ORACLE

Cloud Native Labs

Practical Service Mesh

A Quick Tour Through Some Real Use Cases

Jesse Butler Cloud Native Advocate, Oracle Cloud Infrastructure. @jlb13

#OracleCloudNative

cloudnative.oracle.com

The Inevitable First Slide: What is a Service Mesh?

Connect, secure, control and observe services at scale, often requiring no service code modification

Though many options exist, Linkerd and Istio are the two main projects

Service Mesh 101

- Infrastructure layer for controlling and monitoring service-to-service traffic
- Data plane deployed alongside application services, control plane used to manage the mesh
- Greatly simplifies service implementation offering transparent service discovery, automated retries, timeouts and more

Service Mesh is Not an API Gateway

API Gateways deal with north-south traffic, inbound to your cluster

Service Mesh is concerned with east-west traffic, between your services within your cluster

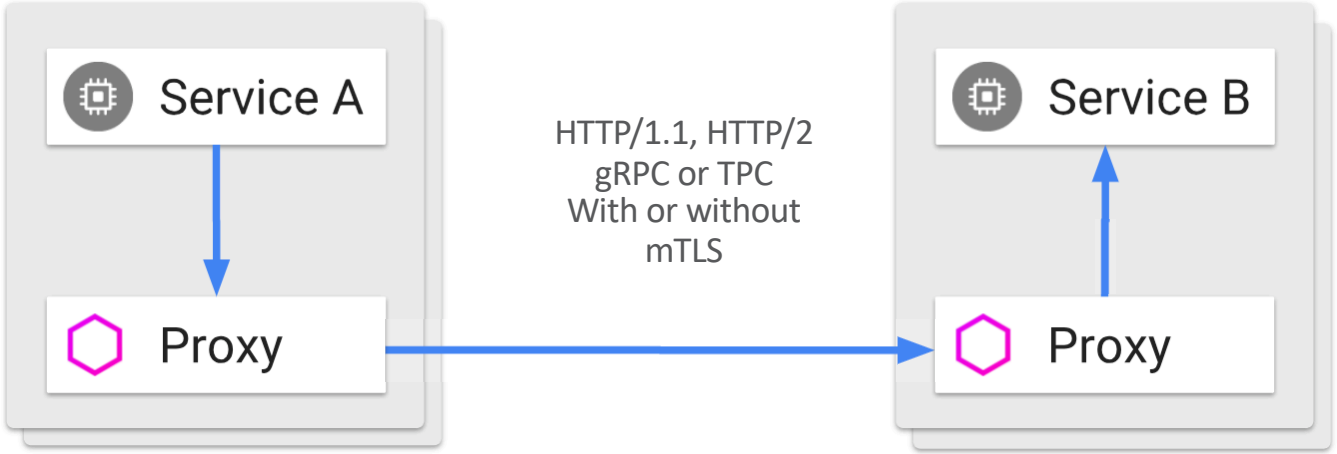
Service Mesh Architecture

- Both Istio and Linkerd use a sidecar pattern, adding a proxy container for each pod added to the mesh
- Each proxy instance manages traffic for its pod, and is fully configurable
- This vantagepoint is what gives a service mesh its power – it sees and knows all

Sidecar Proxy

Sidecar Proxy

Sidecar Proxy



Observability

- Metrics
  - Aggregate data regarding the behavior of a thing over time
- Tracing
  - Instrumentation which provides an instance of an action, traversing the entire stack
- Logging
  - Developer breadcrumbs we leave to give context for a certain code path

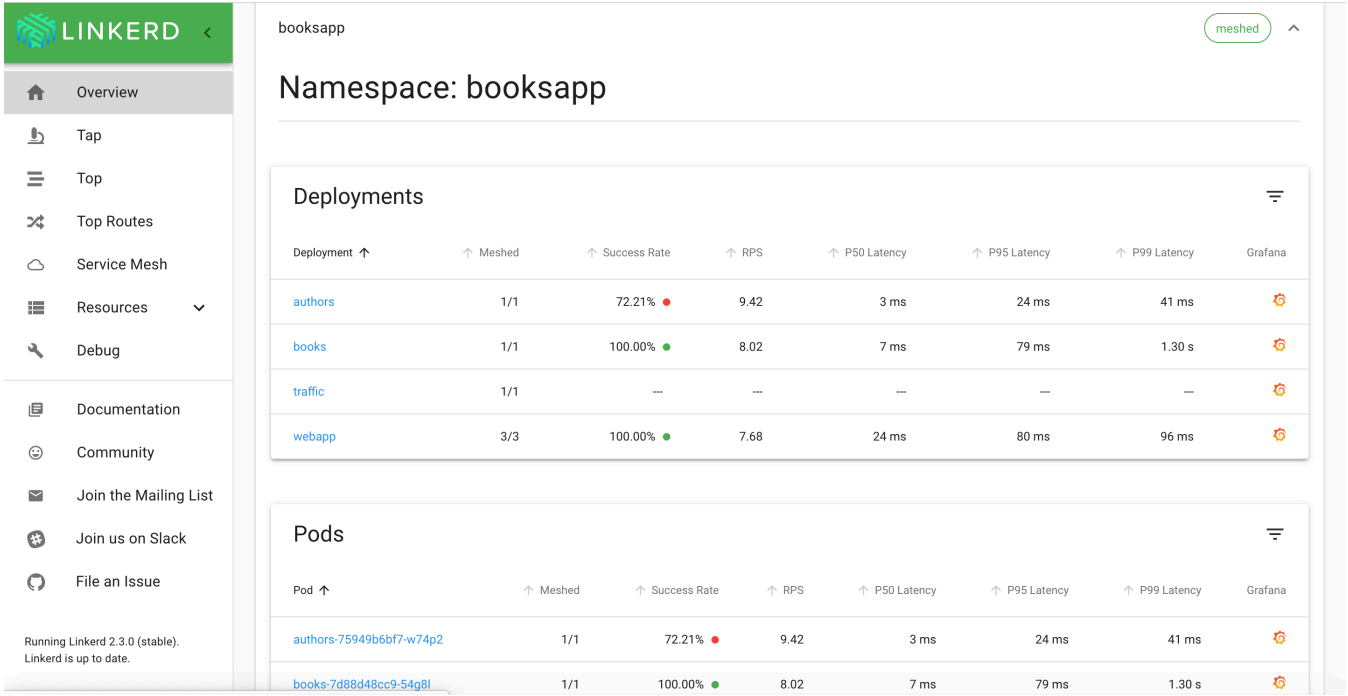
Triaging Issues

- Metrics are instrumented and scraped for analytic use
- Traces are implemented on a per-span basis at points of interest
- Logs are specific, and a gift we give our future selves… treat yourself

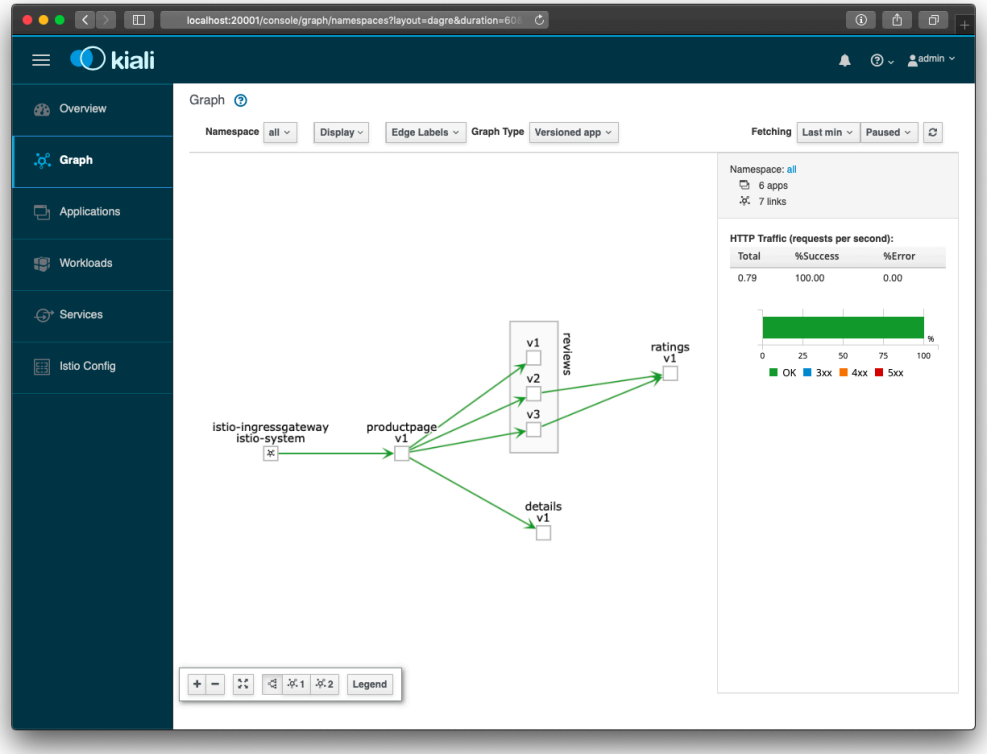
Service Mesh Brings Observability Gifts

- All traffic in the mesh is routed through the proxies
- Boundary tracing, on-wire traffic, calls and status are all obvious in a mesh
- Metrics and traces can be taken for free, with no modifications to code
- Most issues can be triaged with this information

Linkerd Dashboard



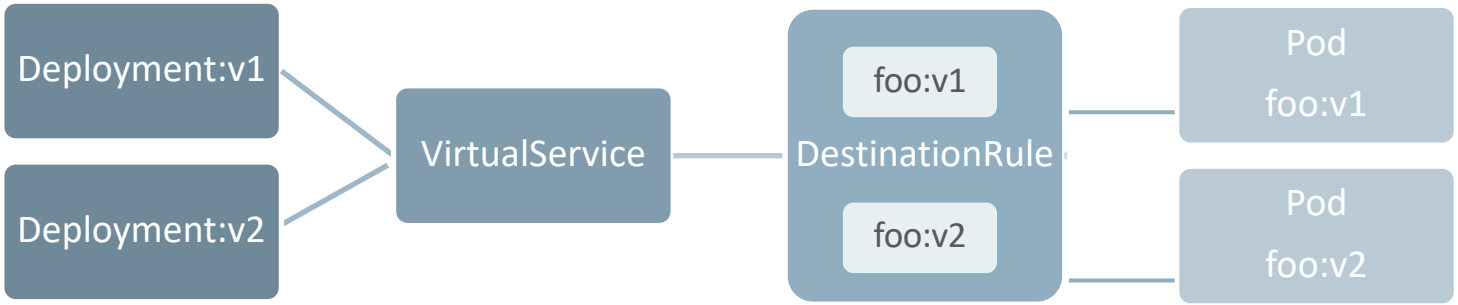
Kiali, the Istio Dashboard



Traffic Management

- Proxy instances provide a traffic shifting capabilities
- We can configure proxies based upon knowledge of our services
- Through proxy configuration we have intelligent routing of our cluster traffic

Traffic Management Details with Istio



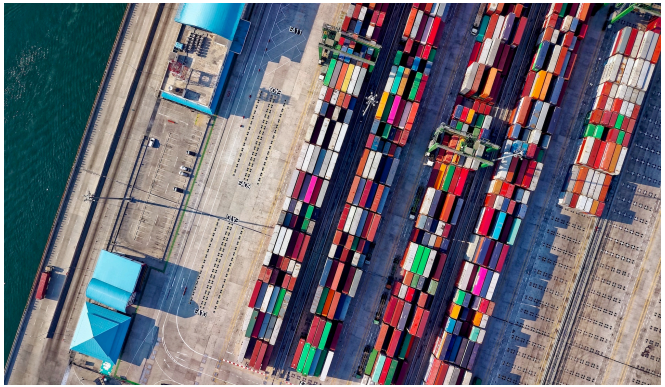
- ‘foo’ deployed services routed through ‘foo’ VirtualService
- DestinationRules for ‘foo:v1’ and ‘foo:v2’ pods, with weights

Speak Kubernetes at your Kubernetes

```yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: simple
spec:
  hosts:
    - simple
  http:
  - route:
    - destination:
        host: simple
        subset: v1
      weight: 50
    - destination:
        host: simple
        subset: v2
      weight: 50
---
  apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: simple
spec:
  host: simple
  subsets:
  - name: v1
    labels:
      version: v1
  - name: v2
    labels:
      version: v2
```

Leveraging Traffic Shifting

- Manage and shift traffic via configuration
- Take advantage of zero-downtime changes in routing between versions
- We can automate deployments of any kind
  - Canary deployments
  - Blue/Green deployments
  - Whatever we want



Traffic Mirroring and Dark Launches

- Traffic shifting, but 100% of production traffic goes to production services
- Mirror as much or as little traffic to other services in the cluster
- These routes can be intelligently filtered
  - Test automation
  - Beta users
  - That one dev who keeps bugging you…

Test in Production Safely

- Isolate traffic as required
- Deploy test candidates
- Mirror real production data to them, shift their responses to test fixtures
- Meanwhile, prod keeps humming along

Testing

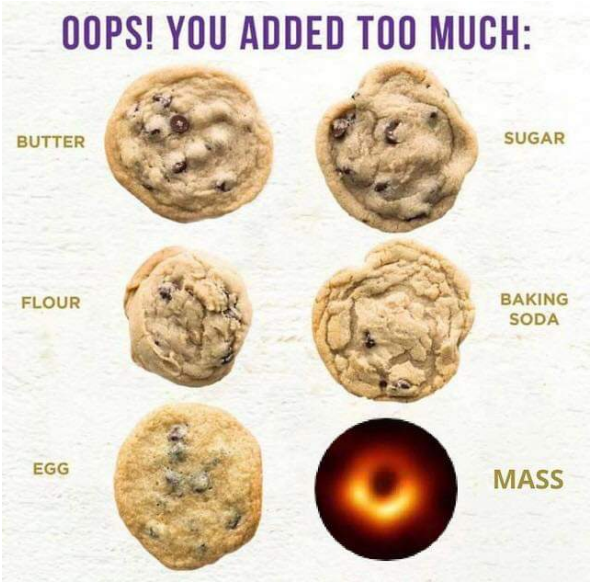
- Core mesh features: retries, timeouts, circuit breakers
- Through the same proxy configuration we can inject latency trivially as well
- Modify on-wire data including message bodies and header information

Welcome to Microservices Fight Club

- Inject faults by modifying reply status or mutate parameter data
- Inject latency to test resilience and response
- Redirect traffic to API stubs / mocks
- Use traffic shifting/mirroring to target test traffic as needed
- Let your imagination run

Caution

- It is trivial to modify on-wire data via mesh configuration
- Service protocols could mutate over time through configuration changes, with no visibility in source code
- This is a recipe for disaster, and would repeat mistakes long-ago learned from



Security

- Deploying services in containers requires careful provisioning, build and deployment practices
- There are options to leverage in both CI/CD and registry scanning
- Once services are deployed in the wild, they are on their own

Security

- Istio and Linkerd are capable of creating a zero-touch, zero-trust network
- Services within your cluster authenticate via the mesh
- Leveraging mTLS, the cluster is transparently hardened and protected from many types of attacks

ORACLE

Cloud Native Labs

Thanks!

cloudnative.oracle.com

@jlb13