach to load testing and results analysis, we aimed to provide a nuanced and comprehensive comparison of the performance, reliability, and scalability of EKS and AKS in the context of the Google Online Boutique application.

TABLE I
Performance Comparison between EKS and AKS

Perf. Metrics	**Metric**	**Low**	**Med.**	**High**	**V. High**
Availability & Reliability	Uptime (%)	eks: 100 aks: 100	eks: 100 aks: 100	eks: 100 aks: 100	eks: 100 aks: 100
	Error rate (%)	eks: 0 aks: 0	eks: 0 aks: 0	eks: 13 aks: 50	eks: 90 aks: 73
Scalability	Throughput (req/s)	eks: 55 aks: 55	eks: 57 aks: 60	eks: 26 aks: 55	eks: 40 aks: 50
Latency & Response Time	Resp. Time (s)	eks: 5.8 aks: 1.94	eks: 11 aks: 10.31	eks: 17 aks: 17.79	eks: 25 aks: 18.2
	Latency (s)	eks: 2.5 aks: 1.3	eks: 12.5 aks: 12	eks: 19 aks: 20	eks: 25 aks: 22
Resource Utilisation	CPU (%)	eks: 55 aks: 45	eks: 63 aks: 56	eks: 42 aks: 52	eks: 28 aks: 50
	Memory Usage (%)	eks: 45 aks: 28	eks: 56 aks: 28	eks: 52 aks: 29	eks: 50 aks: 29
	Storage Thr. (kB/s)	eks: 1.5 aks: 1.5	eks: 1.8 aks: 1.5	eks: 1.7 aks: 2	eks: 2.1 aks: 2.5

the performance comparison between Amazon Elastic Kubernetes Service (EKS) and Azure Kubernetes Service (AKS) reveals notable distinctions across several critical dimensions. Both platforms exhibit robust availability and reliability, maintaining 100% uptime, though AKS experiences a marginally higher error rate under high load. In terms of scalability, EKS demonstrates more consistent throughput, while AKS shows a more significant decrease in throughput under heavy loads. Regarding latency and response time, EKS generally outperforms AKS, providing better responsiveness, especially evident in scenarios with very high loads. Resource utilization efficiency varies, with EKS showcasing higher CPU utilization under lighter loads, and AKS displaying better efficiency under heavier workloads. Memory usage is comparable, although AKS exhibits slightly higher usage during medium and high loads. Finally, storage throughput sees marginal improvements for both platforms with increasing load, and AKS demonstrates a slightly superior increase under very high load conditions.

B. Cost Simulation and Comparison

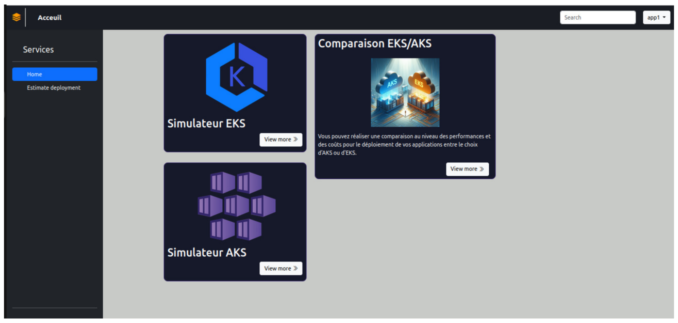
The comparison between the two services. Amazon Elastic Kubernetes Service (EKS), and Azure Kubernetes Service (AKS), aims to determine the most optimal choice for deploying an application, which is the primary objective of our project. This comparison is based on deploying the application on both Amazon and Azure platforms, as well as conducting performance tests by simulating various user loads.

The first part focused on planning, researching, and gathering information to establish the prerequisites for our project. This puts us in a good position to conduct an effective comparison between the two services.

However, it's challenging to reach direct results that can provide a clear judgment based solely on raw data and metrics. This is where the simulator comes in. It uses the information gathered in the first phase to simulate the costs and performance of deploying an application with both AKS and EKS. By offering a detailed comparison between the two, the simulator provides a clear understanding of each service's specific requirements. As a result, users can get a precise idea of what each service entails, helping them make an informed decision based on factors like cost, performance, or a combination of both, depending on their specific situation.

The simulator application was developed using Angular components with HTML, SCSS, Bootstrap, and TypeScript for designing responsive pages. Spring Boot with Java EE and JPA starters were utilized for the API, and MySQL and MongoDB were chosen for managing the databases.

.



Vous pouvez réaliser une comparaison au niveau des performances et des coûts pour le déploiement de vos applications entre le choix d'AKS ou d'EKS.

Fig. 3. Simulator

EKS Cost Simulation:

- Navigate to the EKS Simulator by clicking on "View More" on the homepage.
- On the calculation page, select the services you wish to estimate and enter relevant information.
- Add the selected services to the calculation and proceed to checkout.
- View the cost estimation results for EKS.

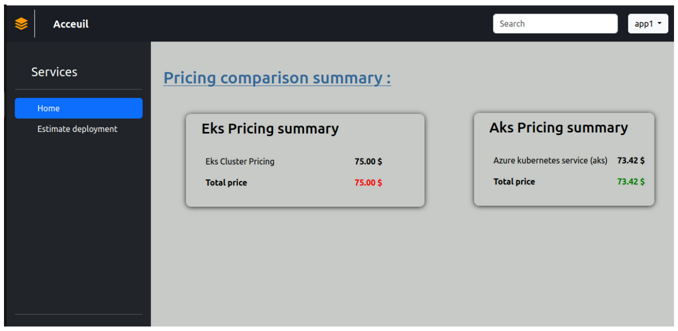
AKS Cost Simulation:

- After completing the EKS cost estimation, navigate to the AKS calculator by transitioning from the EKS results page.
- Alternatively, return to the homepage, select the "AKS Simulator" section, and follow the same process for entering information and estimating costs.
- View the cost estimation results for AKS.

Cost Comparison:

- After obtaining the cost estimations for both EKS and AKS, compare them on the cost comparison page.
- This can be achieved either through a dedicated button on the AKS results page or by selecting the "EKS/AKS Comparison" section on the homepage.
- The cost comparison provides insights into the financial aspects of deploying and managing applications on EKS and AKS, aiding in informed decision-making.

The web application simulator streamlines the estimation process, allowing users to input specific parameters for each platform and visualize the comparative costs effortlessly. This approach facilitates a practical and user-friendly means of assessing the economic considerations associated with choosing between EKS and AKS for Kubernetes-based deployments.



Pricing comparison summary :

Eks Pricing summary

Eks Cluster Pricing	75.00 $
Total price	75.00 $

Aks Pricing summary

Azure kubernetes service (aks)	73.42 $
Total price	73.42 $

Fig. 4. EkS & AKS Cost Estimation

In the realm of cost comparison, Kubernetes service pricing on a monthly basis reveals a subtle difference between Amazon Elastic Kubernetes Service (EKS) and Azure Kubernetes Service (AKS). EKS is priced at $75.00 for cluster usage, while AKS stands at $73.42, indicating a marginally lower cost of $1.58 for AKS. While cost is a crucial factor in decision-making, it's imperative to weigh this against the nuanced performance metrics, features, and services offered by each provider. The choice between EKS and AKS should align not just with budget considerations but also with the specific requirements and preferences of your project.

Performance Comparison:
After each test scenario with different user loads for the application deployed on both Amazon and Azure platforms, we gather the values representing the essential metrics upon which we based our performance comparison. Similarly, for the cost comparison, we provide the appropriate values for each separate load in the performance section.

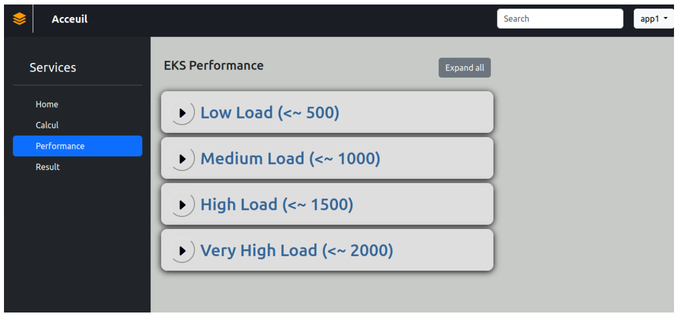


Fig. 5. EkS Performance Estimation Form Example

After recording the data, users can be redirected to the results section (for EKS and AKS services separately) to visualize the results in a table format, showing the percentage for each metric. Once the results for both services are obtained, we can compare them in terms of performance, either in table format or in a histogram format, as shown below.

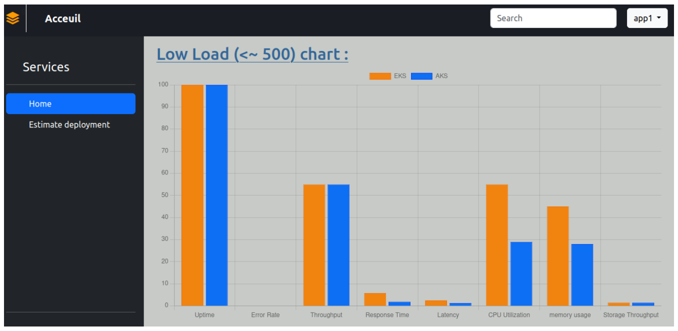


Fig. 6. EkS Performance Estimation Form Example

As depicted in the image above, the histogram illustrates a comparison between EKS and AKS (orange for EKS and blue for AKS), focusing on each metric to demonstrate the superiority of one service over the other across all levels, providing users with the flexibility to decide which service best suits their situation. However, from our perspective, relying on the research and simulation of the application we deployed, it is evident that Azure is the platform representing the least expensive and most performant Kubernetes service across the majority of the criteria studied above.

VI. Conclusion and Future Work

The performance and cost analysis of Amazon Elastic Kubernetes Service (EKS) and Azure Kubernetes Service (AKS) reveals intriguing findings. While AKS excels in certain aspects with slightly lower monthly costs, the decision-making process necessitates careful consideration of project-specific needs, preferences, and scalability requirements. Optimal outcomes depend on aligning the chosen Kubernetes service with unique project demands.

Additionally, investigating long-term operating facets, including maintenance costs, upgrades, security measures, and assessing ecosystems, community support, and integration ease, would offer valuable insights. Exploring hybrid and multi-cloud approaches with these services could further guide decision-making.

References

[1] Amazon Web Services, Inc. 2024. *What is Amazon Elastic Kubernetes Service?* Retrieved from https://docs.aws.amazon.com/eks/latest/userguide/what-is-eks.html
[2] Hightower, K., Burns, B., & Beda, J. 2019. *Kubernetes Up and Running.*
[3] Microsoft. 2024. *Azure Kubernetes Service (AKS).* Retrieved from https://learn.microsoft.com/en-us/azure/aks/
[4] Arundel, J., & Domingus, J. 2019. *Cloud Native DevOps with Kubernetes.*
[5] Storment, J.R., & Fuller, M. 2019. *Cloud FinOps: Collaborative, Real-Time Cloud Financial Management.*
[6] AWS Pricing Calculator, *https://calculator.aws/*
[7] Azure Pricing Calculator, *https://azure.microsoft.com/en-us/pricing/calculator/*