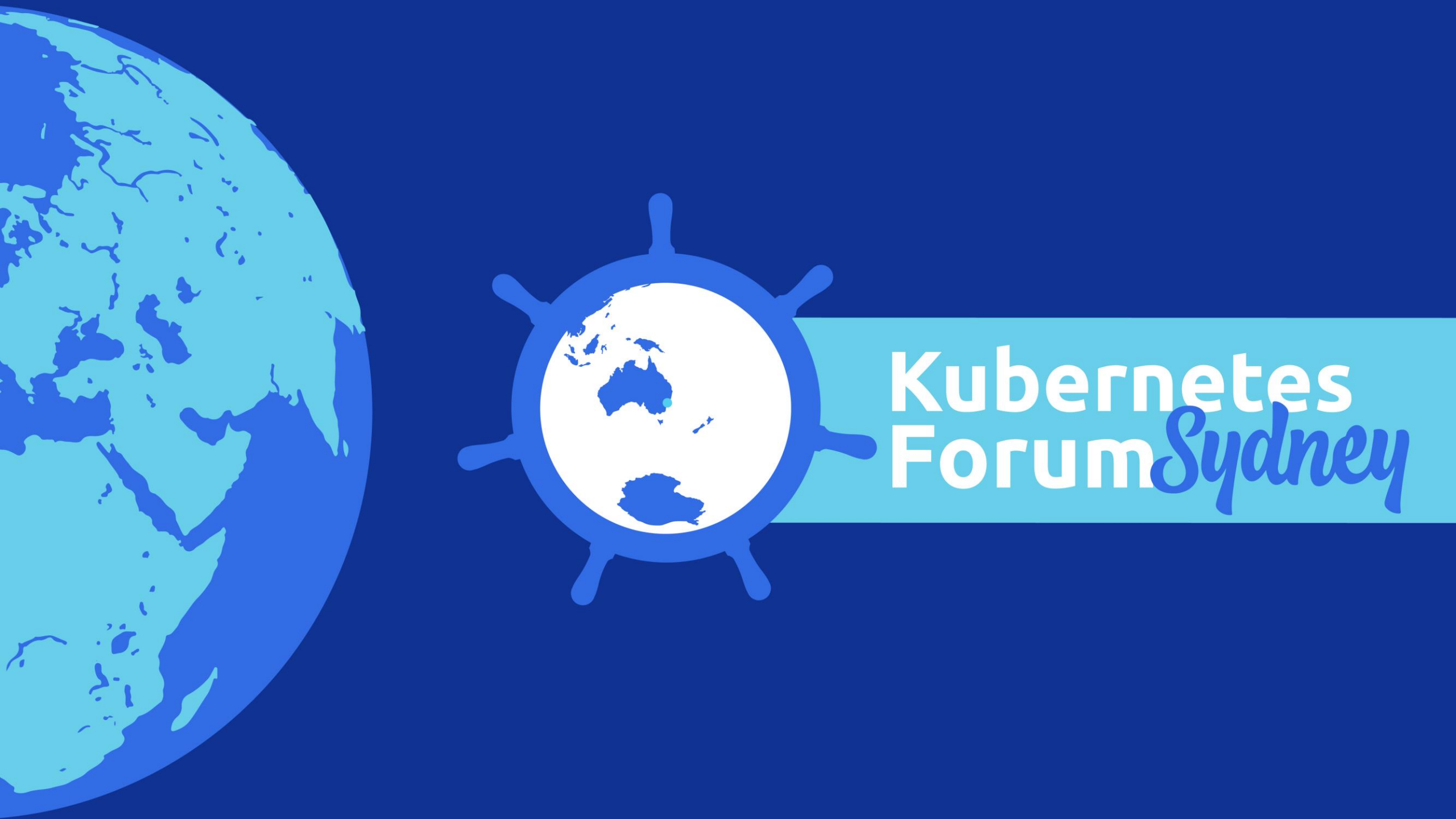

Kubernetes
Forum Sydney

Kubernetes Forum Sydney

Navigating the Service Mesh Landscape

Paul Bouwer

Paul Bouwer
Software Engineer - Microsoft
@pbouwer
paulbouwer

- Why do we need a Service Mesh?
- What does a typical Service Mesh offer?
- Considerations when selecting a Service Mesh
- Istio, Linkerd and Consul
- Look at broader eco-system


Why?

Security

- Firewall is **security boundary**
- Security managed via **ip address**
- **Secure** communication at **boundary**

Security

- **Zero trust** networks
- **Strong identity** and **secure** communication
- Ip addresses are **dynamic**
- Workloads are **dynamic**

Delivery

- **Known or fixed** infrastructure
- **Monolithic** releases

Delivery

- **Dynamic** workloads and versions
- **Progressive**, **A/B** and **canary** releases
- **Observability** requirements
- **Traffic policy** requirements across **multiple dependent** services

Resiliency and Observability

- **Observability**, **policy** (retries, timeouts, rate limiting, circuit breaking) managed in **code** as part of **application**
- **Approved** set of libraries for a **single language** or **framework**

Resiliency and Observability

- Workloads built using **different languages** and **frameworks**
- Requirements for **consistent** observability and policy across **all workloads**


What?

> A service mesh is the connective tissue between your services that adds additional capabilities like traffic control, service discovery, load balancing, resilience, observability, security, and so on.
>
> A service mesh allows applications to offload these capabilities from application-level libraries and allow developers to focus on differentiating business logic.

Introducing Istio Service Mesh for Microservices
by Burr Sutter, Christian Posta

> Istio decouples operational aspects of the services from the implementation of the services.

Eric Brewer, VP Infrastructure & Google Fellow, Google Cloud

> Linkerd moves visibility, reliability, and security constraints down to the infrastructure layer, out of the application layer.

William Morgan, CEO, Buoyant

Typical Capabilities

Traffic management

- **Protocol** – layer 7, http, grpc
- **Dynamic Routing** – conditional, weighting, mirroring
- **Resiliency** – timeouts, retries, circuit breakers
- **Policy** – access control, rate limits, quotas
- **Testing** - fault injection

Typical Capabilities

Security

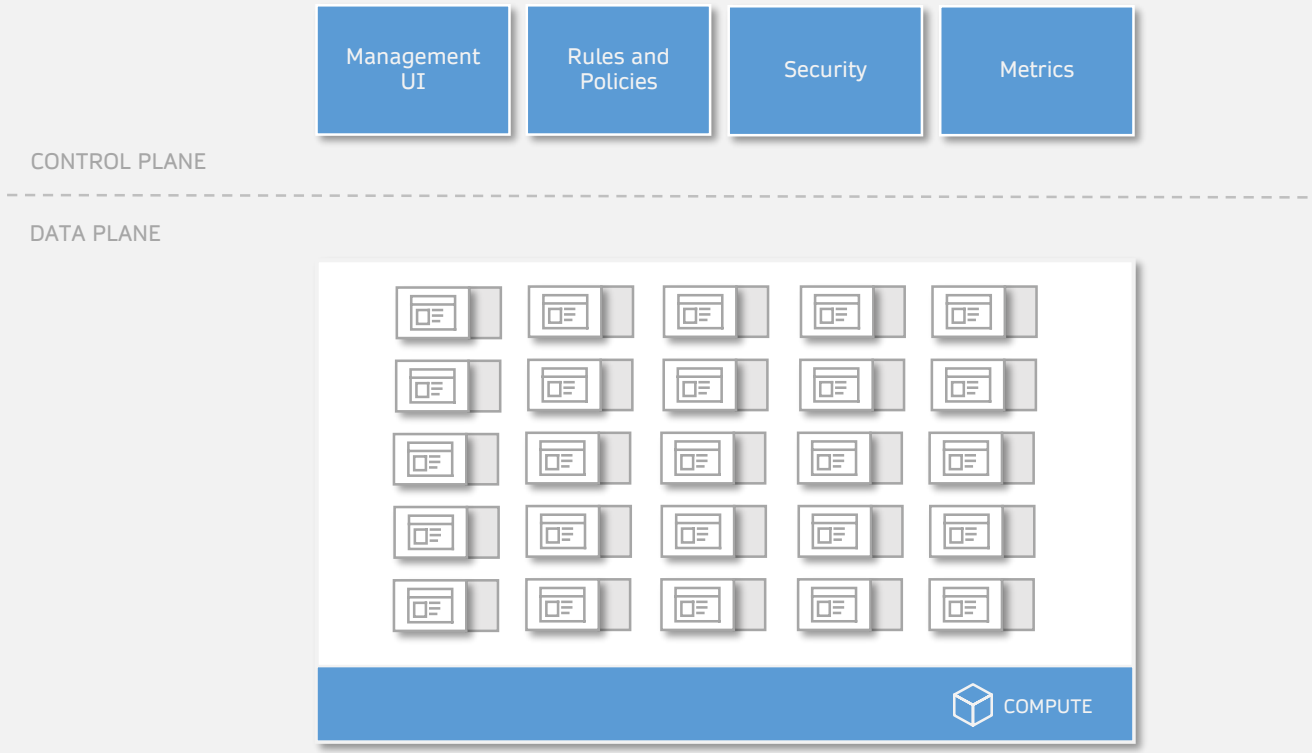
- **Encryption** – mTLS, certificate management, external CA
- **Strong Identity** – SPIFFE or similar
- **Auth** – authentication, authorisation

Typical Capabilities

Observability

- **Metrics** – golden metrics, Prometheus, Grafana
- **Tracing** - traces across workloads
- **Traffic** – cluster, ingress/egress

Typical Architecture



Considerations

Consider Multiple Aspects

- **Technical** - traffic management, policy, security, observability
- **Business** - commercial support, foundation (CNCF), OSS license, governance
- **Operational** – installation/upgrades, resource requirements, performance requirements, integrations, mixed workloads
- **Security** - certificate management and rotation, external CA

Ask the following Questions

- Is an **Ingress Controller** sufficient for my needs?
- Can my workloads and environment **tolerate** the additional **overheads**?
- Is this adding **additional complexity** unnecessarily?
- Can this be adopted in an **incremental** approach?

Service Meshes


Istio


LINKERD


HashiCorp
Consul

Full featured, customisable and extensible service mesh

https://istio.io/

Project Details

- **Announced** - May 2017, GA (1.0) in July 2018
- **Governance** - Google, IBM
- **OSS** - Apache 2.0
- **Commercial support** – Aspen Mesh
- **Cloud offerings** – Google Cloud, IBM Cloud

Full featured, customisable and extensible service mesh

https://istio.io/

Design Goals

- Maximize Transparency
- Extensibility
- Portability
- Policy Uniformity

Full featured, customisable and extensible service mesh

https://istio.io/

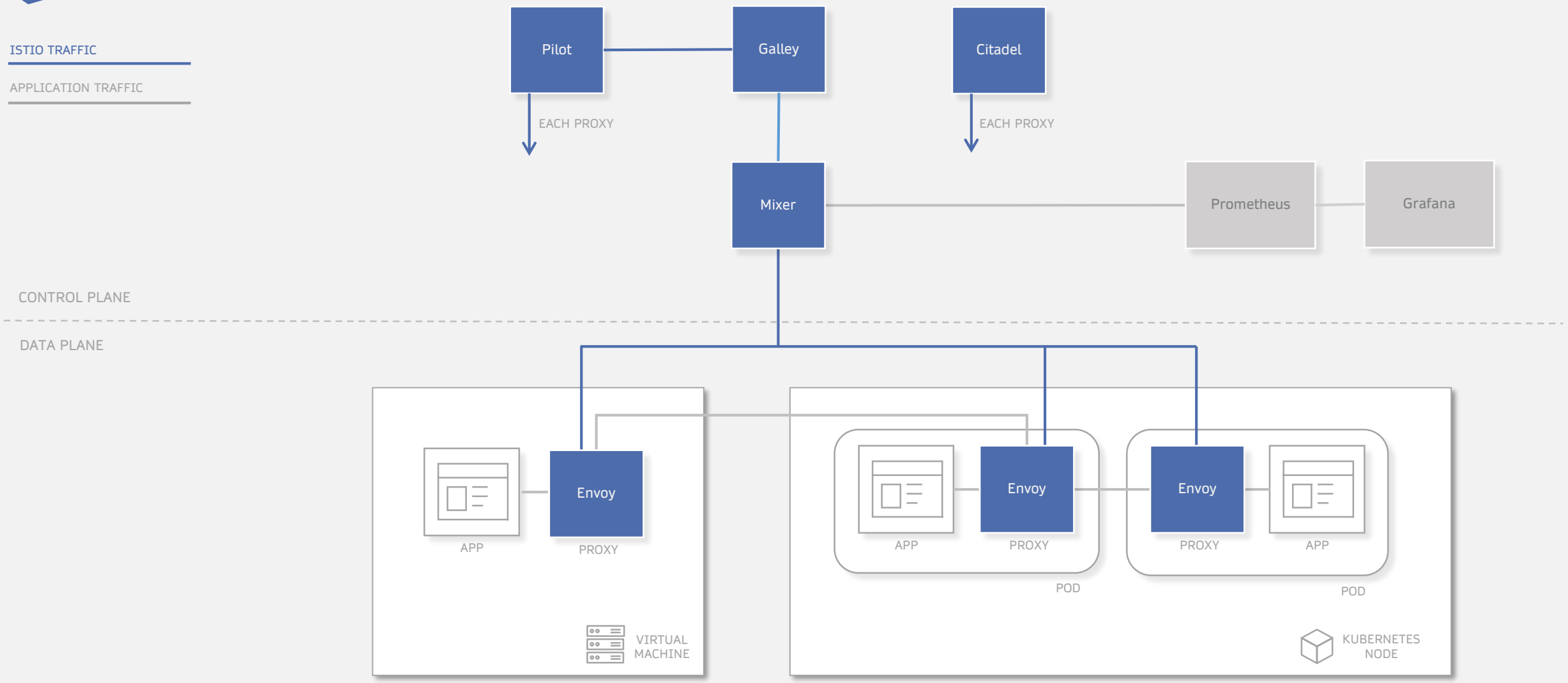
Capabilities

- **Mesh** – gateways (multi-cluster), vms (expansion)
- **Traffic Management** – routing, splitting, timeouts, circuit breakers, retries, ingress, egress
- **Policy** – access control, rate limit, quota, custom policy adapters
- **Security** – authentication (jwt), authorisation, encryption, external CA (HashiCorp Vault), node agent and envoy SDS
- **Observability** – golden metrics, mirror, tracing, custom adapters, Prometheus, Grafana

Istio

ISTIO TRAFFIC

APPLICATION TRAFFIC



Microsoft

Full featured, customisable and extensible service mesh

https://istio.io/

Operational

- **Installation** – istioctl cli, Helm Chart deprecated, Istio Operator coming
- **Configuration** – profiles, manifests, IstioControlPlane schema
- **Operating** – proxy auto-injection
- **Diagnostics** – istioctl describe, istioctl analyze

Full featured, customisable and extensible service mesh

https://istio.io/

Heads Up

- Complex installation and options
- Complex configuration
- Resource overhead
- Management of certificates for mTLS

Full featured, customisable and extensible service mesh

https://istio.io/

Scenarios

- Require extensibility and rich set of capabilities
- Routing and access control using auth and claims
- Mesh expansion to VM based workloads
- Multi-cluster service mesh


Istio


LINKERD


HashiCorp
Consul

Easy to use and lightweight service mesh

https://linkerd.io/

Project Details

- **Announced** - December 2017, GA (2.0) in September 2018
- **Governance** - CNCF
- **OSS** - Apache 2.0
- **Commercial support** – Buoyant

Linkerd 1.x announced in February 2016, and went GA (1.0) in April 2017

Easy to use and lightweight service mesh

https://linkerd.io/

Design Goals

- Keep it simple
- Minimize resource requirements
- Just work

Easy to use and lightweight service mesh

https://linkerd.io/

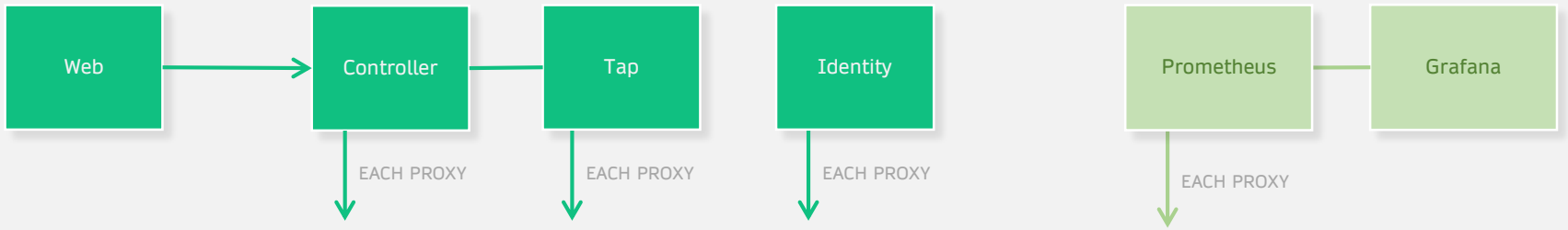
Capabilities

- **Mesh** – built in debugging endpoints and debugging sidecar
- **Traffic Management** – splitting, timeouts, retries, ingress
- **Security** – encryption, certificates auto-rotated every 24 hours, external CA (cert-manager)
- **Observability** – golden metrics, tap, tracing, service profiles and per route metrics, web dashboard with topology graphs, Prometheus, Grafana

LINKERD

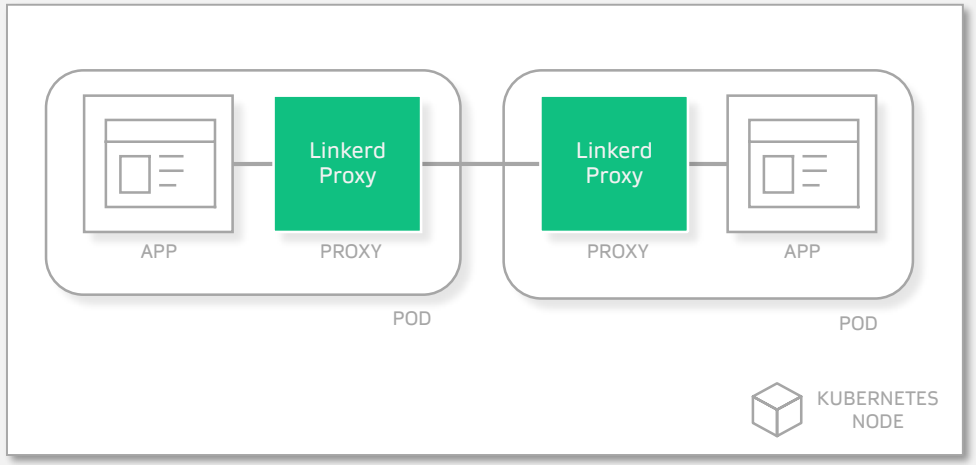
LINKERD TRAFFIC

APPLICATION TRAFFIC



CONTROL PLANE

DATA PLANE



Easy to use and lightweight service mesh

https://linkerd.io/

Operational

- **Installation** – linkerd cli with pre and post checks, multi-stage install for security roles (cluster owner, service owner), optional Helm Chart
- **Configuration** – built in HA mode for control plane, flags via linkerd cli, values via Helm Chart
- **Operating** – proxy auto-injection, dashboard
- **Diagnostics** – tap, debugging endpoints, debugging sidecar

Easy to use and lightweight service mesh

https://linkerd.io/

Heads Up

- Smaller core set of features
- Some features not built in
- Philosophical opinions on core feature set and what should be leveraged from Kubernetes

Easy to use and lightweight service mesh

https://linkerd.io/

Scenarios

- Simple to use with core set of capability requirements
- Low latency, low overhead, with focus on observability and simple traffic management
- Operational simplicity and great out the box experience


Istio


LINKERD


HashiCorp
Consul

A multi data centre aware service mesh to connect and secure services across runtime platforms

https://consul.io/mesh.html

Project Details

- **Announced** - June 2016, GA (1.0) in June 2018
- **Governance** - HashiCorp
- **OSS** - Mozilla Public License 2.0
- **Commercial support** – HashiCorp via Consul Enterprise

A multi data centre aware service mesh to connect and secure services across runtime platforms

https://consul.io/mesh.html

Design Goals

- API Driven
- Run and Connect Anywhere
- Extend and Integrate

A multi data centre aware service mesh to connect and secure services across runtime platforms

https://consul.io/mesh.html

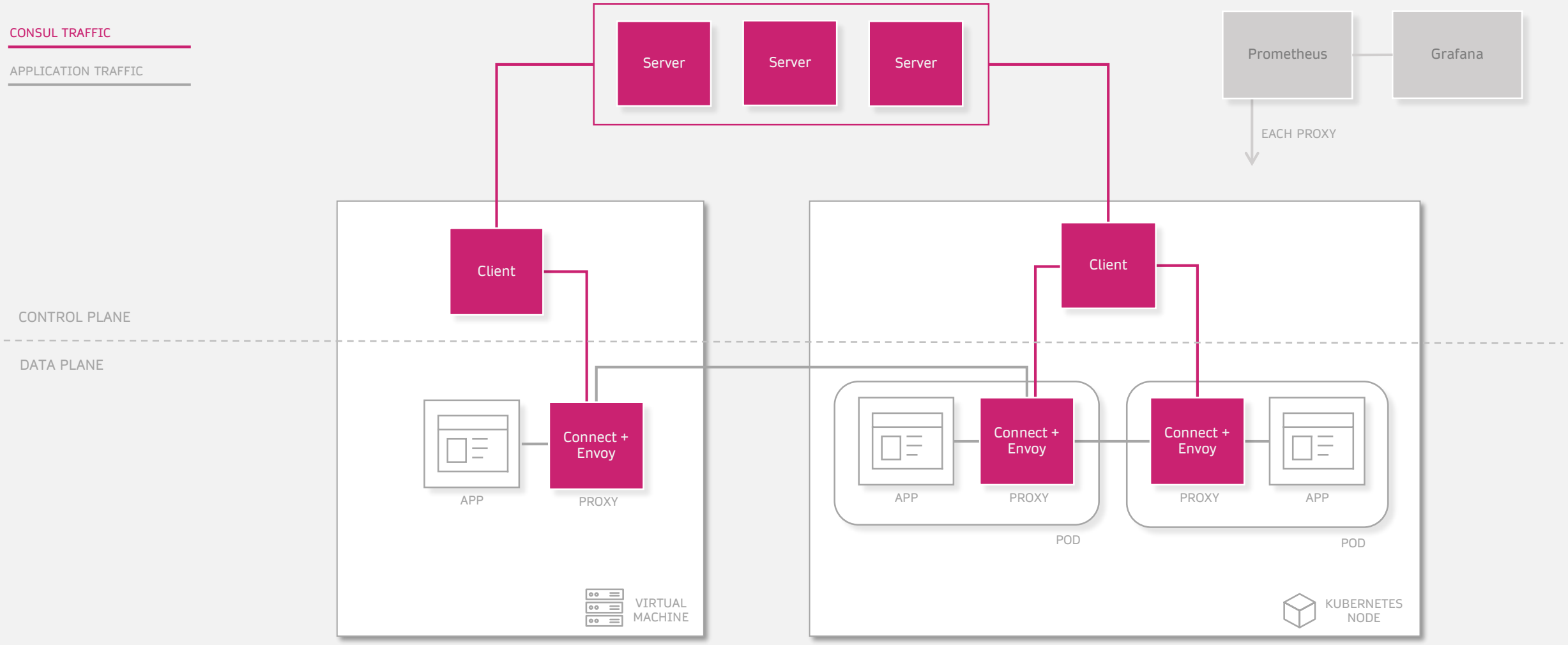
Capabilities

- **Mesh** – gateway (multi data centre), vms (out of cluster nodes), service sync, built in debugging option
- **Proxies** – Envoy, built-in proxy, pluggable, l4 proxy available for Windows workloads
- **Traffic Management** – routing, splitting, resolution
- **Policy** – intentions, ACLs
- **Security** – authorisation, authentication, encryption, SPIFFE based identities, external CA (Vault), certificate management and rotation
- **Observability** – metrics, dashboard, Prometheus, Grafana

HashiCorp Consul

CONSUL TRAFFIC

APPLICATION TRAFFIC



A multi data centre aware service mesh to connect and secure services across runtime platforms

https://consul.io/mesh.html

Operational

- **Installation** – Helm Chart
- **Configuration** – HCL, values via Helm Chart
- **Operating** – proxy auto-injection, dashboard
- **Diagnostics** – Consul Agent proxy

A multi data centre aware service mesh to connect and secure services across runtime platforms

https://consul.io/mesh.html

Heads Up

- Complex installation (lots of Consul specific concepts)
- Need basic understanding of Consul
- No tracing support
- Doesn’t use Kubernetes DNS
- Metrics require additional effort to expose
- Proxied services all listen on localhost

A multi data centre aware service mesh to connect and secure services across runtime platforms

https://consul.io/mesh.html

Scenarios

- Extending existing Consul connected workloads
- Compliance requirements around certificate management
- Multi cluster and/or VM based workloads to be included in the service mesh


Istio


LINKERD


HashiCorp
Consul


Istio


LINKERD


HashiCorp
Consul


Kuma


mæsh

Standardisation

Service Mesh Interface

A standard interface for service meshes on Kubernetes.

https://smi-spec.io/

- A standard interface for service meshes on Kubernetes
- A basic feature set for the most common service mesh use cases
- Flexibility to support new service mesh capabilities over time
- Space for the ecosystem to innovate with service mesh technology

Service Mesh Interface

A standard interface for service meshes on Kubernetes.

https://smi-spec.io/

Microsoft
LINKERD
HashiCorp
solo.io
Red Hat
RANCHER
docker
meshery
weaveworks
ASPEN MESH
Pivotal
vmware
kuma
CANONICAL
kubecost

Service Mesh Interface

A standard interface for service meshes on Kubernetes.

https://smi-spec.io/

- **Traffic Access Control** - configure access to specific pods and routes based on the identity of a client for locking down applications to only allowed users and services.
- **Traffic Specs** - define how traffic looks on a per-protocol basis. These resources work in concert with access control and other types of policy to manage traffic at a protocol level.
- **Traffic Split** - incrementally direct percentages of traffic between various services to assist in building out canary rollouts.
- **Traffic Metrics** - expose common traffic metrics for use by tools such as dashboards and autoscalers.

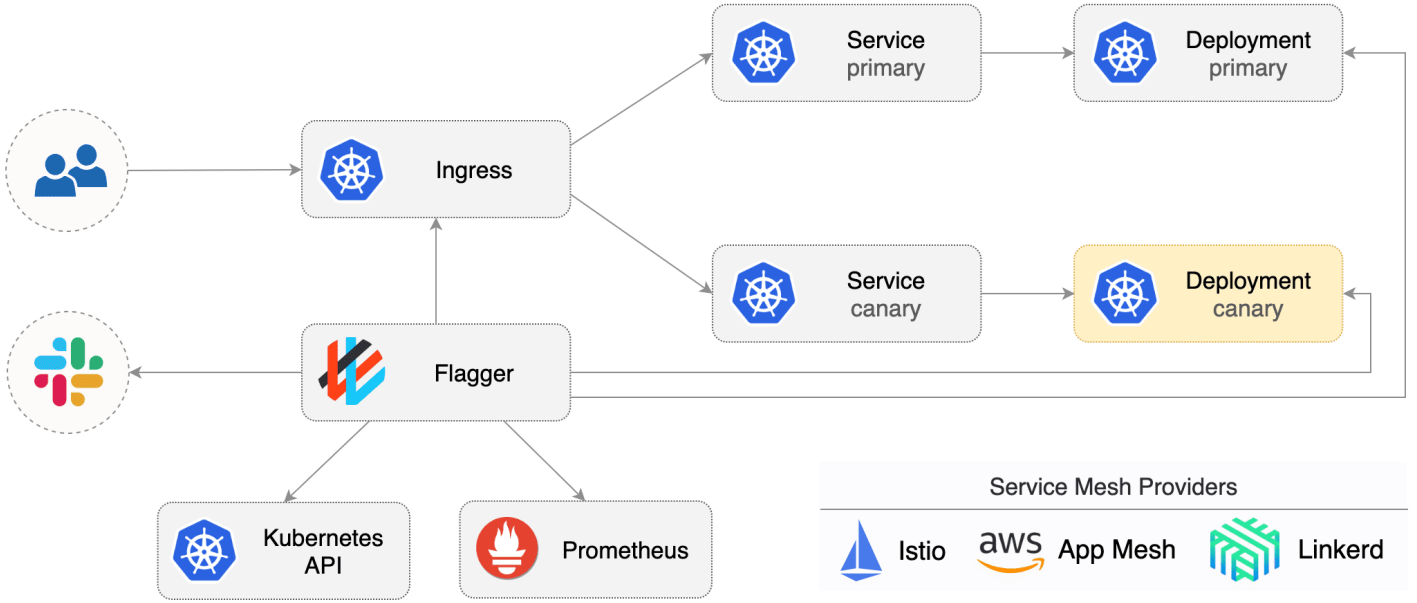

Eco-system

Flagger

Automate and manage canary and other advanced deployments with Istio, Linkerd, AWS App Mesh or NGINX for traffic shifting.

Integrated Prometheus metrics control canary deployment success or failure.

https://weave.works/oss/flagger
https://docs.flagger.app

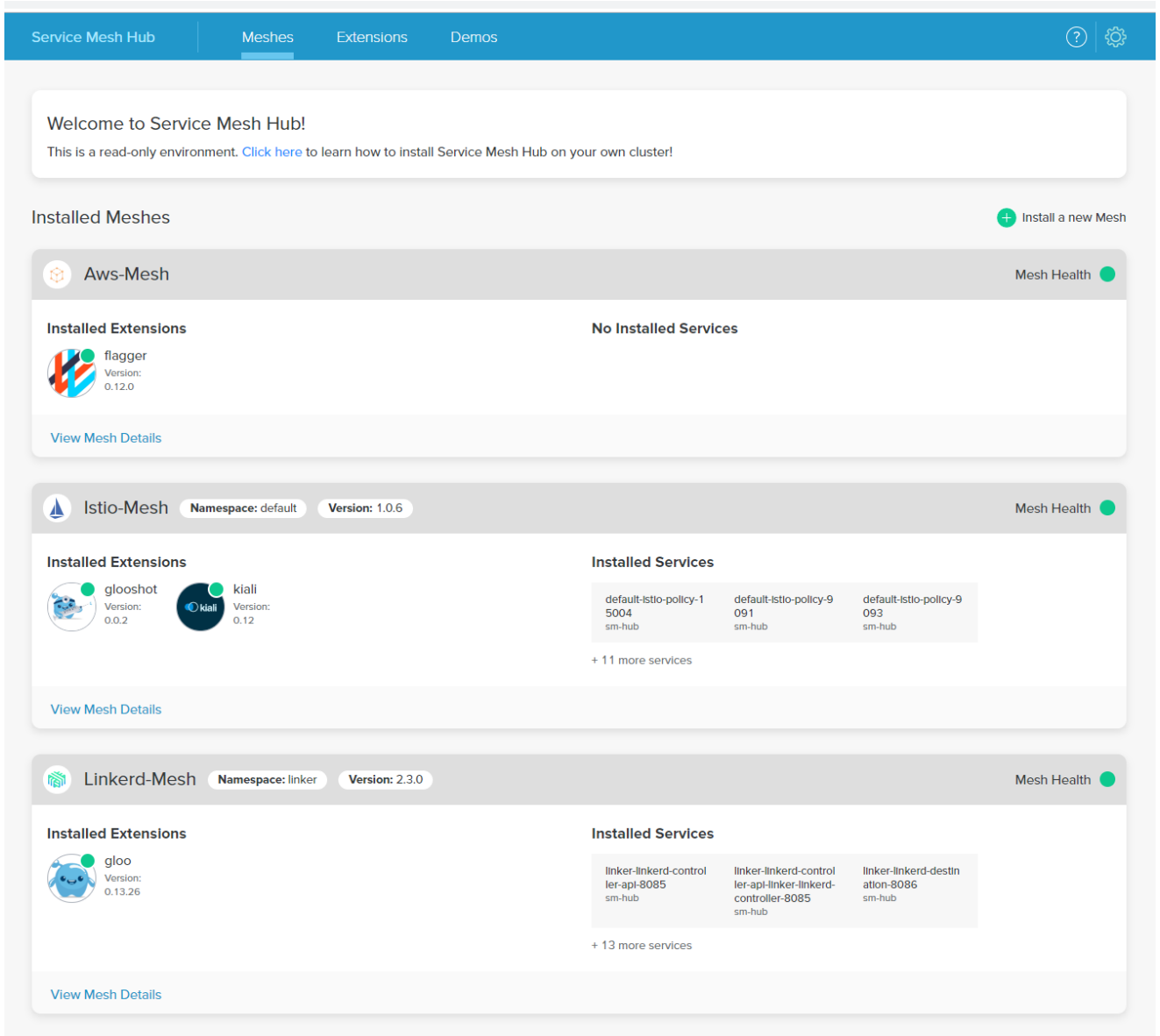


solo.io

The **Service Mesh Hub** is an industry hub designed for the community and ecosystem to collaborate on service mesh technology and for organizations to deploy and operate their any service mesh on any cloud.

The Service Mesh Hub simplifies the adoption of service mesh for end users and the ecosystem.

https://www.solo.io/servicemeshhub
https://servicemeshhub.io/meshes



Microsoft

Meshery

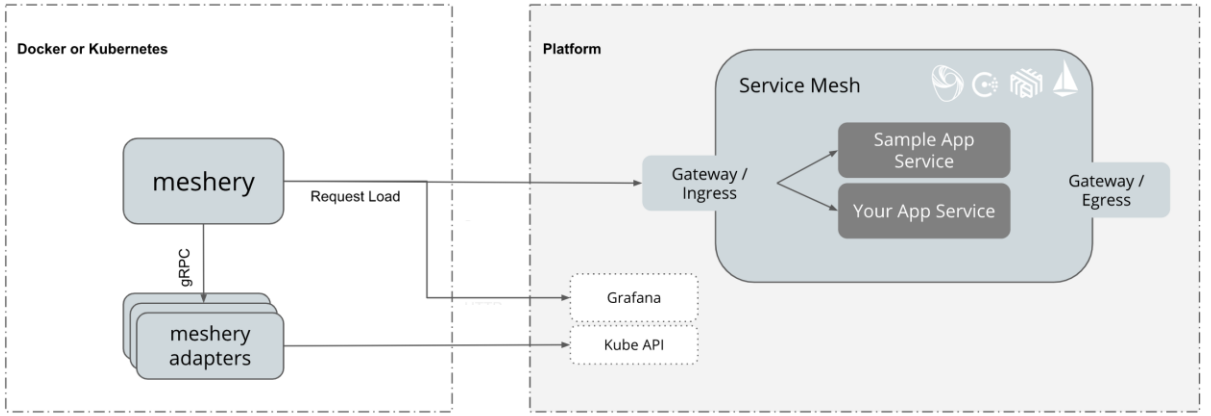
Playground

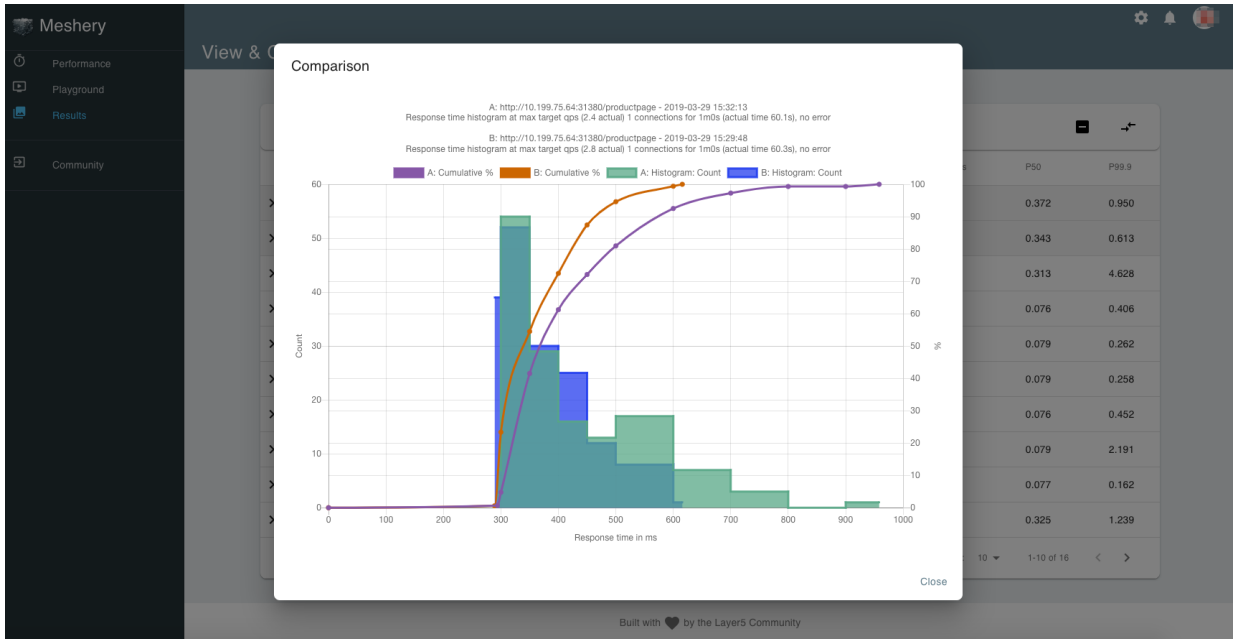
Learn about the functionality of different service meshes and visually manipulate mesh configuration.

Benchmark

Benchmark the performance of your application across different service meshes and compare their overhead.

https://layer5.io/meshery





Microsoft

Takeaways

- Very active space
- Ensure you understand your requirements before selecting a mesh
- Understand the impact of deploying a mesh in your cluster
- Keep an eye on standardisation efforts like Service Mesh Interface (SMI), and the eco-system around Flagger, Service Mesh Hub, and Meshery.


thanks!