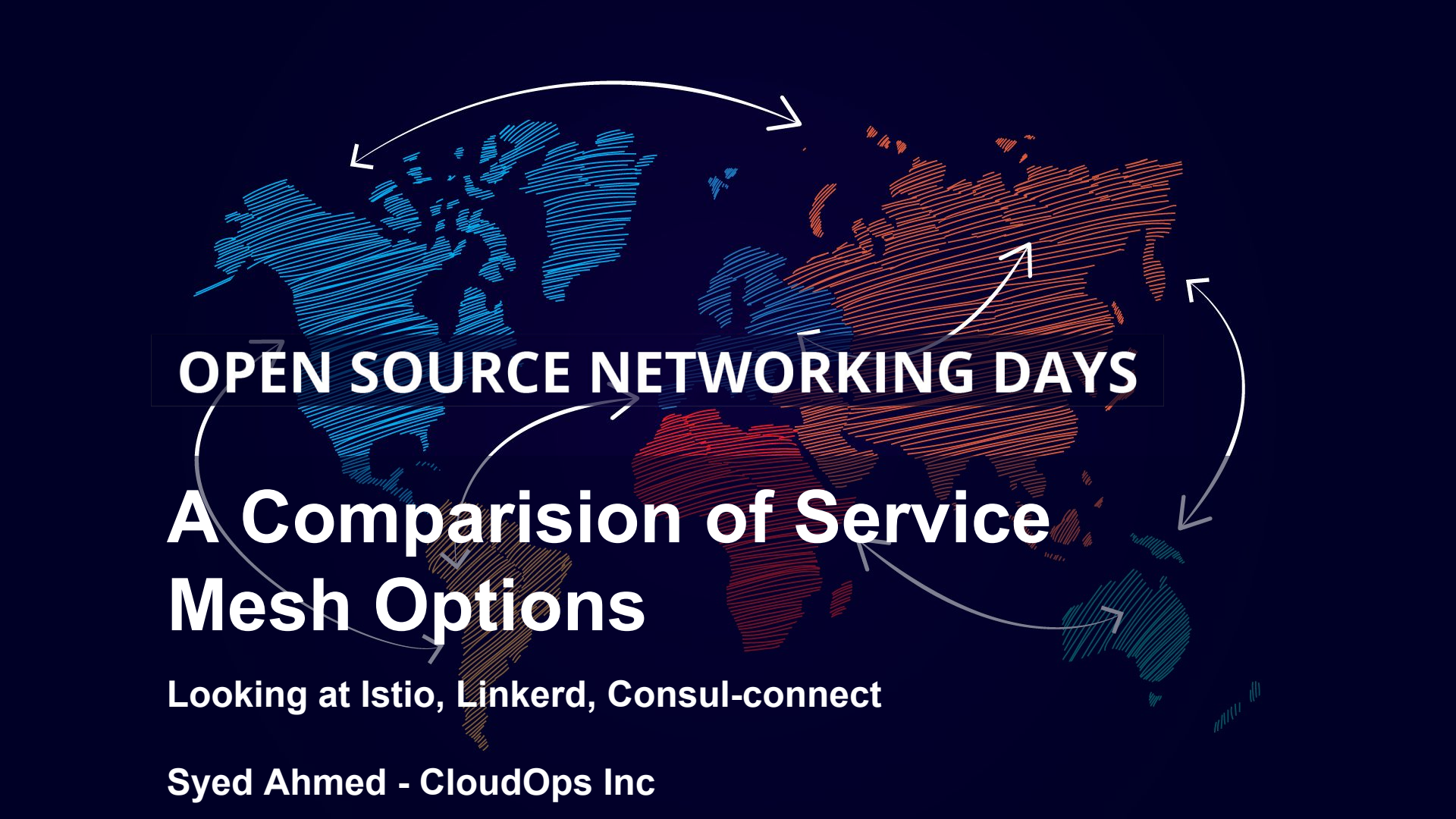
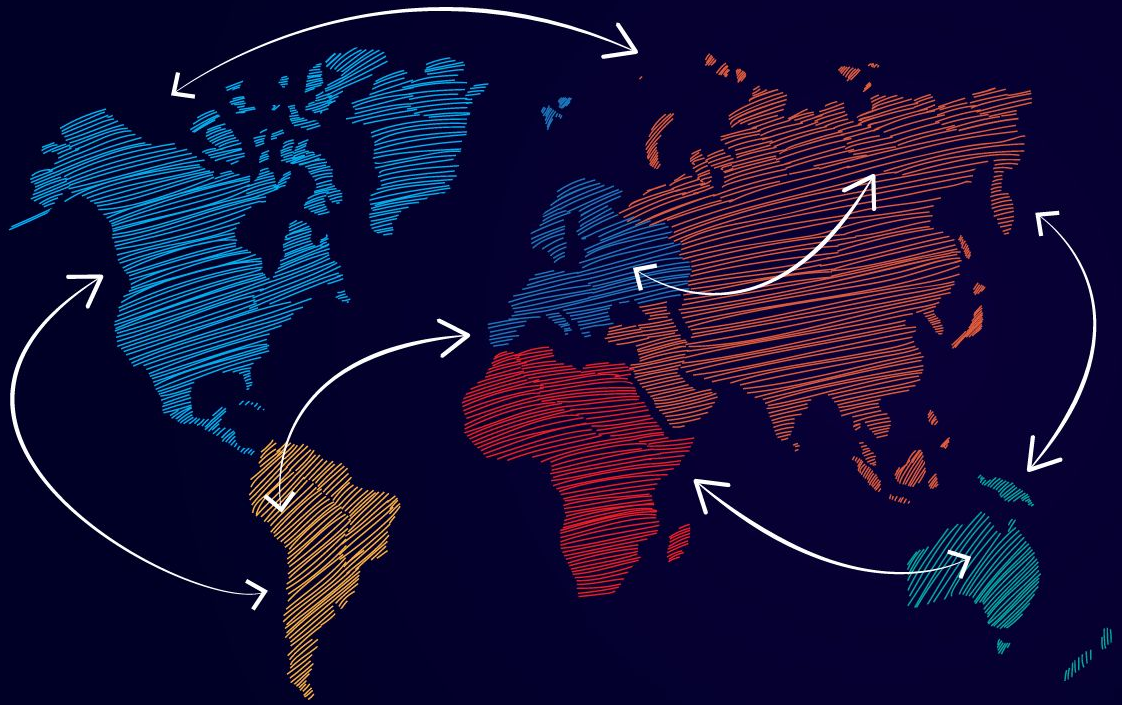


A Comparision of Service Mesh Options

Looking at Istio, Linkerd, Consul-connect

Syed Ahmed - CloudOps Inc



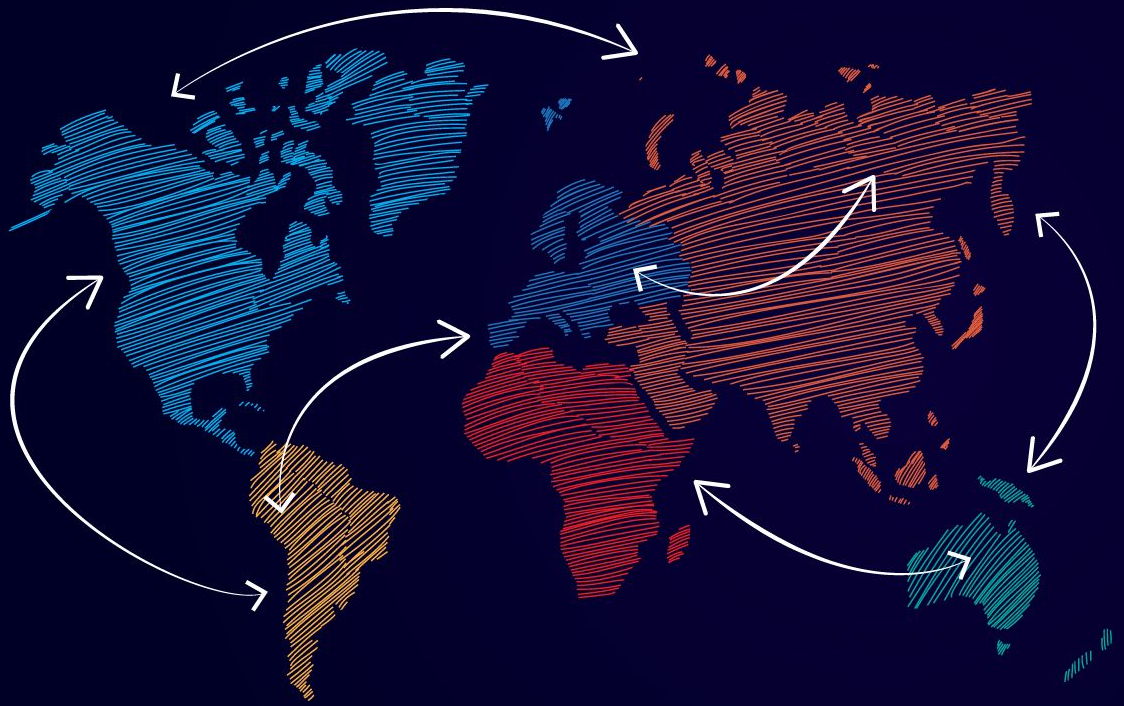
Introduction

About Me

- Cloud Software Architect @ CloudOps
- PMC for Apache CloudStack
- Worked on network modules in Openstack and CloudStack
- Previously worked on the Netscaler LB
- Part of the DevOps team @ Yahoo!

About CloudOps

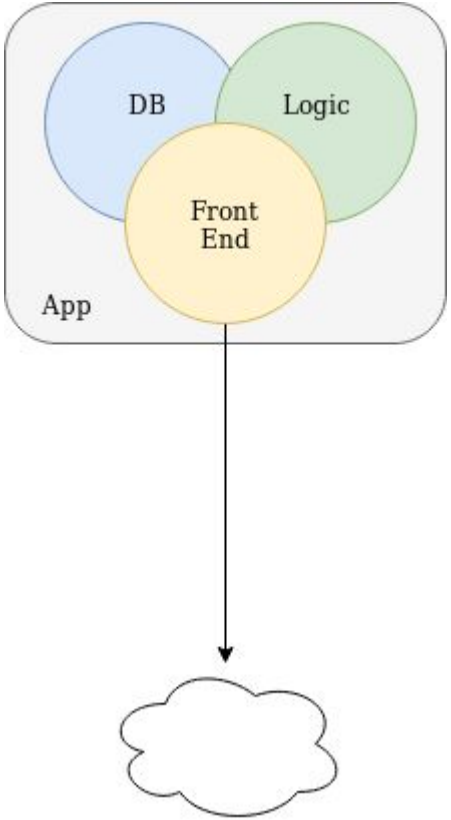
- We Design, Build and Operate Clouds
- Help customer own their destiny in the Cloud
- Vender/Cloud Agnostic



A Case for Service Mesh

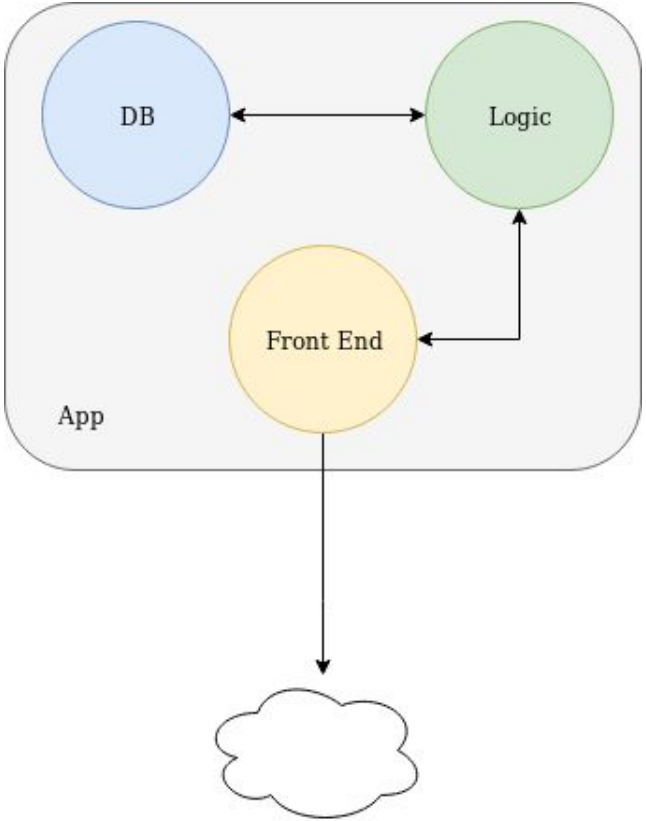
Monolithic Architecture

- Strong Coupling between different modules causing anti-patterns in communicating between different modules
- Difficulties in Scaling
- Updating to new version requires complete re-install
- Problem in one module can cause the whole application to crash
- Difficult to move to a new framework or technology

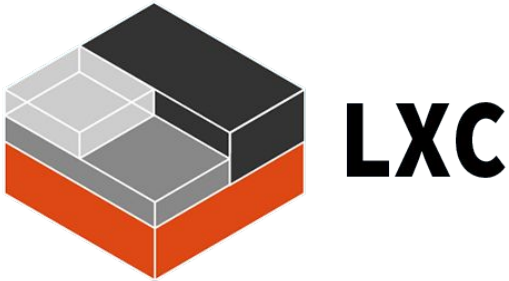


Microservices Architecture

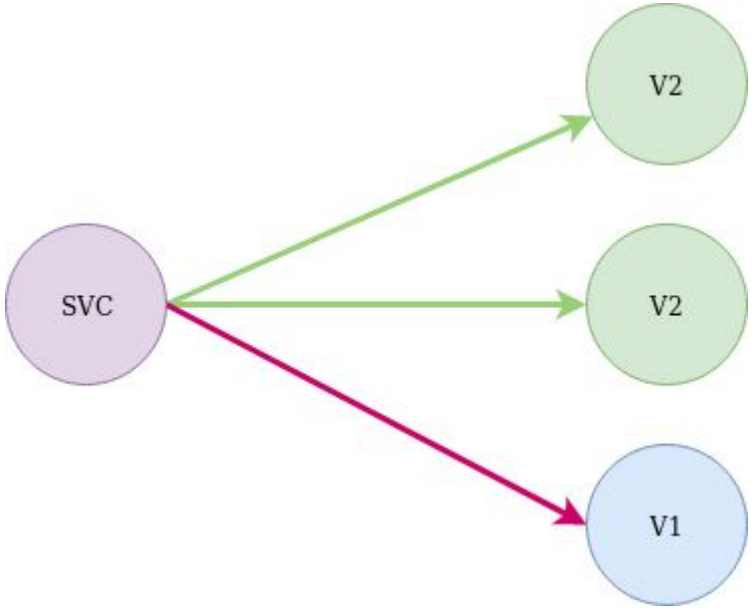
- API contract between different modules/service ensures that each module can be developed and maintained independently
- Each service can be scaled independently
- Updating to new version requires only updates to a specific services
- Allows for easier CI/CD



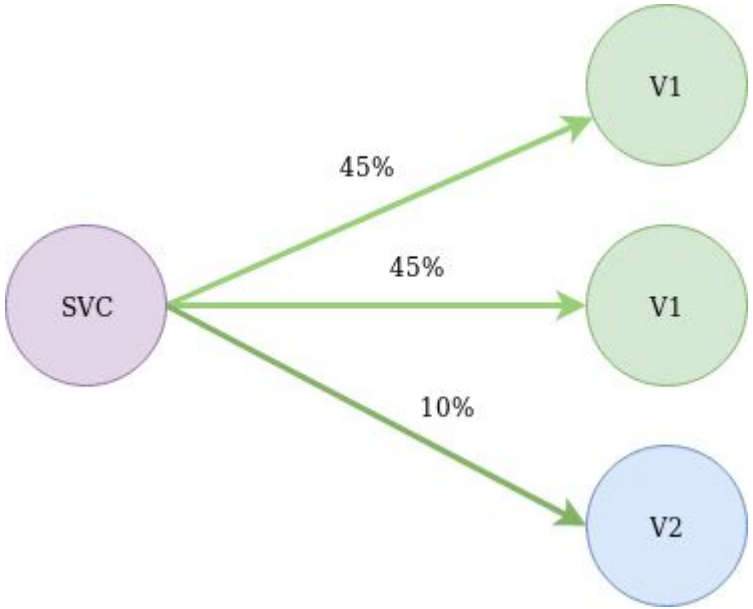
Evolution of the Ecosystem



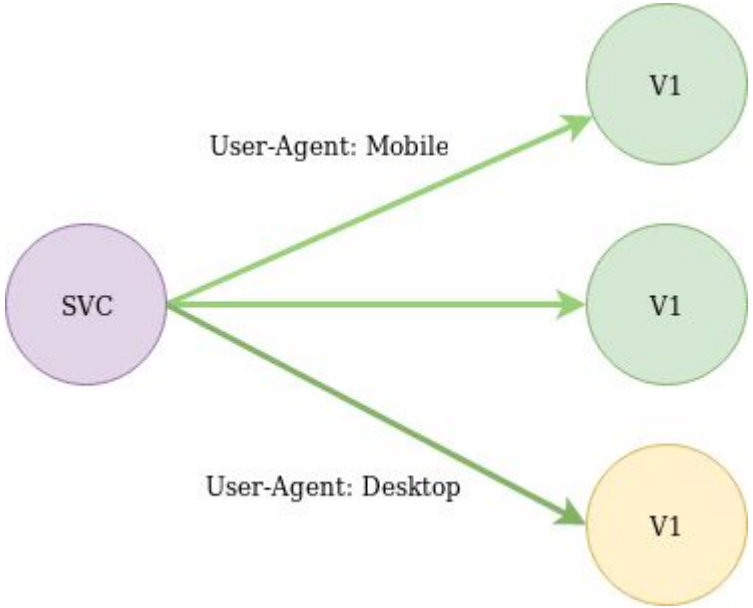
Challenges with the Microservices Architecture



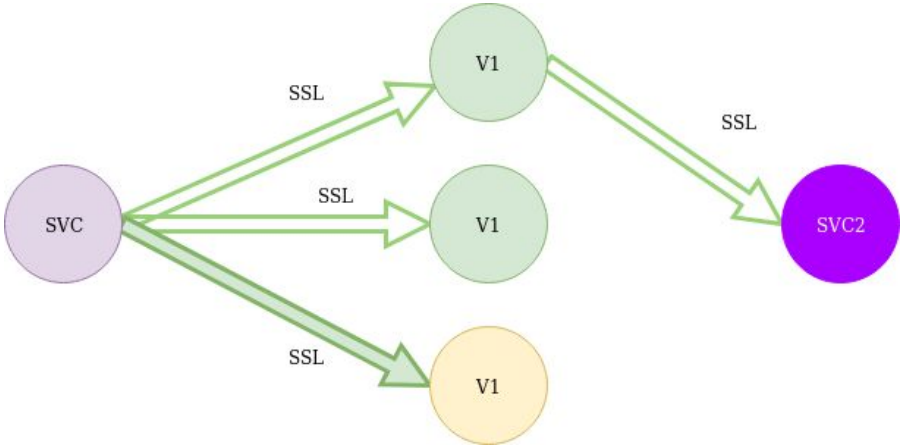
Challenges with the Microservices Architecture



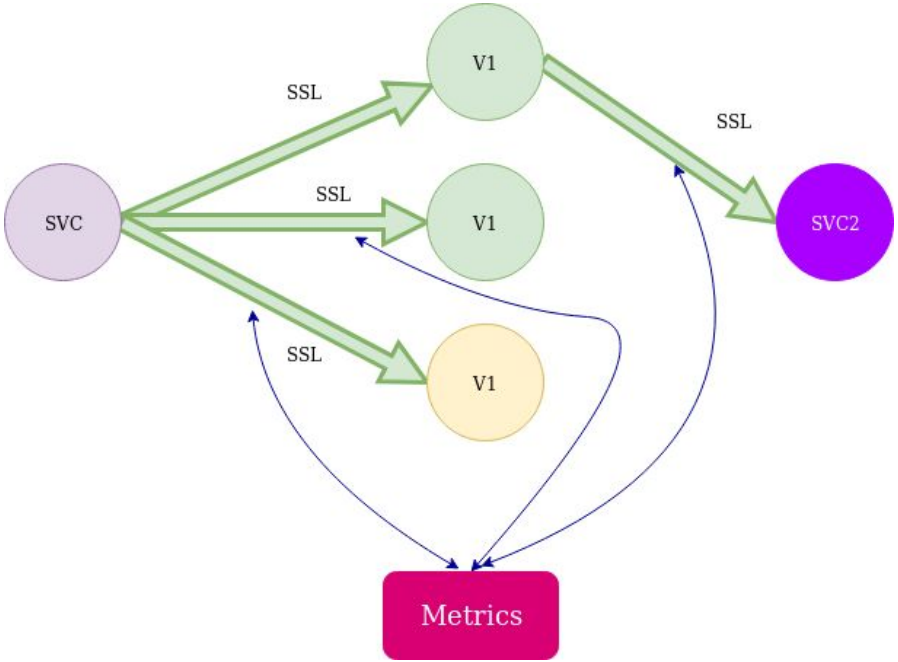
Challenges with the Microservices Architecture



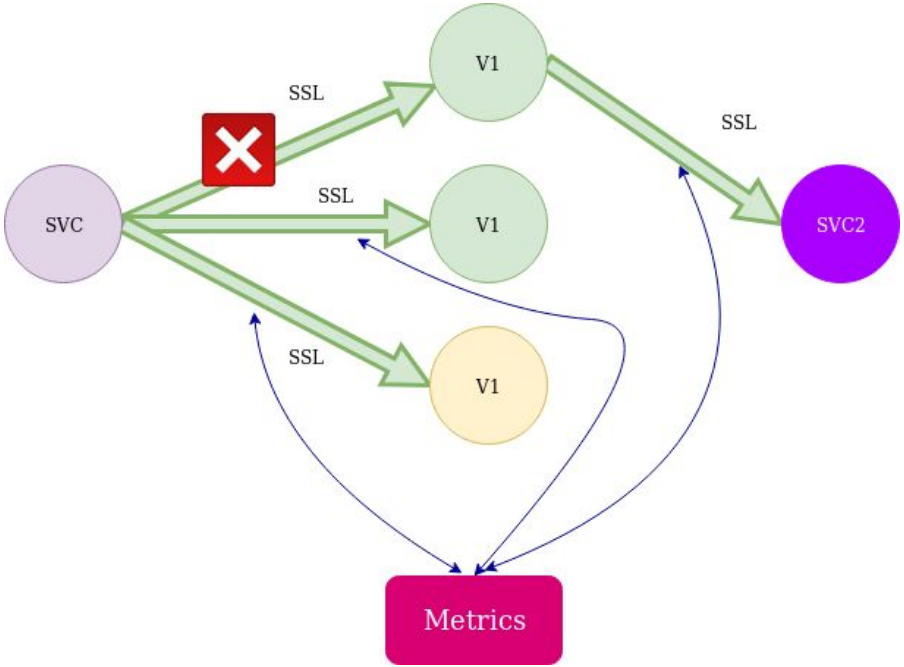
Challenges with the Microservices Architecture



Challenges with the Microservices Architecture



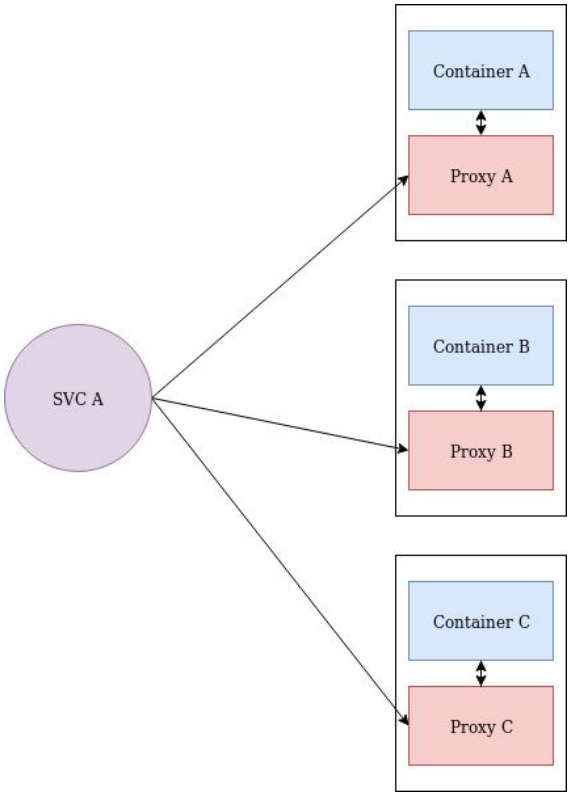
Challenges with the Microservices Architecture

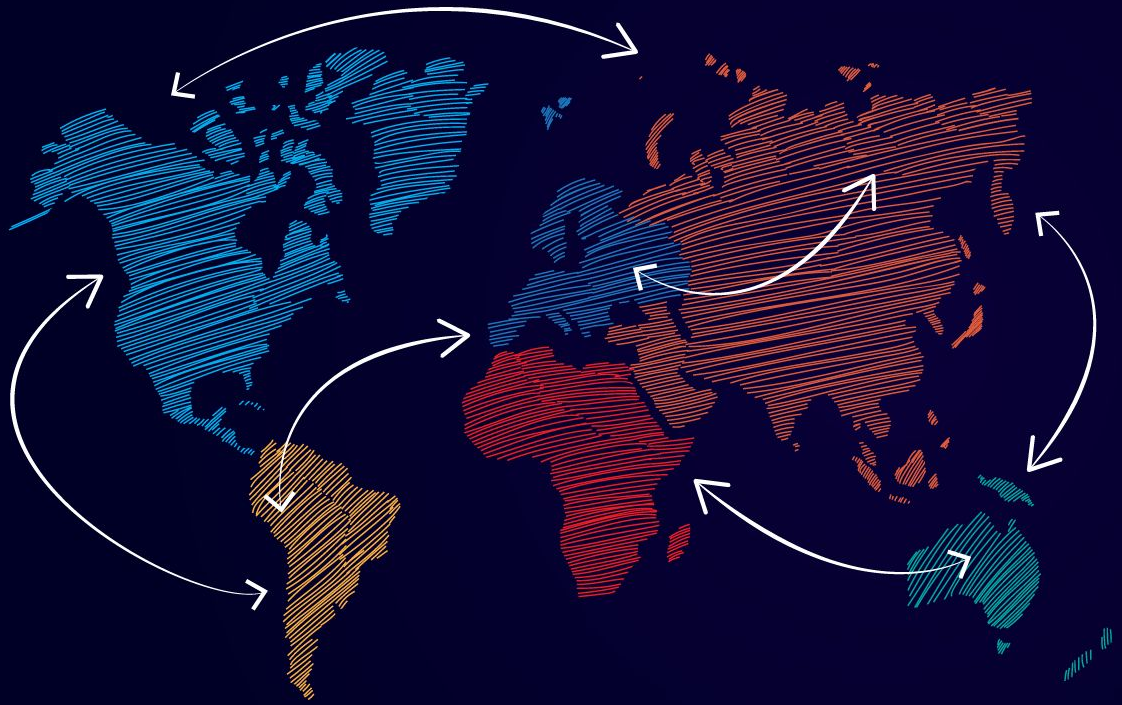


Service Mesh as a Solution

A Service Mesh is the substrate between different microservices that makes connectivity between different microservices possible. In addition to providing networking, a Service Mesh can also provide other features like Service Discovery, Authentication and Authorization, Monitoring, Tracing and Traffic Shaping.

Sidecar Pattern





Istio

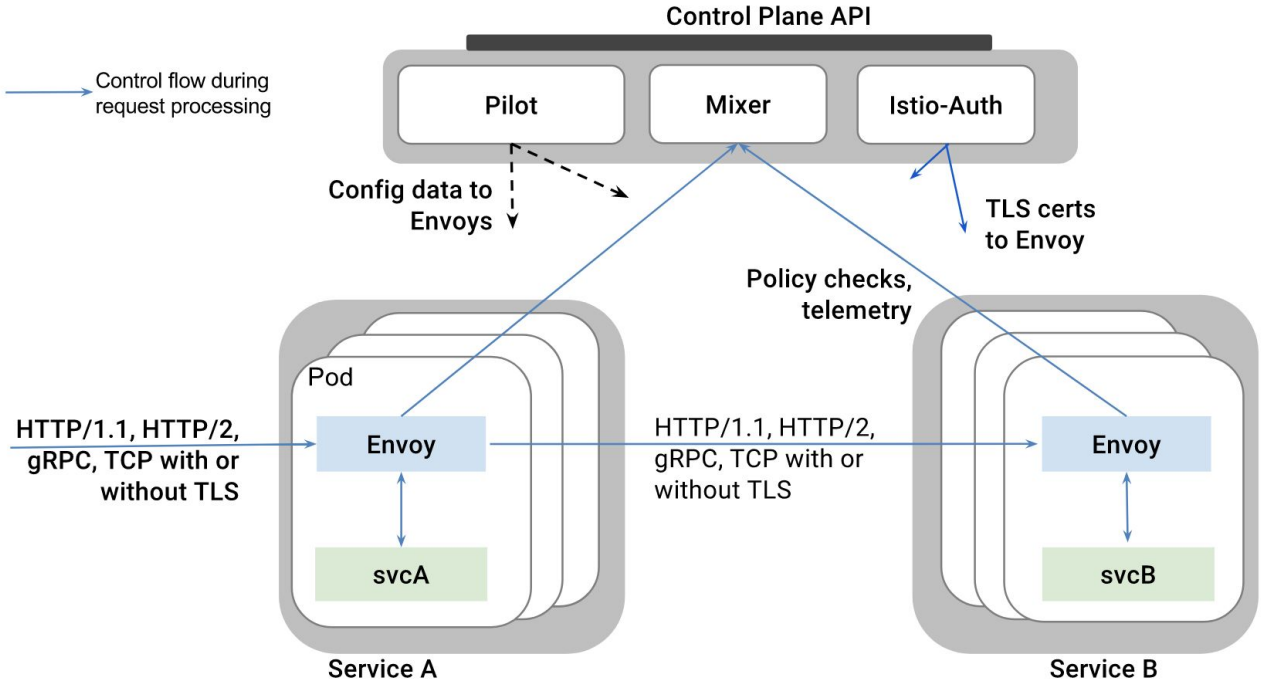
Istio

- Open Sourced by Google, IBM & Lyft in May 2017
- Service Mesh designed to connect, secure and monitor microservices

Istio Features

- Automatic load balancing for HTTP, gRPC, WebSocket, and TCP traffic.
- Fine-grained control of traffic behavior with rich routing rules, retries, failovers, and fault injection.
- A pluggable policy layer and configuration API supporting access controls, rate limits and quotas.
- Automatic metrics, logs, and traces for all traffic within a cluster, including cluster ingress and egress.
- Secure service-to-service communication in a cluster with strong identity-based authentication and authorization.

Istio Architecture



Istio Architecture

Istio Architecture

- **Envoy:** high-performance proxy developed in C++ provides Dynamic service discovery, Load balancing, TLS termination, HTTP/2 and gRPC proxies, Circuit breakers, Health checks, Staged rollouts with %-based traffic split, Fault injection, Rich metrics

- **Pilot:** The core component used for traffic management in Istio is Pilot, which manages and configures all the Envoy proxy instances deployed in a particular Istio service mesh

- **Mixer:** Mixer is a platform-independent component. Mixer enforces access control and usage policies across the service mesh, and collects telemetry data from the Envoy proxy and other services. The proxy extracts request level attributes, and sends them to Mixer for evaluation

- **Citadel:** Citadel provides strong service-to-service and end-user authentication with built-in identity and credential management. You can use Citadel to upgrade unencrypted traffic in the service mesh. Using Citadel, operators can enforce policies based on service identity rather than on network controls

Istio Gateway

Gateway describes a load balancer operating at the edge of the mesh receiving incoming or outgoing HTTP/TCP connections.

```
apiVersion: networking.istio.io/v1alpha3
kind: Gateway
metadata:
  name: httpbin-gateway
spec:
  selector:
    istio: ingressgateway
  servers:
  - port:
      number: 80
      name: http
      protocol: HTTP
    hosts:
    - "httpbin.example.com"
```

Istio VirtualService

A **VirtualService** defines a set of traffic routing rules to apply when a host is addressed. Each routing rule defines matching criteria for traffic of a specific protocol. If the traffic is matched, then it is sent to a named destination service (or subset/version of it) defined in the registry.

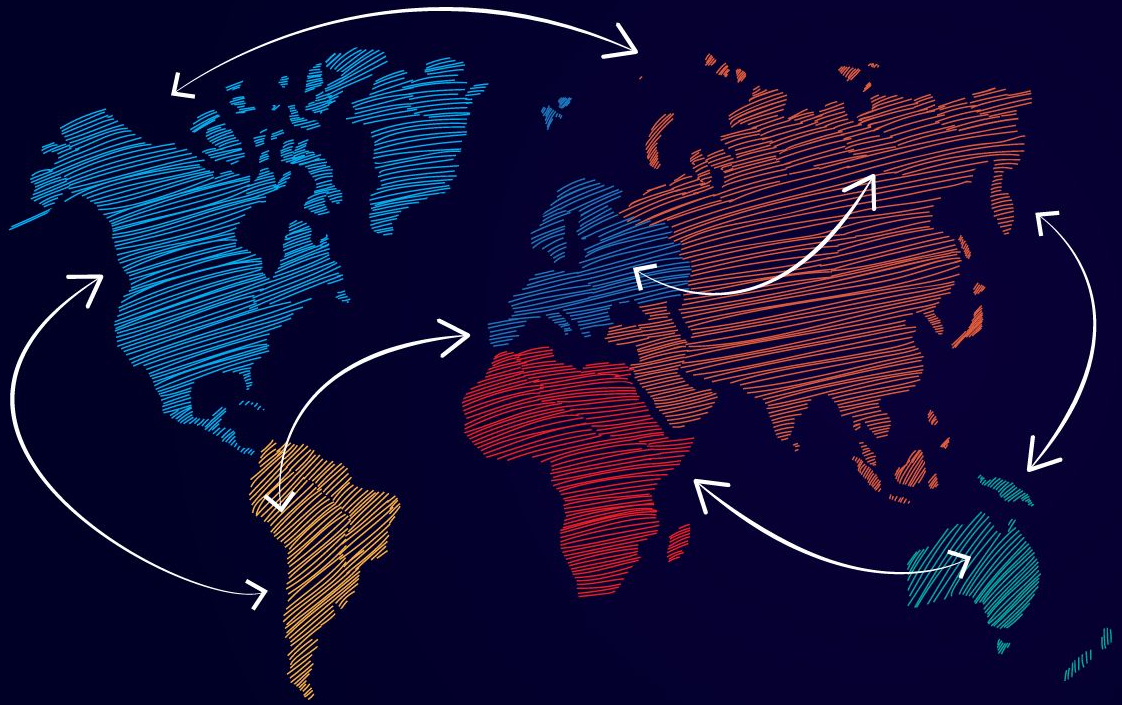
```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: reviews-route
spec:
  - route:
    - destination:
        host: reviews.prod.svc.cluster.local
        subset: v2
      weight: 25

    - destination:
        host: reviews.prod.svc.cluster.local
        subset: v1
      weight: 75
```

Istio DestinationRule

DestinationRule defines policies that apply to traffic intended for a service after routing has occurred. These rules specify configuration for load balancing, connection pool size from the sidecar, and outlier detection settings to detect and evict unhealthy hosts from the load balancing pool.

```
apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: bookinfo-ratings
spec:
  host: ratings.prod.svc.cluster.local
  trafficPolicy:
    loadBalancer:
      simple: LEAST_CONN
```

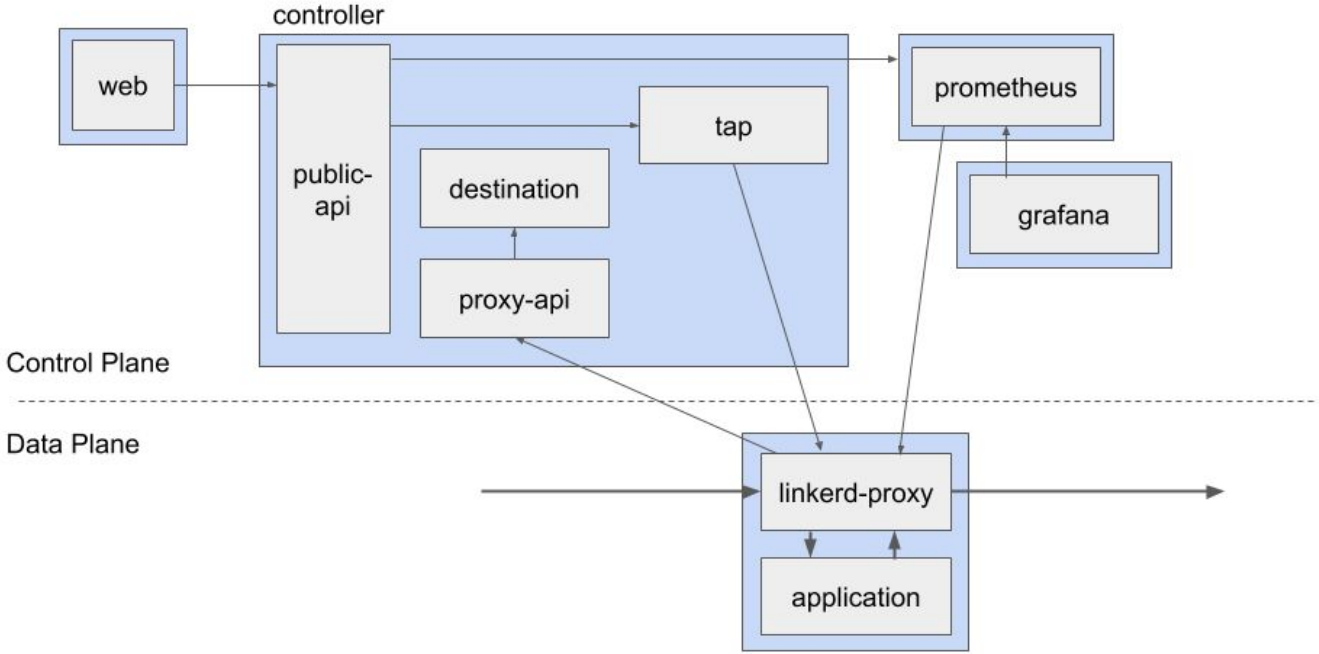


Linkerd

Linkerd

- Initially started as a network proxy (v1.0) for enabling service mesh
- Merged with Conduit to form Linkerd 2.0 in Sept 2018

Linkerd Architecture



Linkerd Architecture

- **Controller:** The controller consists of multiple containers (public-api, proxy-api, destination, tap) that provide the bulk of the control plane's functionality

- **Web:** The web deployment provides the Linkerd dashboard

- **Prometheus:** All of the metrics exposed by Linkerd are scraped via Prometheus and stored. An instance of Prometheus that has been configured to work specifically with the data that Linkerd generates is deployed

- **Grafana:** Linkerd comes with many dashboards out of the box. The Grafana component is used to render and display these dashboards. You can reach these dashboards via links in the Linkerd dashboard itself.

Linkerd Capabilities

- Linkerd’s philosophy is to be a very lightweight addition on top of existing platform
- No need to be a Platform admin to use linkerd
- Simple installation and CLI tools to get started
- Small sidecar proxy written in Rust
- Can do end-to-end encryption and automatic proxy injection
- Lacks complex routing and tracing capabilities

Linkerd Commands

Install:

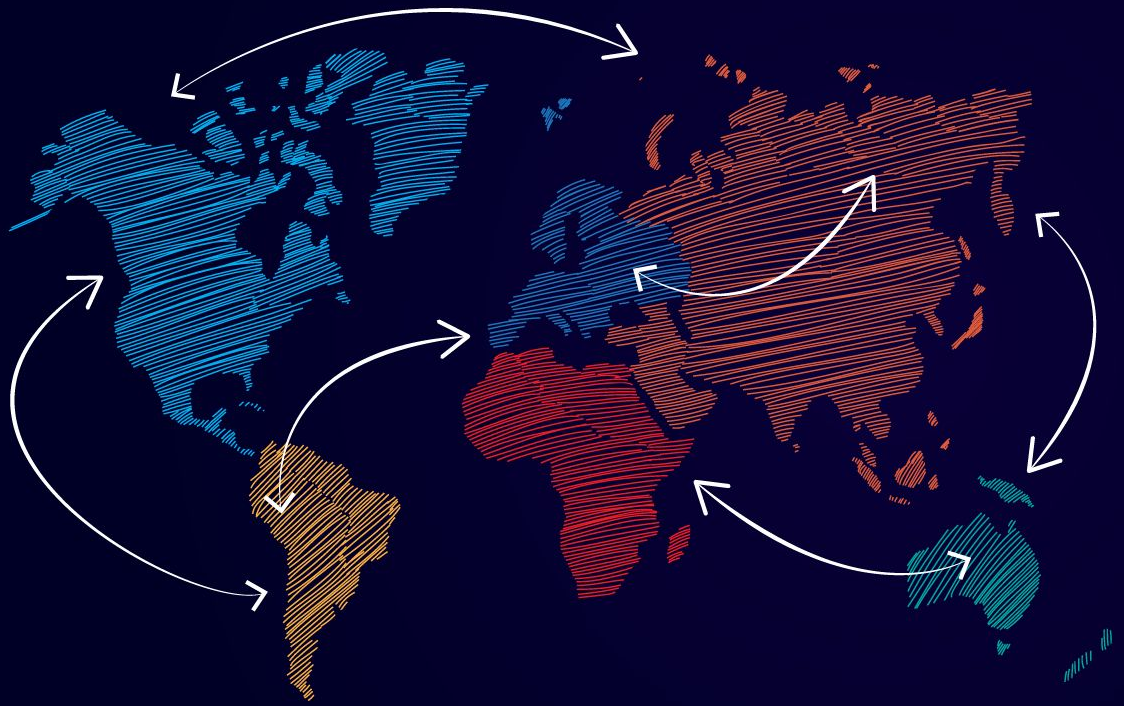
```
linkerd check --pre
linkerd install | kubectl apply -f -
```

Inject:

```
kubectl get -n emojivoto deploy -o yaml \
  | linkerd inject - \
  | kubectl apply -f -
```

Inspect:

```
linkerd -n emojivoto stat deploy
linkerd -n emojivoto top deploy
linkerd -n emojivoto tap deploy/web
```



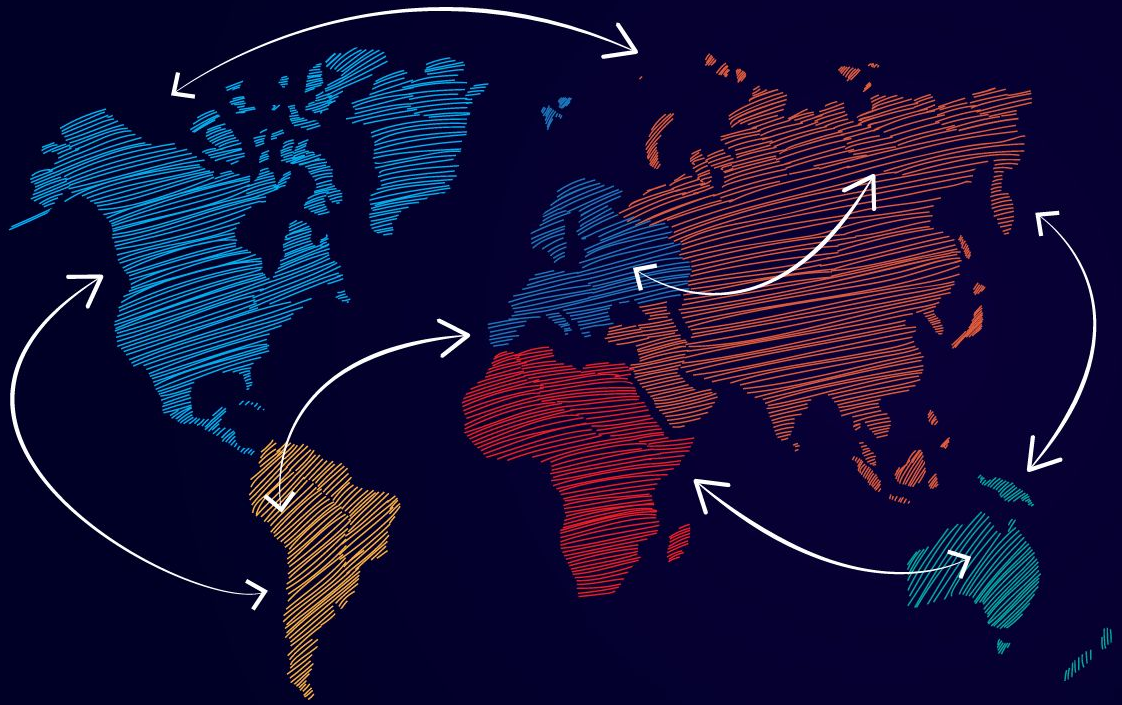
Consul Connect

Consul Connect

- Consul is a highly available and distributed service discovery and KV store
- Consul Connect augments Consul and adds Service Mesh Capabilities and was added in July 2018

Consul Connect Features

- Provides secure service-to-service communication with automatic TLS encryption and identity-based authorization.
- Uses envoy proxy sidecar as the dataplane
- Integration with Vault for certificate and secret management
- Service discovery already provided by Consul
- Useful if you want to use services outside Kubernetes as Consul can do a 2 way sync between k8s services and Consul services
- No routing features. Main focus on service discovery and Service Identity management



Conclusion

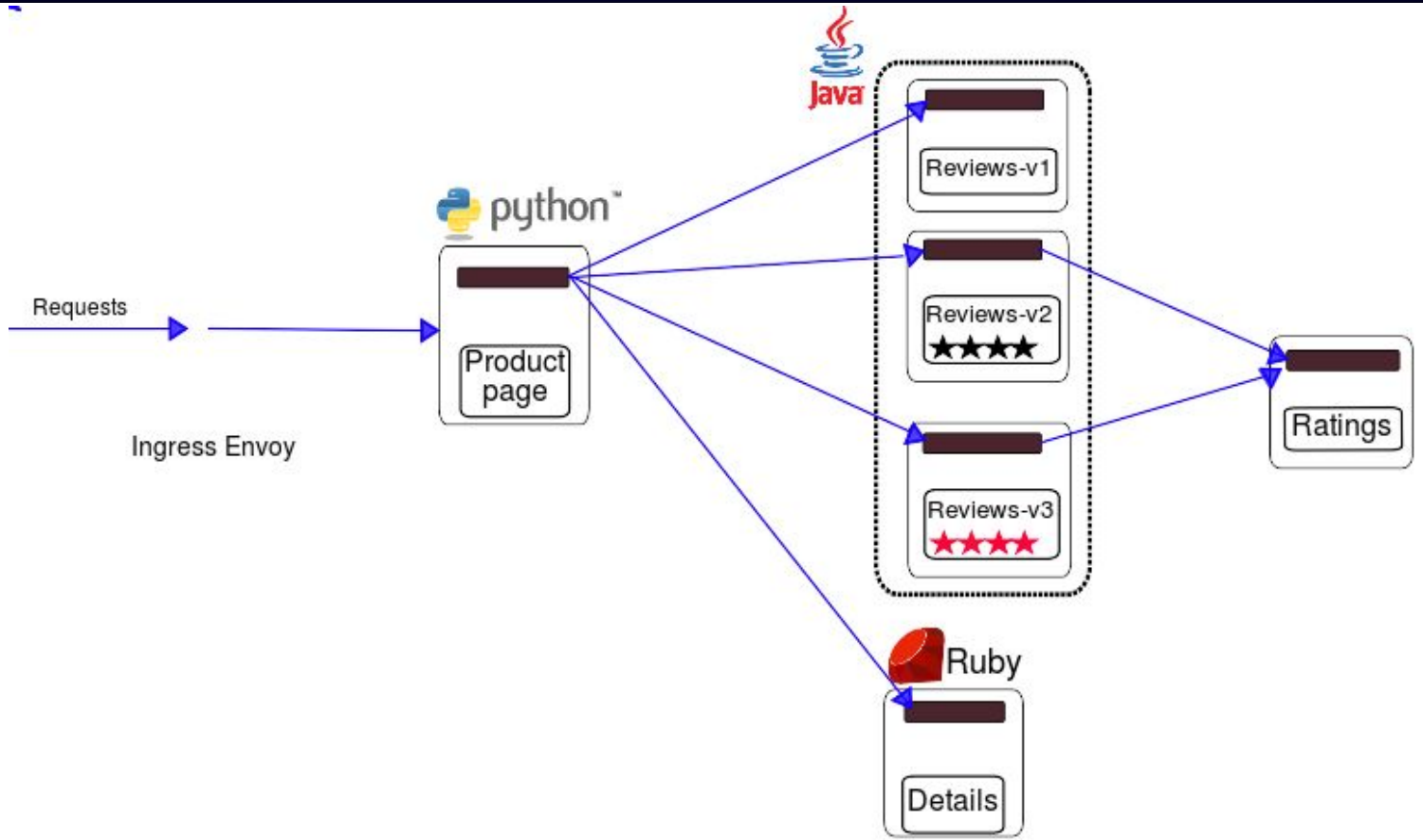
Conclusion

Feature	Istio	Linkerd	Consul Connect
Traffic Redirection (Blue/Green deployment)	Yes	No	No
Traffic Splitting (Canary deployment)	Yes	No	No
Attribute based routing	Yes	No	No
Service Identification	Yes	No	Yes
Auto Proxy Injection	Yes	Yes	Yes
Non-Admin installation	No	Yes	No
Built-in Dashboard	Yes	Yes	No
Certificate Management	Yes	No	Yes

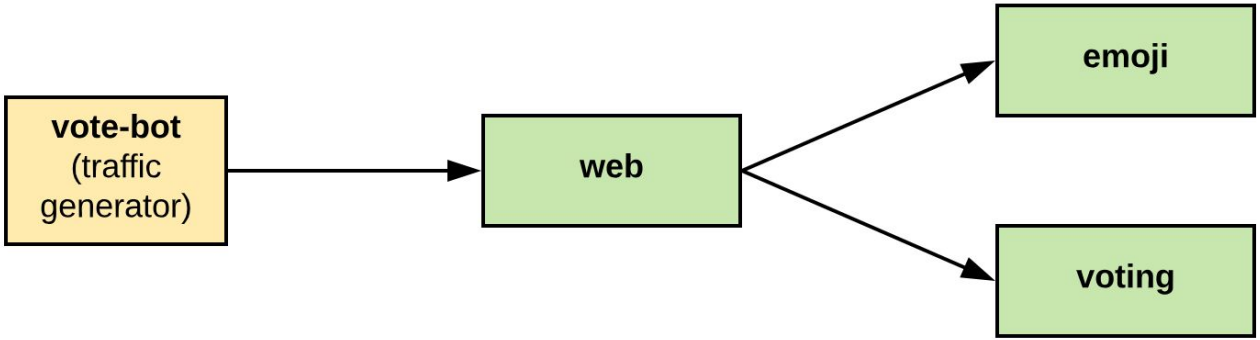
Conclusion

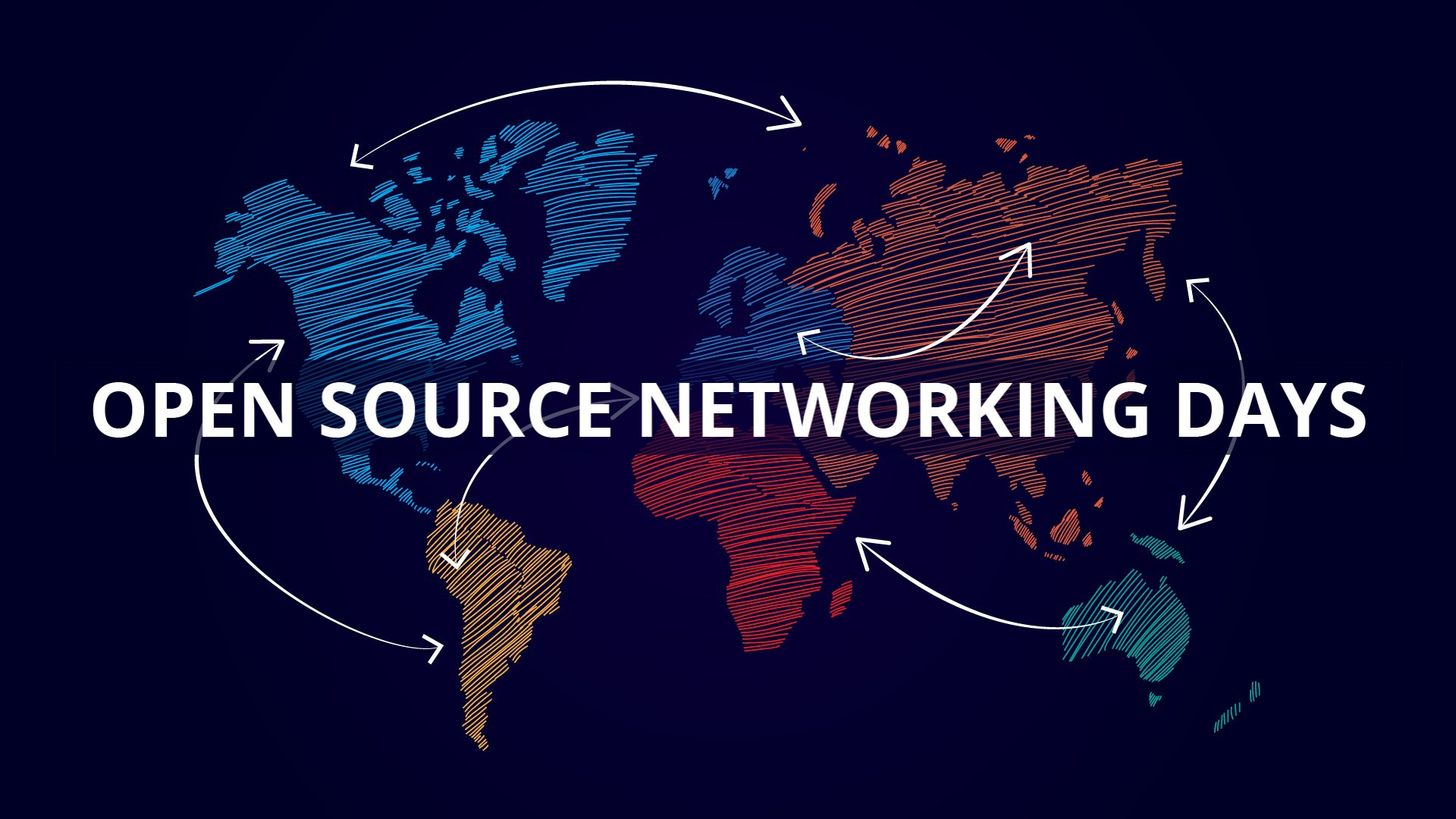
Feature	Istio	Linkerd	Consul Connect
Metrics Collection	Yes	Yes	No
Built-In Dashboard	Yes	Yes	No
TLS	Yes	Yes	Yes
External Service Support	Yes	No	Yes
Rate Limiting	Yes	No	No
Tracing	Yes	No	No

Appendix (BookInfo App)



Appendix (Emojivoto App)




OPEN SOURCE NETWORKING DAYS