MoSIG 2 HECS 2020-2021
ACS Homework

AYOUB SOUSSI
SOUAD ELAISSAOUI

Homework Part Three - Service Mesh

“Service mesh” is set of products that seek to solve the problems that microservices’ architectures create. Because this kind of architecture produces more moving parts to connect and secure (Managing all of the network services, load balancing, traffic management, authentication and authorization, etc). To have a solution for these challenges, we can can decouple the application that we work on it, at layer five of the network stack.

1) Describe the 2 platforms : Istrio & Maesh

a- Istrio
Born from a collaboration between IBM, Google, Red Hat and Lyft (a California competitor of Uber), Istrio aims to offer a service networking solution to promote communication as well as dynamic interactions between services in a modern micro-services.

Istrio functions as a service mesh providing two basic architectural elements, namely: a data plane and a control plane. The data plane manages network traffic between services in the mesh. All this traffic is intercepted and redirected by a network proxy system.

As for Istrio, the proxy is provided by an open source project called Envoy. A second component of the data plane, Mixer, brings together Envoy telemetry and statistics data and traffic flow between services. The control plane, the heart of Istrio, manages and secures the data plane. It configures both Envoy proxies and Mixers that apply network policies for services, for example to define who can communicate with whom and when. The control plane also provides a programmatic abstraction layer for the data plane and all of its behaviors. Three other Istio services complete the offer: Istrio Pilot, Istrio Citadel and Istio Gallery. Istio Pilot takes the traffic behavior rules provided by the control plane and converts them into configurations applied by Envoy taking into account the local management mode. Pilot allows Istio to work with orchestration systems other than Kubernetes, as long as they behave consistently with each other. Istio Citadel controls authentication and identity management between services. Finally, Istio Gallery takes the user-specified configurations for Istio and converts them to valid configurations for the other components of the control plane. This is another element that allows Istio to use different orchestration systems in a transparent way. [1]

b- Maesh
Maesh is a brand-new open source service mesh designed to deliver the advantages of a service mesh while eliminating the complexity inherent to similar infrastructure layers. Maesh is designed to install easily and can be put into use within minutes. It’s built with a lightweight simplicity that lets developers connect, secure, and monitor traffic within their microservices-based application environments, without the need to overcome a daunting learning curve and steep overheads. With its simpler service mesh experience, Maesh helps drive greater microservices adoption by making it

possible for developers to easily improve application performance while visualizing traffic patterns, optimizing internal traffic, and securing communication within any Kubernetes environment.

Maesh is built on top of Traefik, an open source cloud-native edge router also known for its simplicity-by-design. Maesh is container-native and compliant with the latest Service Mesh Interface specification, ensuring its interoperability with pre-existing solutions in Kubernetes.

Maesh endpoints are able to run in parallel with existing user services, giving developers the choice to opt-in and add services to the mesh as they wish, while withholding others as preferred. In the same way, developers can easily test out services on the Maesh service mesh, and revert them to user services if necessary.

2) The comparison criterias

Comparing service meshes can be difficult, because there are so many categories of features to choose from and so many aspects to consider within each category. So the best solution to decide which a service mesh ti choose is to determine the two or three most important features.

a-Service Mesh Architecture

Istio's conduit uses centralized services. It uses a “sidecar” pattern that places a proxy running in a separate container within each pod. This sidecar container receives the data from and sends the data to the application. It also does the heavy lifting involved with moving or transforming the data to other pods or to spaces outside the cluster. Istio uses the Envoy proxy to perform this function, which appears to be the best-documented and best-supported choice. [2]
Maesh’s mesh controller runs in a dedicated Kubernetes pod, directly handling all configuration parsing and routing of communication through proxy endpoints that run on each node. In this way, Maesh avoids the need for sidecar containers, or making any modifications to Kubernetes objects; the Maesh endpoints are directly accessible. [4]

b-Traffic Management

The traffic management picture is somewhat complicated.
At present, Istio has more traffic management features than Maesh. But, Maesh Connect has been trying to do the same, recently adding features, we talk about load balancing, retries and fail-overs, circuit breakers, and rate limits.

c-Security

Both products have good basic support for certificate rotation and external root certificate support, but Istio leads the pack when it comes to security features. With respect to mutual TLS (mTLS), Istio and maesh offers support for both HTTP and TCP. Maesh has an easy visibility into microservices traffic, it provides fully-secure access controls. We should also mention that Istio is stronger on the policy management front, because it allows different providers to integrate their products into the “template” policy management framework, and it allows administrators to set rules that determine which applications can communicate with each other.

D-Observability

First of all we could talk about monitoring (Metrics) : Istio has close integration with Kiali. Kiali is an observability tool designed for Istio that can produce metrics, infer network topology, and integrate with Grafana for more advanced querying capabilities [2], As *far as we know*, there are no such thing in maesh.

Secondly, the tracing : In order to fully take advantage of tracing applications, your applications may need to be adjusted to add appropriate headers. In this sense, tracing differs from other service mesh features. According to the servicemesh.es website [3], Istio is compatible only with Jaeger's, Zipkin's, and Solarwinds' tracing backends, Maesh does not supports any provider adhering to the OpenCensus standard. This includes Jaeger and Zipkin and Solarwinds [2]
Finally, Logging : If you run a service mesh, then it is quite likely that you will want to log events like network activity and policy violations in addition to maintaining your standard application logs. Both of these products have the capability to link up to the standard Kubernetes logging stacks. [2].

e-Maintainability

Service meshes are indead difficult to set up and maintain.
Istio has been considered to be especially difficult to install and operate. The project has tried to address this by abandoning its microservices architecture in favor of a monolithic approach. While it maintains a microservices philosophy internally, with strict boundaries between the code and interactions between what were formerly separate services, from the perspective of the cluster administrator, it is a single process: istiod. While this flexible approach is good for engineering, it can be a challenge to maintain your operation's stability in the face of changes like these.[2]

Both, products can be installed using Helm, so there is little difference among them on that front. Maesh has a reputation as being easier to configure and operate due to its relative architectural simplicity.

3) The comparison grid :

//	Istrio	Maesh
License	1.8	Apache License 2.0
Difficulty	Relatively difficult to install and operate	Easy to get started with and to roll out across a micro-service application.
Developed by	Google, IBM, Lyft	containous/traefik labs
Service Proxy	Envoy	Built on top of Traefik, a Golang open source reverse proxy and load balancer
Ingress Controller	Envoy / Own Concept	The mesh controller runs in a dedicated pod that handles deploying configuration to the proxy nodes.
Used in production	yes	yes
Advantages	Istio can be adapted and extended like no other Mesh. Its many features are available for Kubernetes and other platforms. [3]	Maesh is designed to be non-invasive and is optimized for performance and usability. Therefore, it requires little time to adopt. It focuses on a selection of features to achieve good performance.

Drawbacks	Istio's flexibility can be overwhelming for teams who don't have the capacity for more complex technology. Also, Istio takes control of the ingress controller.[3]	Built on top of Traefik which currently does not support transparent TLS encryption.
Supported Protocols	TCP, HTTP/1.1+,HTTP/2, gRPC	Maesh supports both TCP and HTTP
Sidecar / Data Plane	Automatic Sidecar Injection and CNI plugin.	Maesh does not use a sidecar container but rather handles routing through Maesh proxy endpoints running on each node.
Platform and Extensibility	-Platform : Kubernetes -Cloud Integrations : Google Cloud, Alibaba, IBM Cloud -Mesh Expansion and Multi-Cluster Mesh	-Platform : Kubernetes -Maesh written in Golang and built on top of the cloud native edge router Traefik.
Service Mesh Interface Compatibility	Traffic Access Control, Traffic Specs, Traffic Split, and Traffic Metrics	Maesh supports configuration with SMI
Monitoring Features	Access Log Generation, “Golden Signal” Metrics Generation, Integrated, pre-configured Prometheus, Integrated, pre-configured Grafana, Per-Route Metrics, Compatible Tracing-Backends,Integrated, pre-configured Tracing-Backends,and Dashboard	Network observability is implemented with OpenTracing
Routing Features	Percentage-based Traffic Splits, Load Balancing, and Header- and Path-based Traffic Splits	Load balancing.
Resilience Features	Circuit Breaking, Retry & Timeout, Fault Injection, and Delay Injection	Circuit breakers, retries, and rate limits
Security Features	MTLS, External CA certificate and key pluggable, and Authorization Rules	Certificate rotation and external root certificate support

References

[1] https://www.redhat.com/en/topics/microservices/what-is-istio

[2]https://logz.io/blog/istio-linkerd-consul-comparison-service-meshes/

[3] https://servicemesh.es/

[4] https://jaxenter.com/maesh-service-mesh-163336.html