What can a Service Mesh do for you?

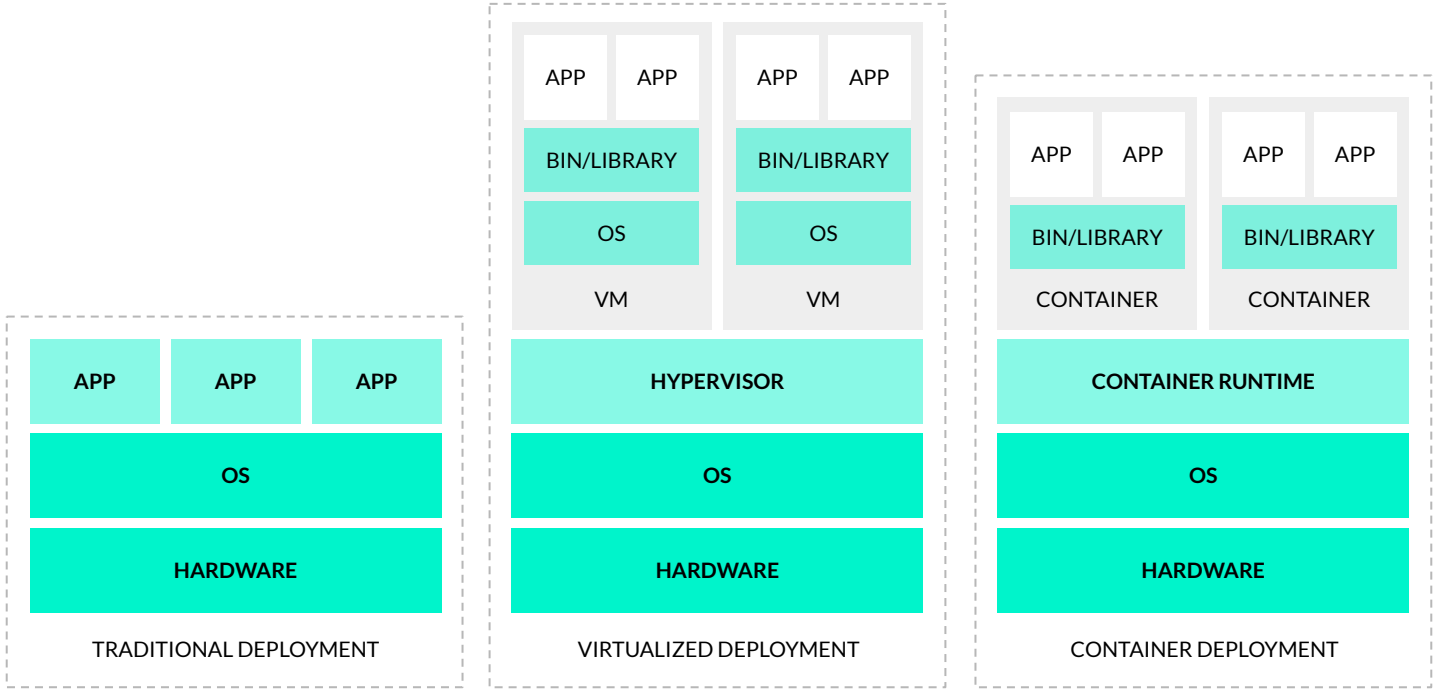
Mike Stoltz

containership

Who am I?

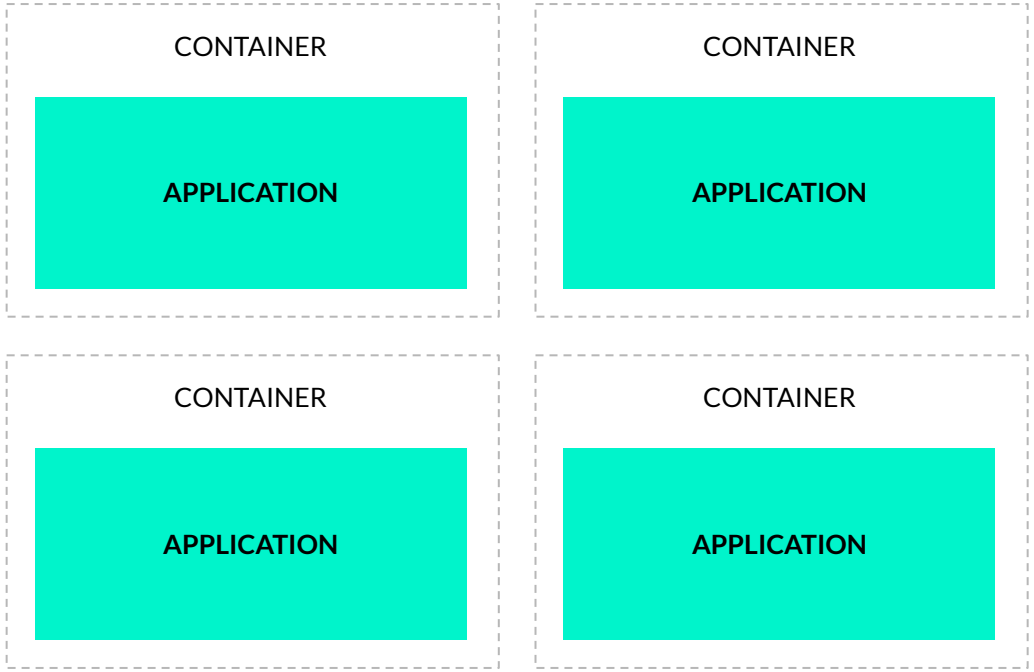
- Mike Stoltz
- Software Engineer
- Containership Inc
- 4 Years Container Orchestrators
- 3 Years Kubernetes

Virtualization History



Container Infrastructure Challenges

- Manual application scaling
- Networking & Routing
- Containers can run anywhere



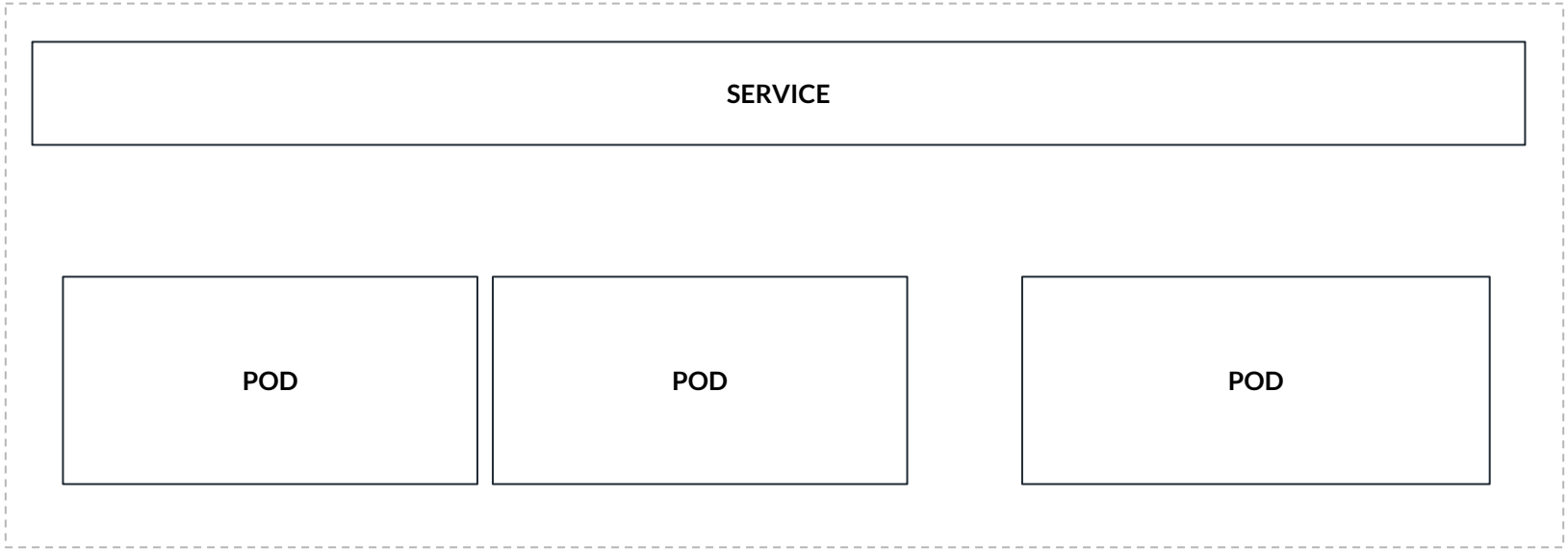
Kubernetes

- Open source container orchestrator for automating application management and deployment across clusters of hosts.

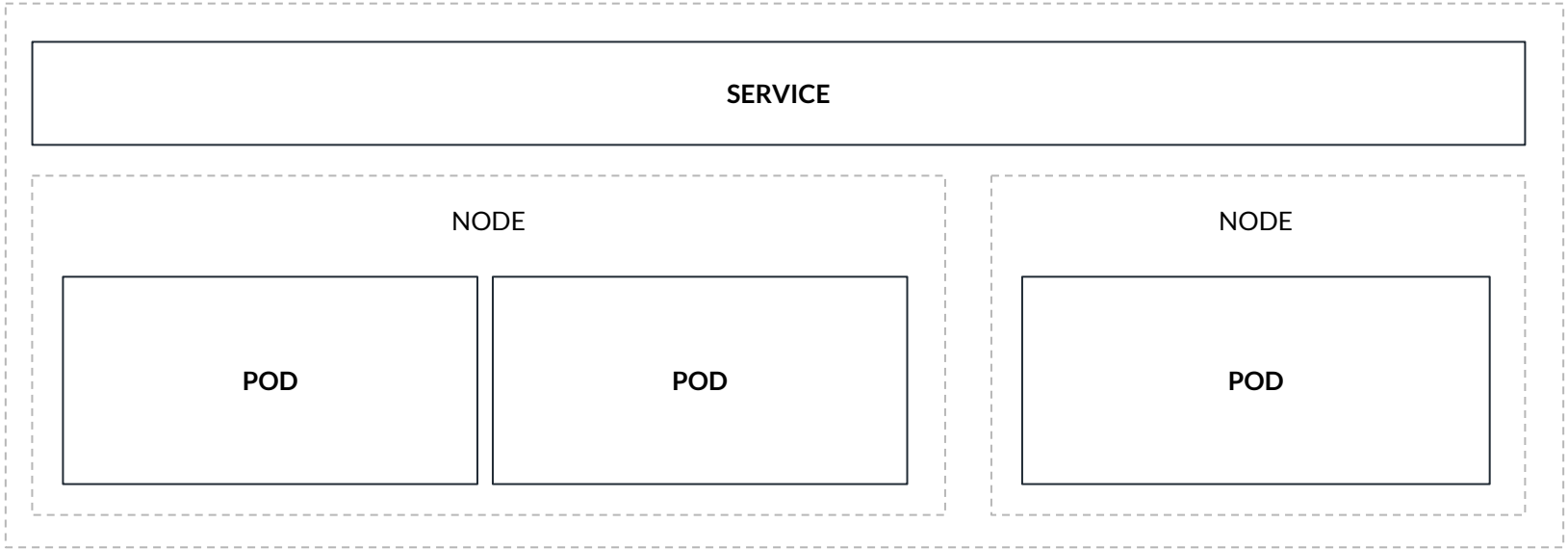
Kubernetes Terms

- **Node** - Virtual machine or physical server belonging to the cluster that containers will be scheduled on
- **Pod** - One or more containers and their environments that will share a virtual network
- **Service** - An abstraction which defines a logical set of Pods and a policy by which to access them
- **Selectors** - Labels used to match Kubernetes objects

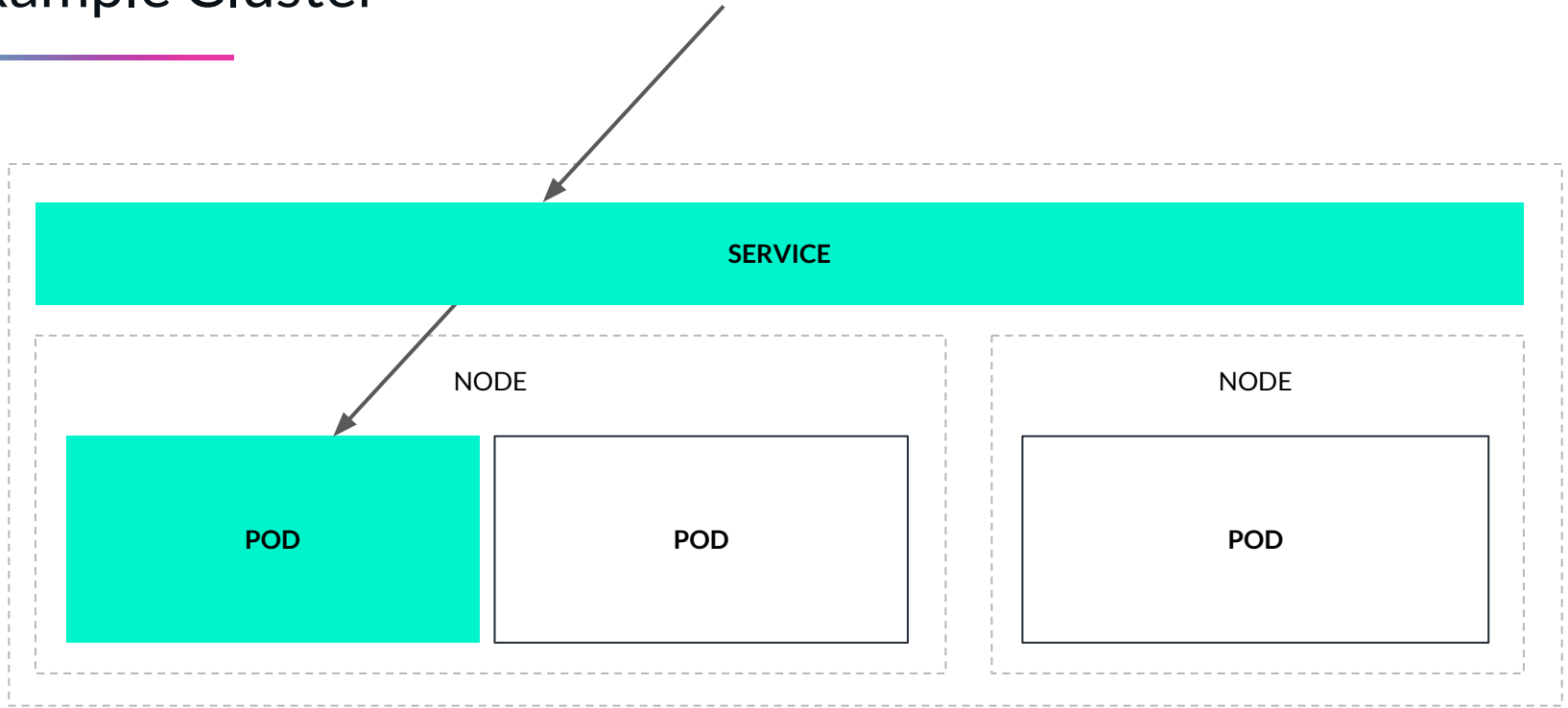
Example Cluster



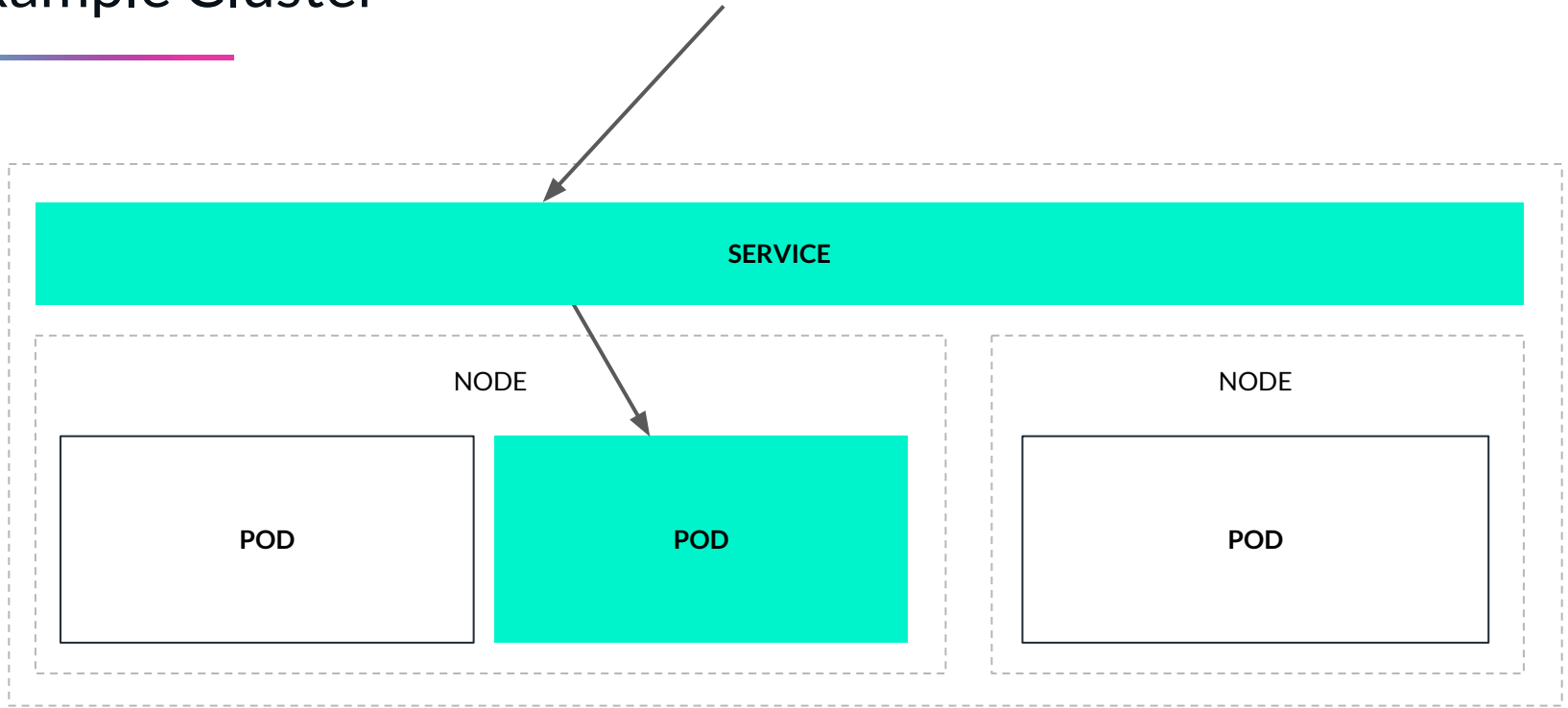
Example Cluster



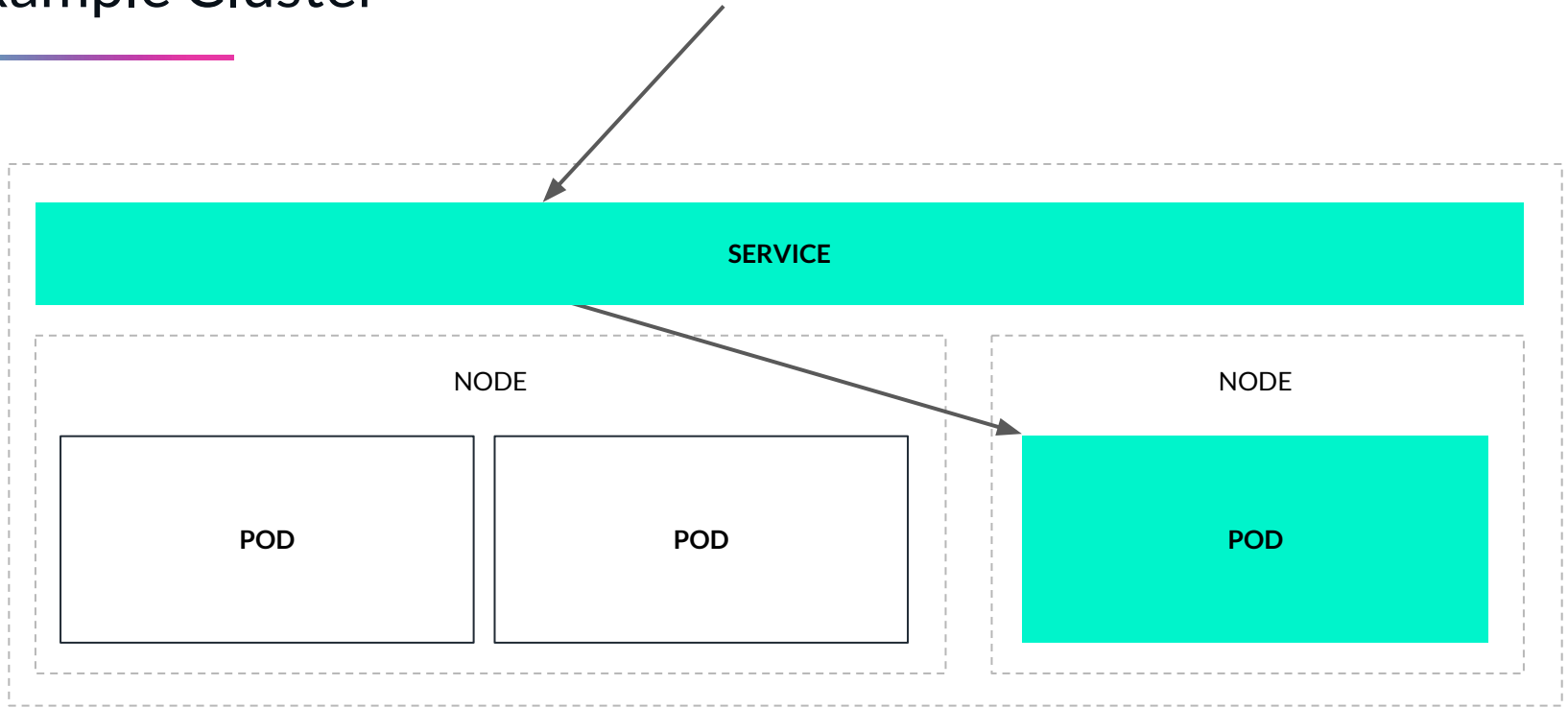
Example Cluster



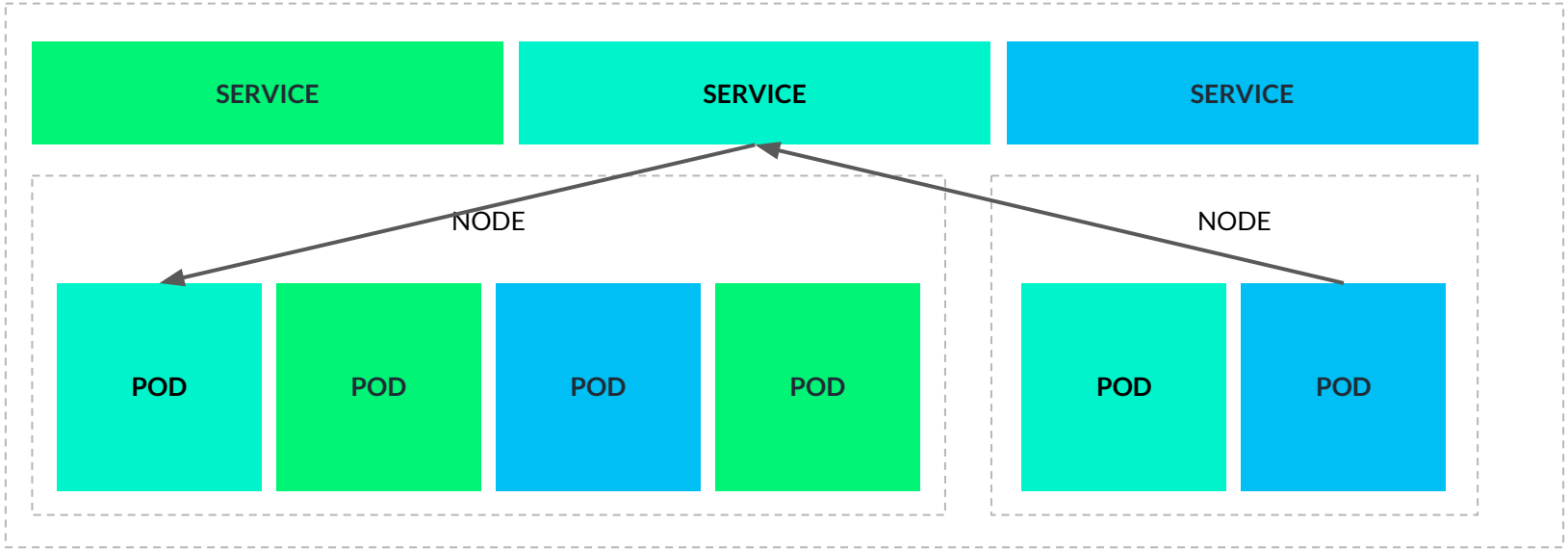
Example Cluster



Example Cluster



Example Cluster



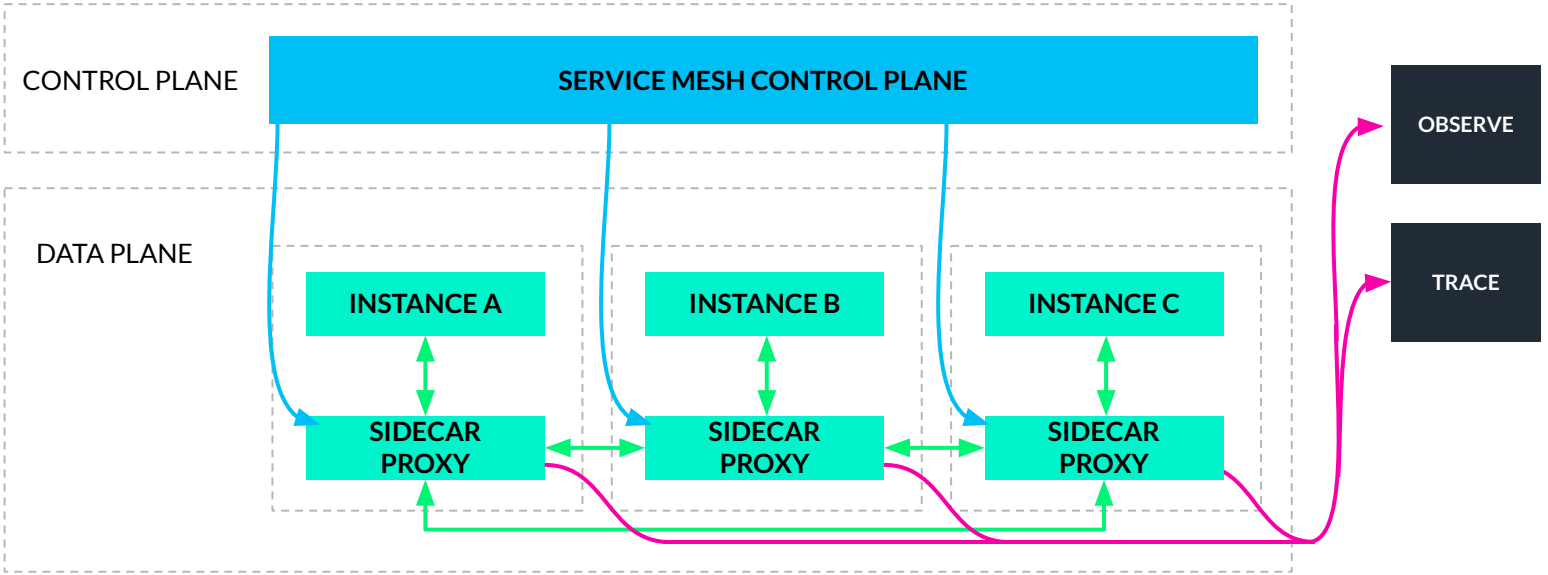
Service Mesh - Advantages

- Infrastructure layer for handling service-to-service communication
- Observability
- Tracing
- Loadbalancing (L7)
- Encryption
- Authentication
- Separation of concerns

What makes up a service mesh?

- **Control plane** - a number of “services” running as controllers that accomplish things such as metrics collection and tracing, tls asset management, injecting the data plane proxy, and service discovery
- **Data plane** - a number of sidecar proxies that run alongside every pod, routes all traffic to and from the containers it is running alongside

What makes up a service mesh?

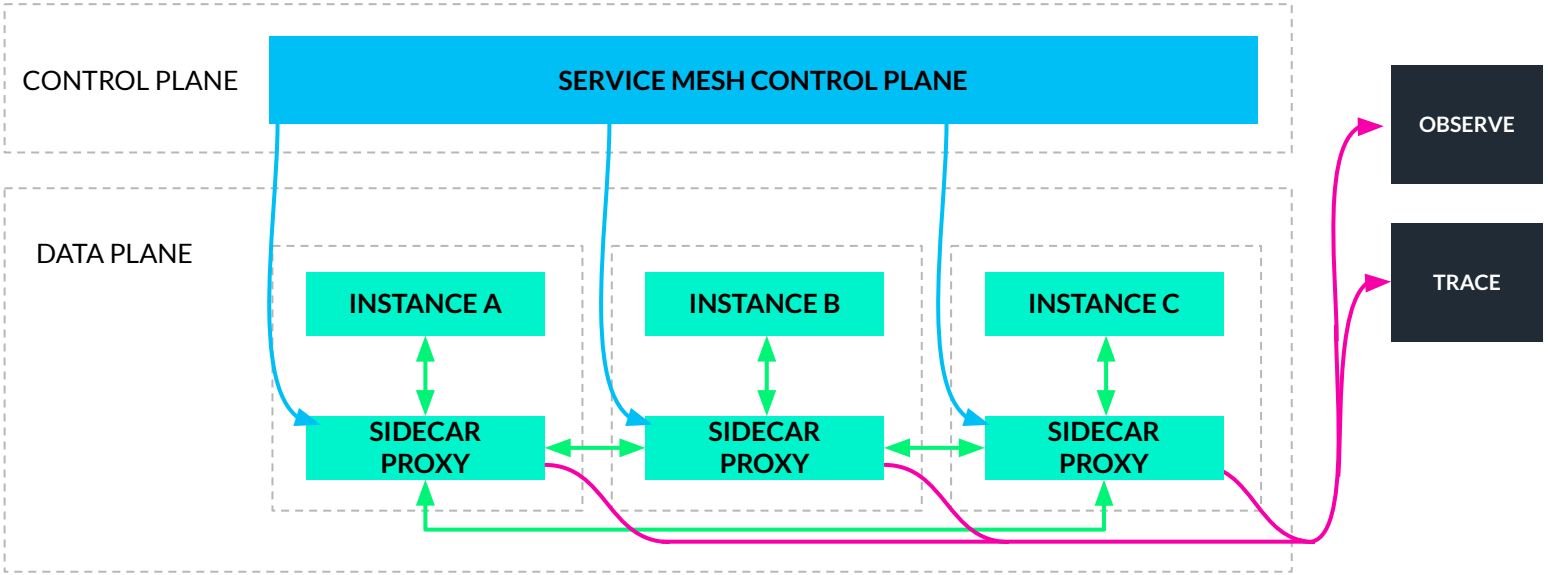


What service meshes are there?

Linkerd

Istio

What makes up a service mesh?



Linkerd2 - Installation

```
$ curl -sL https://run.linkerd.io/install | sh
```

```
$ linkerd check --pre

$ linkerd install | kubectl apply -f -
```


Linkerd2 - Installation

```
$ namespace/linkerd created
$ clusterrole.rbac.authorization.k8s.io/linkerd-linkerd-identity created
$ clusterrolebinding.rbac.authorization.k8s.io/linkerd-linkerd-identity created
$ serviceaccount/linkerd-identity created
$ clusterrole.rbac.authorization.k8s.io/linkerd-linkerd-controller created
$ clusterrolebinding.rbac.authorization.k8s.io/linkerd-linkerd-controller created
$ serviceaccount/linkerd-controller created
$ serviceaccount/linkerd-web created
```


Linkerd2 - Installation

```
$ customresourcedefinition.apiextensions.k8s.io/serviceprofiles.linkerd.io created
$ customresourcedefinition.apiextensions.k8s.io/trafficsplits.split.smi-spec.io created
$ clusterrole.rbac.authorization.k8s.io/linkerd-linkerd-prometheus created
$ clusterrolebinding.rbac.authorization.k8s.io/linkerd-linkerd-prometheus created
$ serviceaccount/linkerd-prometheus created
$ serviceaccount/linkerd-grafana created
$ clusterrole.rbac.authorization.k8s.io/linkerd-linkerd-proxy-injector created
$ clusterrolebinding.rbac.authorization.k8s.io/linkerd-linkerd-proxy-injector created
```


Linkerd2 - Installation

```
$ serviceaccount/linkerd-proxy-injector created
$ secret/linkerd-proxy-injector-tls created
$ mutatingwebhookconfiguration.admissionregistration.k8s.io/linkerd-proxy-injector-webhoo
$ clusterrole.rbac.authorization.k8s.io/linkerd-linkerd-sp-validator created
$ clusterrolebinding.rbac.authorization.k8s.io/linkerd-linkerd-sp-validator created
$ serviceaccount/linkerd-sp-validator created
$ secret/linkerd-sp-validator-tls created
$ validatingwebhookconfiguration.admissionregistration.k8s.io/linkerd-sp-validator-webhoo
```


Linkerd2 - Installation

```
$ clusterrole.rbac.authorization.k8s.io/linkerd-linkerd-tap created
$ clusterrolebinding.rbac.authorization.k8s.io/linkerd-linkerd-tap created
$ serviceaccount/linkerd-tap created
$ podsecuritypolicy.policy/linkerd-linkerd-control-plane created
$ role.rbac.authorization.k8s.io/linkerd-psp created
$ rolebinding.rbac.authorization.k8s.io/linkerd-psp created
$ configmap/linkerd-config created
$ secret/linkerd-identity-issuer created
```


Linkerd2 - Installation

```
$ service/linkerd-identity created
$ deployment.extensions/linkerd-identity created
$ service/linkerd-controller-api created
$ service/linkerd-destination created
$ deployment.extensions/linkerd-controller created
$ service/linkerd-web created
$ deployment.extensions/linkerd-web created
$ configmap/linkerd-prometheus-config created
```


Linkerd2 - Installation

```
$ service/linkerd-prometheus created
$ deployment.extensions/linkerd-prometheus created
$ configmap/linkerd-grafana-config created
$ service/linkerd-grafana created
$ deployment.extensions/linkerd-grafana created
$ deployment.apps/linkerd-proxy-injector created
$ service/linkerd-proxy-injector created
$ service/linkerd-sp-validator created
```

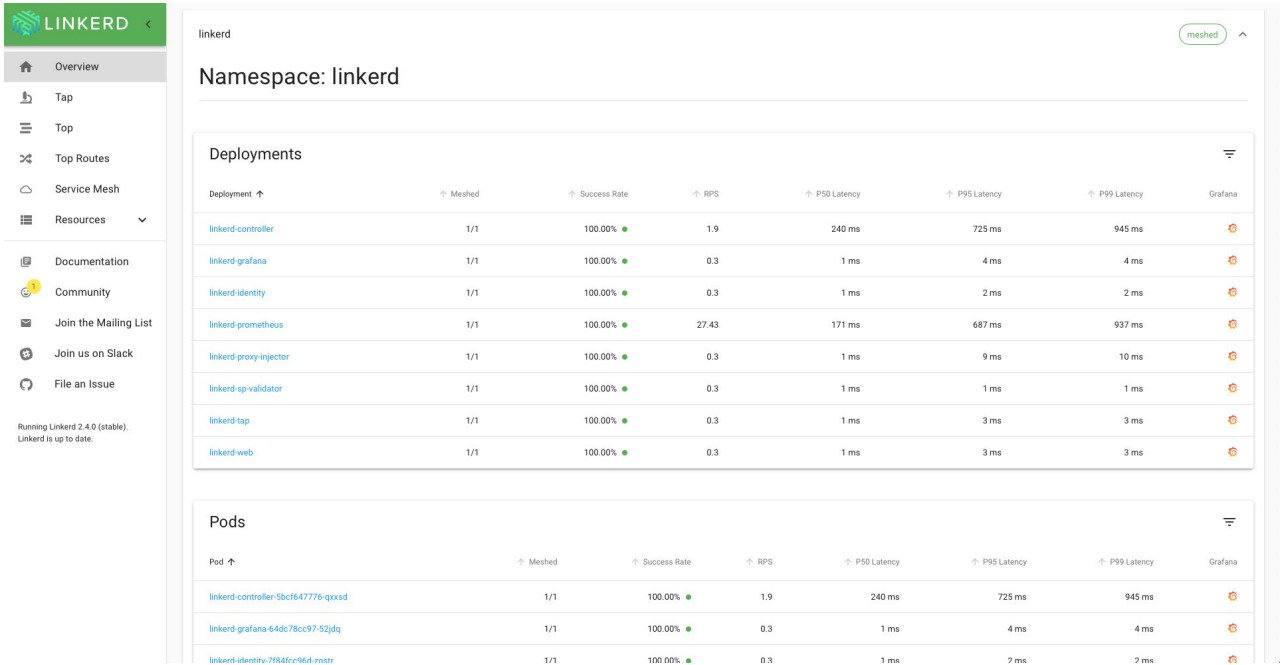

Linkerd2 - Installation

```
$ deployment.extensions/linkerd-sp-validator created

$ service/linkerd-tap created

$ deployment.extensions/linkerd-tap created
```


Linkerd2 - Installation

```
$   kubectl get pod -n linkerd

NAME                                READY   STATUS      RESTARTS    AGE
Linkerd-controller-5bcf647776-qxxsd   3/3   Running      0          17m
Linkerd-grafana-64dc78cc97-52jdq      2/2   Running      0          16m
Linkerd-identity-7f84fcc96d-zqstr     2/2   Running      0          17m
Linkerd-prometheus-7df8689c4f-x64lr   2/2   Running      0          17m
Linkerd-proxy-injector-876d98c99-fb7fp  2/2 Running      0          16m
Linkerd-sp-validator-d6b4496d6-zjfv   2/2   Running      0          16m
linkerd-tap-7df59c45b7-ggbsv          2/2   Running      0          16m
linkerd-web-7ff6bbd984-fp5gc          2/2   Running      0          17m
```


Linkerd2 - Dashboard Demo



Linkerd2

- Mutual TLS
- Automatic proxy injection
- L7 Loadbalancing

- Retries
- Timeouts
- Traffic splitting (not covered)

Linkerd2 - Service Profile

- Kubernetes CRD
- Provide extra information about a meshed service
- Routes
- Per route settings
- Retries
- Timeouts

Linkerd2 - Service Profile

```
spec:
 retryBudget:
  retryRatio: 0.2
  minRetriesPerSecond: 10
  ttl: 10s
 routes:
 - name: GET /api/annotations
  condition:
   method: GET
   pathRegex: /api/annotations
  isRetryable: true
```

```
spec:
 routes:
 - condition:
    method: HEAD
    pathRegex: /authors/[^/]*\.json
  name: HEAD /authors/{id}.json
  timeout: 300ms
```


Istio

- All covered Linkerd functionality
- Additional functionality
- Much more complex
- All new resource types
- Installed using Helm package manager

Istio - Custom Resources

- **Virtual Service** - a virtual service sets the default version and route logic for a microservice
- **Destination Rule** - defines policies that apply to traffic intended for a service after routing has occurred
- **Ingress Gateway** - describes a load balancer operating at the edge of the mesh receiving incoming or outgoing HTTP/TCP connections. The specification describes a set of ports that should be exposed, the type of protocol to use, SNI configuration for the load balancer
- **Service Entry** - used for stricter control of outbound requests to external resources, outside of the mesh and cluster

Istio - Virtual Service

```
spec:
 gateways:
 - bookinfo-gateway
 hosts:
 - details
 Http:
- match:
  - headers:
      end-user:
       exact: sam
 - route:
  - destination:
      host: details
      subset: v1
```

```
hosts:
- reviews
http:
- route:
 - destination:
     host: reviews
     subset: v1
   weight: 50
 - destination:
     host: reviews
     subset: v3
   weight: 50
 timeout: 0.5s
```


Downfalls

- Latency
- Resource overhead
- Proxy protocol support
- Increased complexity
- Application code changes for advanced tracing

Final Notes

- Kubernetes is not required for service meshes
- Observability is key
- Meshes add security
- Linkerd2 and Istio are both great solution
- Linkerd2 has small barrier to entry
- Istio allows for complex routing and settings

Thank you!

Mike Stoltz

 @michael_stoltz

 mike stoltz