Meshery - The Service Mesh Management Plane Case Study

Krzysztof Kwiecień, Łukasz Sochacki, Szymon Sumara, Patryk Studziński
Group: Meshery-20

March 30, 2023

Contents

1 Introduction **1**

2 Theoretical background/technology stack **1**

3 Case study concept description **2**

4 Solution architecture **3**

5 Environment configuration description **3**

6 Installation method **3**

7 How to reproduce - step by step **3**
7.1 Infrastructure as Code approach 3

8 Demo deployment steps **3**
8.1 Configuration set-up 3
8.2 Data preparation 3
8.3 Execution procedure 3
8.4 Results presentation 3

9 Summary – conclusions **3**

1 Introduction

Meshery is an open-source service mesh management platform that simplifies the deployment and management of multiple service meshes in Kubernetes and other containerized environments. It provides built-in performance testing and observability tools, allowing users to monitor and optimize their application's performance across multiple meshes and clusters.

Meshery helps users achieve greater visibility, control, and scalability in their complex multi-mesh environment while also simplifying their operations and reducing the risk of service disruptions.

More information about the project can be found on the official Meshery website.

2 Theoretical background/technology stack

As the adoption of microservices architectures continues to grow, managing and monitoring these services becomes increasingly complex. Service meshes like Istio, Linkerd, and Consul provide a solution to this problem, but their configuration and management can be time-consuming and complex. This is particularly true when organizations use multiple cloud providers and platforms, which require different configurations and setups for service meshes.

Meshery simplifies service mesh management by providing a unified platform that supports multiple service mesh providers and cloud platforms. The platform offers a simple and intuitive interface for service mesh deployment, configuration, and testing, making it easy for developers and operators to manage service meshes across different environments.

Meshery also provides a range of testing and observability features, including performance testing, chaos engineering, and distributed tracing, which enable organizations to ensure the reliability and resilience of their microservices architectures, even as they scale and evolve.

We plan to deploy our system, consisting of several microservices, using an AWS Elastic Kubernetes Cluster. We will use Istio as the service mesh. We will manage the entire system using Meshery.

Below are brief descriptions of the technologies we have selected in addition to Meshery which has been described in previous section:

- **AWS EKS** [1] (Elastic Kubernetes Service) is a managed Kubernetes service that simplifies the deployment, management, and scaling of containerized applications using Kubernetes in the AWS Cloud.
- **Istio** [2] is an open-source service mesh platform that provides common networking, security, and observability features for managing communication between microservices in a distributed application. It is built on top of Envoy and provides powerful features such as traffic management, service discovery, load balancing, circuit breaking, security, and observability.
- **Docker** [3] open-source containerization platform used to create, deploy, and run applications in isolated containers. We will be using Docker to conternaize our applications for their deployment in EKS.

3 Case study concept description

We will be simulating a factory inspired system that consists of four microservices communicating with each other using different protocols. Each microservice will be communicating with at least one other microservice and will be doing data processing. One of the services will be taking requests from an outside source and will provide responses that are produced as a result of other microservices' work.

We want to explore Meshery's observability features (such as tracing of requests) and utilize the performance testing, apply meshery features to debugging and monitoring microservices, so that we can find the optimal component setup for given input data patterns. We want also to check the possibilities of microservices configuration, such as traffic control, inject latency or perform context-based routing.

[1] https://aws.amazon.com/eks/
[2] https://istio.io/
[3] https://www.docker.com/

4 Solution architecture

5 Environment configuration description

6 Installation method

7 How to reproduce - step by step

7.1 Infrastructure as Code approach

8 Demo deployment steps

8.1 Configuration set-up

8.2 Data preparation

8.3 Execution procedure

8.4 Results presentation

9 Summary – conclusions

References

Meshery Webinar: https://www.cncf.io/online-programs/cncf-on-demand-webinar-meshery-the-service-mesh-manager/