Service Mesh Interface

Brendan Burns – QCon New York – 2019
The Service Mesh Landscape
The problem for users
The problem for users – Adoption timeline

- Excitement
- Adoption
- Production
Problem for users - Complexity
The problem for the ecosystem
The solution? Moar Abstraction!
Service Mesh Interface: Community

https://smi-spec.io
Service Mesh Interface: Goals

- Isolate concepts from implementation
- Provide the “core concepts” of service mesh
- Release and iterate
- Build a community around Service Mesh as a concept
This isn't a new pattern...

- Open Container Image
- Container Network Interface
- Container Storage Interface
- Storage Volumes
- Ingress
- NetworkPolicy
- ...

...
Good reasons for this approach

- Users need concepts, not implementation
- Tool vendors need abstraction, not specialization
- Implementors need isolation from users
Service Mesh Interface: Routes

apiVersion: v1beta1
kind: HTTPRouteGroup
metadata:
  name: api-route
matches:
- name: api
  pathRegex: /api
  methods:
    - GET
Service Mesh Interface: Routes

apiVersion: v1beta1
kind: TCPRoute
metadata:
  name: my-db-route
Service Mesh Interface: TrafficTarget

kind: TrafficTarget
apiVersion: access.smi-spec.io/v1alpha1
metadata:
    name: example-target
destination:
    # destination spec here
specs:
    # route spec here
sources:
    # source spec(s) here
Service Mesh Interface: Destinations

...  
destination:

  # This selects a set of Pods
  kind: ServiceAccount
  name: my-api-impl
  # This defines the traffic
  port: 8080

...
Service Mesh Interface: TrafficTarget

kind: TrafficTarget
apiVersion: access.smi-spec.io/v1alpha1
metadata:
  name: example-target
destination:
  # destination spec here
specs:
  # route spec here
sources:
  # source spec(s) here
Service Mesh Interface: Routes

...  

# This selects a set of paths
specs:
- kind: HTTPRouteGroup
  name: api-route
  matches:
    - api

...
Service Mesh Interface: TrafficTarget

kind: TrafficTarget
apiVersion: access.smi-spec.io/v1alpha1
metadata:
    name: example-target
destination:
    # destination spec here
specs:
    # route spec here
sources:
    # source spec(s) here
Service Mesh Interface: Sources

...

# This identifies the allowed sources
sources:
  # This selects a set of Pods
- kind: ServiceAccount
  name: my-api-callers

...
Putting it all together...

ServiceAccount: my-api-callers

Sources

SMI TrafficTarget

Destination

ServiceAccount: my-api-impl

SMI HTTPRouteGroup

https://service:8080/api
Service Mesh Interface: TrafficSplit

kind: TrafficSplit
apiVersion: split.smi-spec.io/v1alpha1
metadata:
  name: one-percent-experiment
spec:
  backends:
    - service: experiment
      weight: 1
    - service: canary
      weight: 10
    - service: production
      weight: 100
Service Mesh Interface – Traffic Split

SMI
TrafficSplit
my-experiment

my-experiment
Service

Kubernetes Services

experiment
Service

canary
Service

production
Service
Service Mesh Interface - TrafficMetrics

kind: TrafficMetrics
...
resource:
  name: my-pod-asdae
  kind: Pod
edge:
  ...
timestamp: 2019-06-26T12:00:00
window: 30s
metrics:
  ...
Service Mesh Interface - TrafficMetrics

# all in-bound traffic
direction: to
resource: {}

# all out-bound traffic to Pod foobar
direction: from
resource:
  name: foobar
  kind: Pod
Service Mesh Interface - TrafficMetrics

# all in-bound traffic from a Service edge:

direction: to

resource:
    name: my-service
    kind: Service
Service Mesh Interface - TrafficMetrics

... metrics:
- name: p99_response_latency
  unit: seconds
  value: 987m
- name: p90_response_latency
  unit: seconds
  value: 250m

...
Service Mesh Interface – TrafficMetrics Overview

Prometheus, etc. → traffic.metrics.k8s.io → Kubernetes API Server → Traffic Metrics Server

metrics scrape/push
Concerns: Lowest Common Denominators
Service Mesh Interface: Approach to iteration
Service Mesh Interface: Iteration plan.

Start with the basics → Lots of custom extensions → Adopt the common extensions to v+1
Service Mesh Interface: State of the art.

Implementations:
- Consul
- LinkerD
- Istio

Tooling:
- Flagger (WeaveWorks)
- Rio (Rancher)
- ...
Service Mesh Interface: Futures

- Come and join us!
- https://smi-spec.io
- https://github.com/deislabs/smi-spec
- https://github.com/deislabs/smi-sdk-go
- https://github.com/weaveworks/flagger/blob/master/docs/gitbook/tutorials/flagger-smi-istio.md
- https://github.com/hashicorp/microsoft-smi-webinar
Questions?