devfest

Secure, Reliable, and Observable Service Mesh on Microservices-based Architecture using Anthos Mesh

Google Developer Groups

Hello World!

Ananda Dwi Rahmawati

- Cloud & DevOps Engineer, Singapore
- Google Developer Expert Cloud - Modern Architecture
- Master of Computer Science - University of Texas at Austin
- https://linktr.ee/misskecupbung

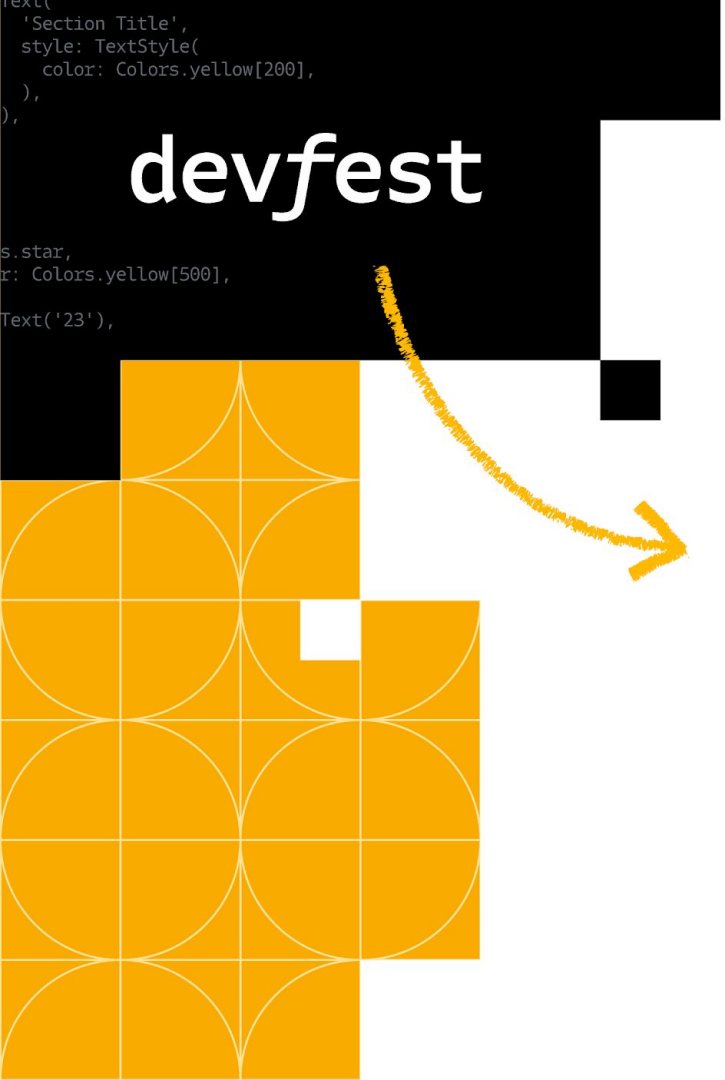
Today's Agenda

Challenges in Microservices

Service Mesh – What and Why?

Anthos Service Mesh

Demo

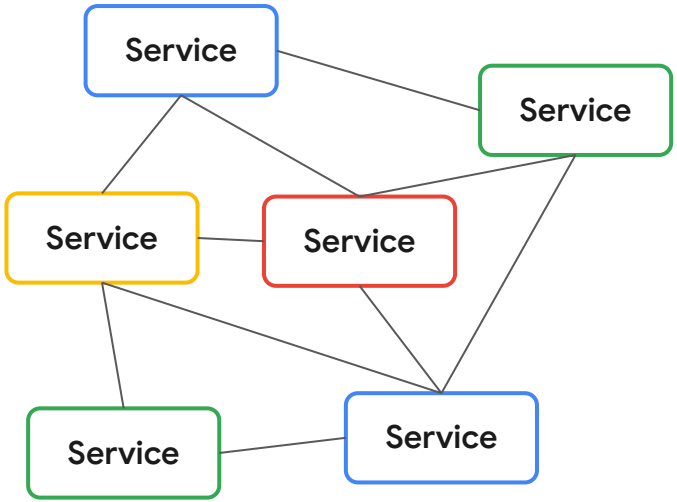


Google Developer Groups

Challenges

Challenges

- Interaction between services
- Traffic Management at each service endpoint
- Communication security
- Timeout and Communication failures



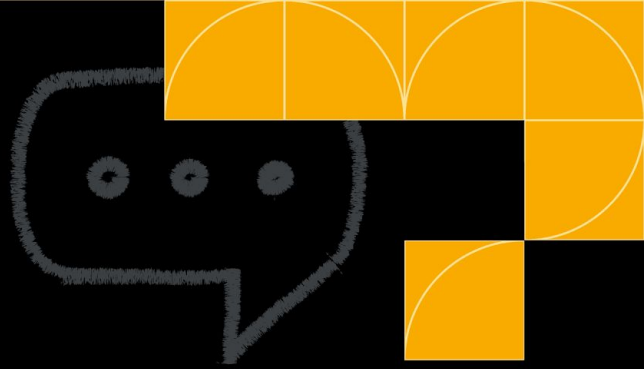
Services need to communicate with each other

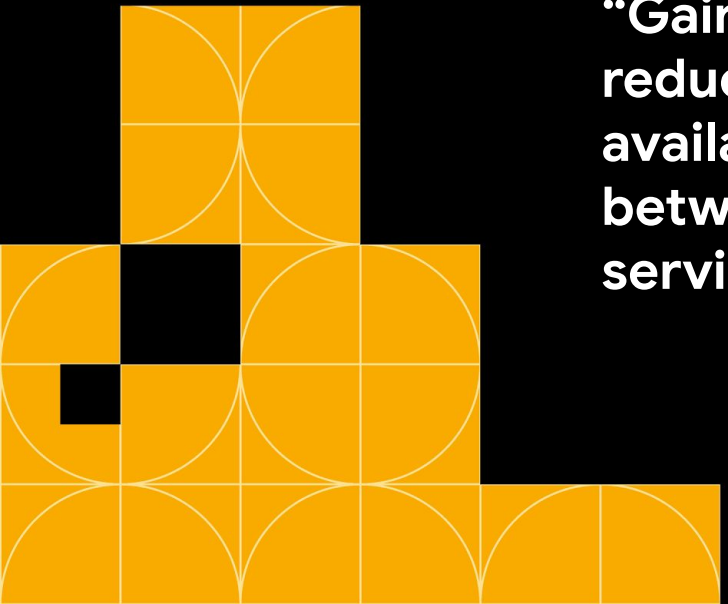
```
Text(
  'Simple Statement or URL',
  style: TextStyle(
    color: Colors.yellow[200],
  ),
),
```

devfest

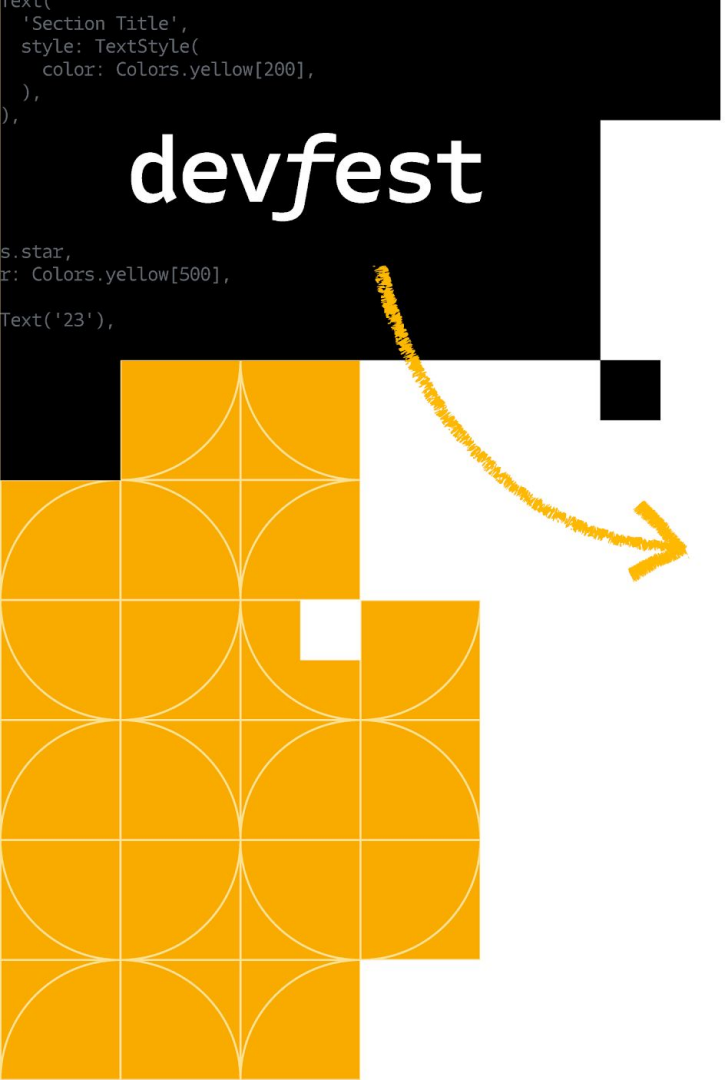
```
s.star,
r: Colors.yellow[500],

Text('23'),
```





“Gain greater observability and reliability, reduce complexity, and ensuring high availability and fault tolerance communication between containerized application with service mesh”

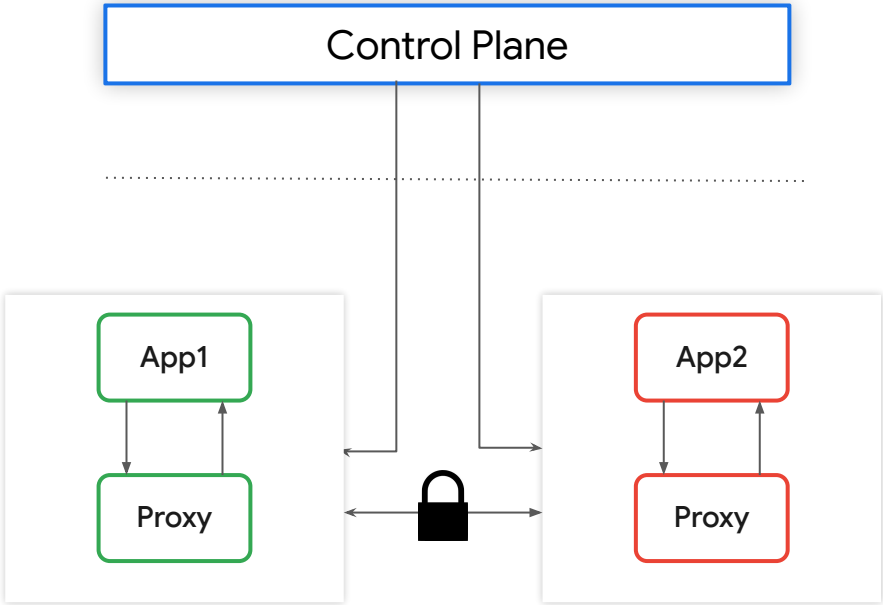


Google Developer Groups

Service Mesh

Service Mesh – What?

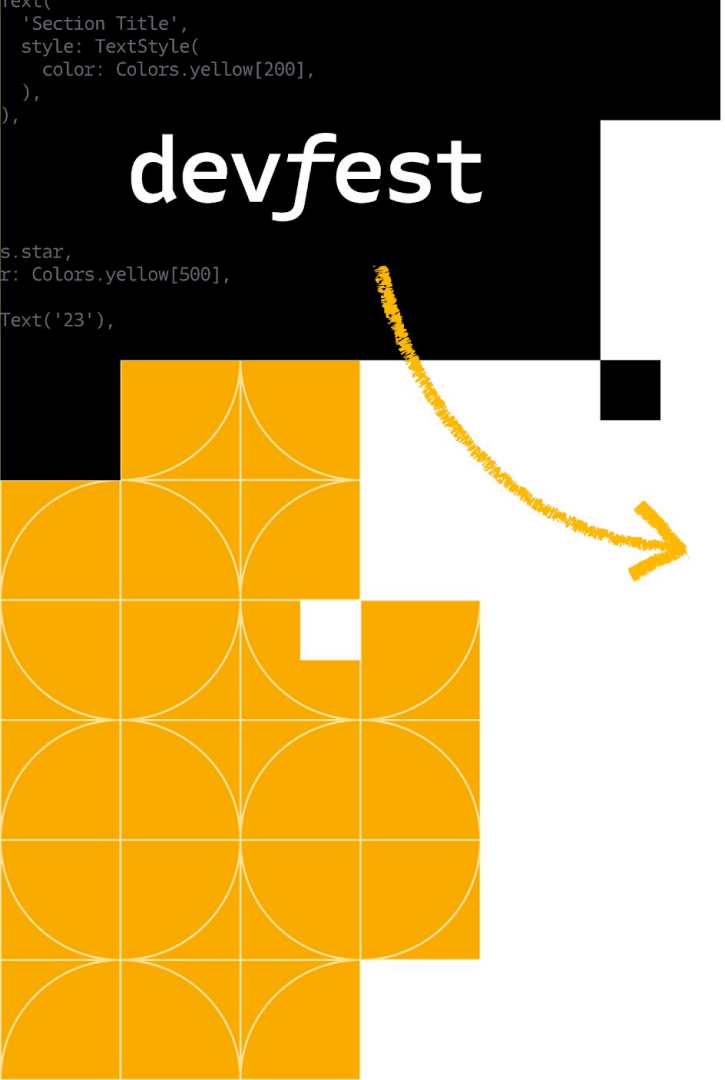
Agnostic programmable framework that has policies and controls to govern how microservices interact



Service Mesh Behaviors

- Traffic shaping with dynamic routing controls
- Resiliency support for service communication
- Observability of traffic
- Tracing of communication flows
- Secure communication

	Istio	Linkerd	Consul	Anthos Service Mesh
Workloads	Kubernetes + VMs + baremetal	Kubernetes	Kubernetes + VMs + baremetal	Kubernetes, VMs, bare metal
Supported Ingress Controller	Istio Ingress	No	Yes (Envoy)	Yes
Traffic Management Feature	Load balancing, routing, fault injection, circuit breaking, service discovery, telemetry	Load balancing, routing, fault injection, circuit breaking, service discovery, telemetry	Service discovery, service health checking, load balancing, routing	Load balancing, routing, fault injection, circuit breaking, service discovery, telemetry
Monitoring	Grafana, Prometheus, Kiali, Jaeger	Prometheus, Grafana, Jaeger	Prometheus, Grafana, Datadog	Grafana, Prometheus, Kiali, Jaeger
Multicluster	Yes	No	Yes	Yes
Deployment	Helm and Operator	Helm	Helm	GCloud Console, CLI, asmcli tool
Complexity	High	Low	Medium	Medium complexity



Google Developer Groups

Anthos Service Mesh

Antos Service Mesh

Anthos Service Mesh (ASM) is a managed service mesh built on top of Istio that helps you manage, monitor, and secure microservices architectures.

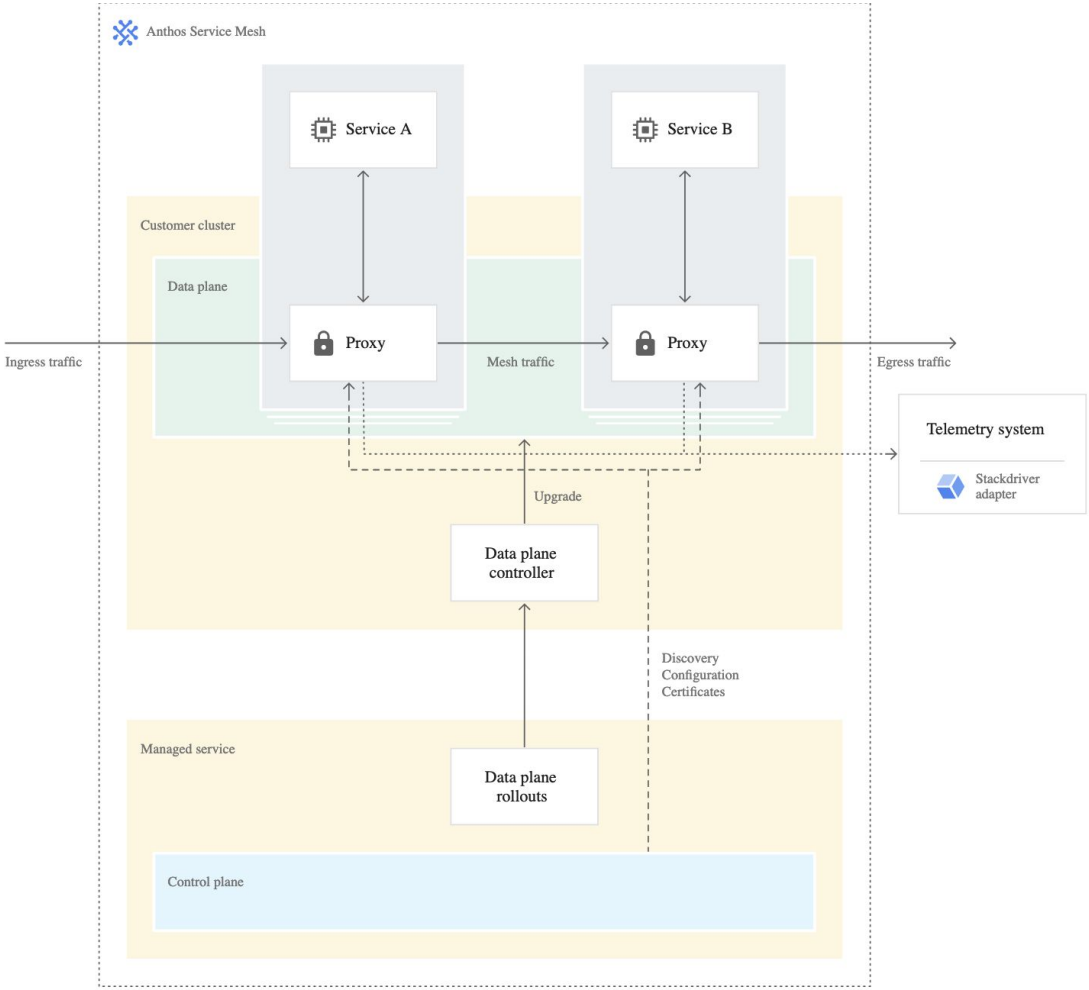
ASM is available in two deployment options:

- **Managed Anthos Service Mesh:** This is the simplest and most recommended deployment option. ASM provisions and manages a dedicated control plane for your mesh. You only need to install the ASM agent on your workloads.
- **In-cluster Anthos Service Mesh**: This deployment option allows you to run ASM on your own Kubernetes clusters. This gives you more control over the deployment and management of ASM, but it also requires more effort to set up and maintain.

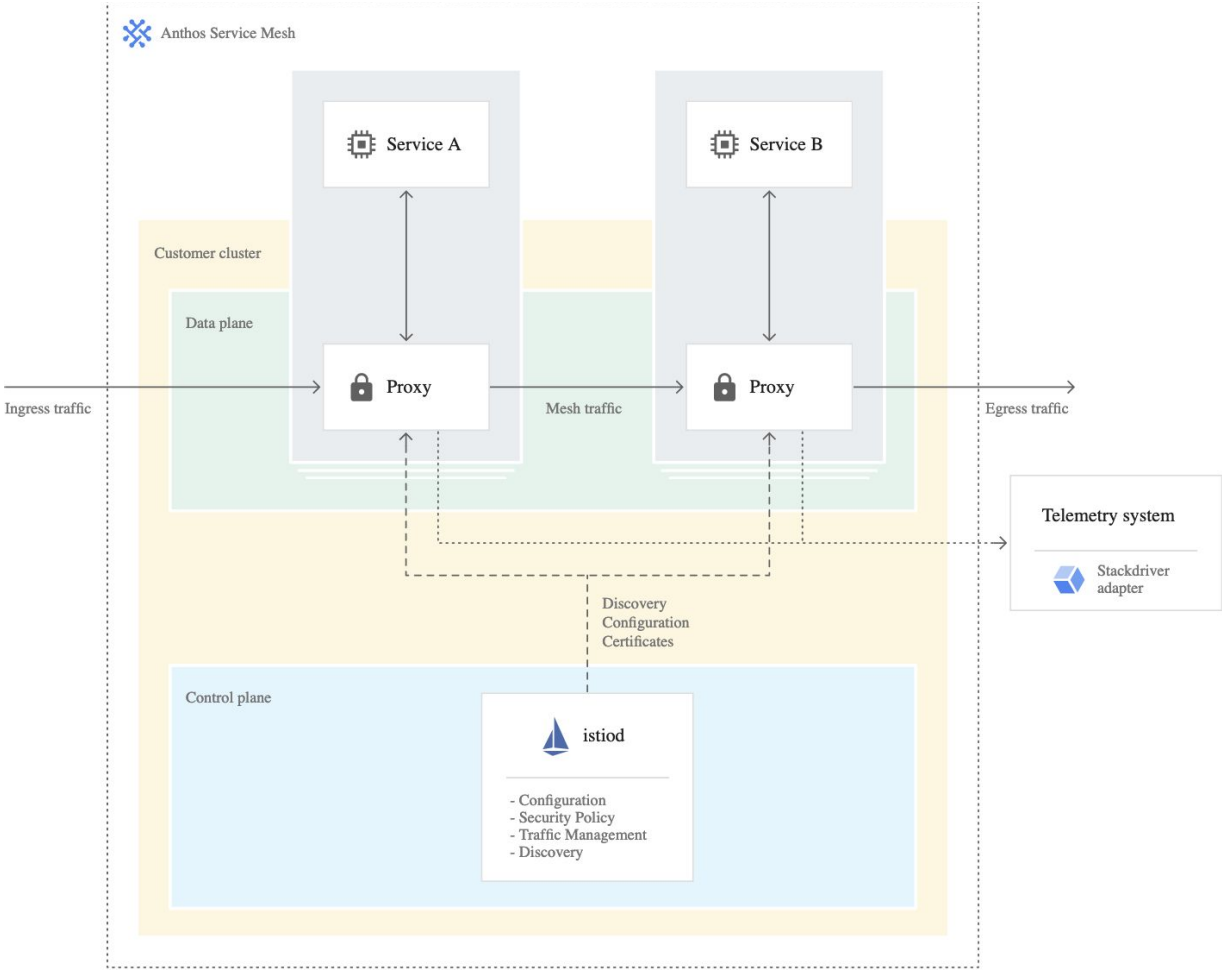
ASM Features

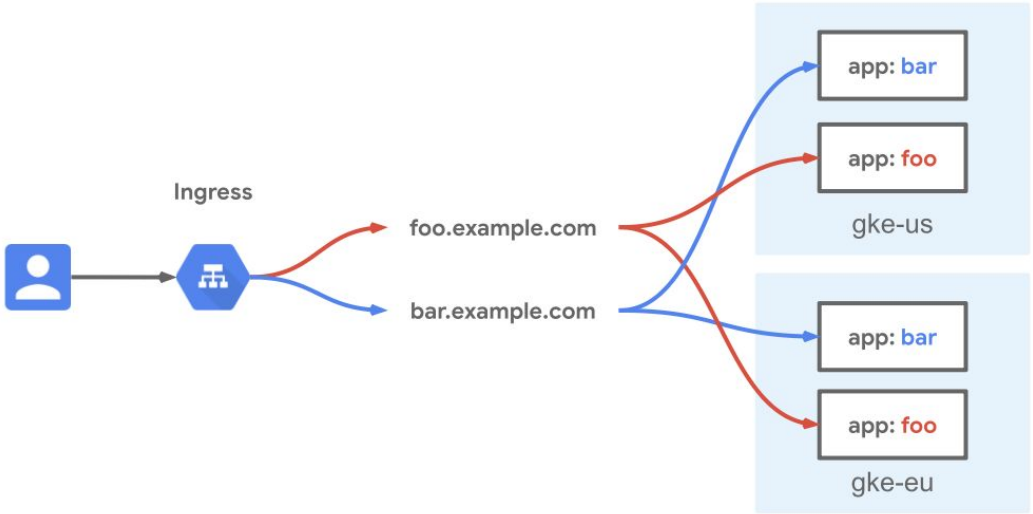
- **Service discovery and load balancing**: ASM automatically discovers all services in your mesh and load balances traffic between them.
- **Traffic management:** ASM allows you to route traffic between your services in a variety of ways, including based on path, header, or load balancing policy.
- **Security**: ASM provides a number of security features, including mutual TLS encryption, service identity, and traffic authorization.
- **Observability**: ASM provides a rich set of observability features, including tracing, monitoring, and logging.

Managed Anthos Service Mesh



In-cluster Control Plane




Ingress
foo.example.com
bar.example.com
app: bar
app: foo
gke-us
app: bar
app: foo
gke-eu

Enable the Anthos Service Mesh fleet feature

```
gcloud container fleet mesh enable --project FLEET_PROJECT_ID
```

Create an operator manifest for the egress gateway

```
cat << EOF > egressgateway-operator.yaml
apiVersion: install.istio.io/v1alpha1
kind: IstioOperator
metadata:
  name: egressgateway-operator
  annotations:
    config.kubernetes.io/local-config: "true"
spec:
  profile: empty
  revision: REVISION
  components:
    egressGateways:
    - name: istio-egressgateway
      namespace: istio-egress
      enabled: true
  values:
    gateways:
      istio-egressgateway:
        injectionTemplate: gateway
        tolerations:
          - key: "dedicated"
            operator: "Equal"
            value: "gateway"
        nodeSelector:
          cloud.google.com/gke-nodepool: "gateway"
EOF
```

Apply a default PeerAuthentication policy for the mesh

```
cat <<EOF | kubectl apply -f -
apiVersion: "security.istio.io/v1beta1"
kind: "PeerAuthentication"
metadata:
  name: "default"
  namespace: "istio-system"
spec:
  mtls:
    mode: STRICT
EOF
```

After cluster is registered to fleet

```
ananda_dwirahmawati313@cloudshell:~ (antoshmesh)$ gcloud container fleet memberships list --project ${PROJECT_ID}
NAME: antoshmesh
EXTERNAL_ID: 7590860e-63d1-4337-9312-4ee44d71085c
LOCATION: asia-southeast1
ananda_dwirahmawati313@cloudshell:~ (antoshmesh)$
```

ASM configuration will automatically injected

```
ananda_dwirahmawati313@cloudshell:~ (antoshmesh)$ kubectl create namespace istio-egress
namespace/istio-egress created
ananda_dwirahmawati313@cloudshell:~ (antoshmesh)$ kubectl -n istio-system get
controlplanerevision
NAME          RECONCILED   STALLED   AGE
asm-managed   True         False     16m
ananda_dwirahmawati313@cloudshell:~ (antoshmesh)$
```

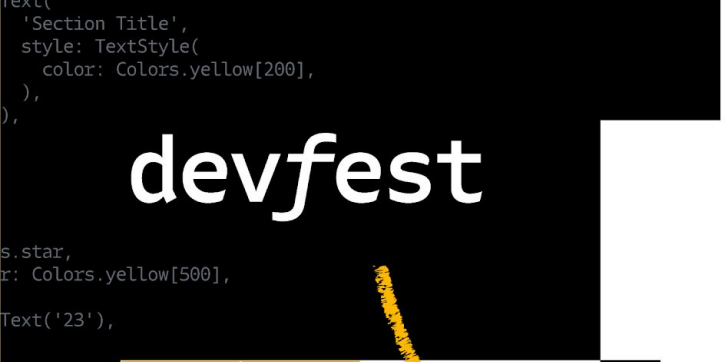
Control Plan has been provisioned

```
ananda_dwirahmawati313@cloudshell:~ (antoshmesh)$ gcloud container fleet mesh describe --project ${PROJECT_ID}
createTime: '2023-11-10T15:47:22.374045451Z'
membershipSpecs:
  projects/287832608307/locations/asia-southeast1/memberships/antoshmesh:
    mesh:
      management: MANAGEMENT_AUTOMATIC
membershipStates:
  projects/287832608307/locations/asia-southeast1/memberships/antoshmesh:
    servicemesh:
      controlPlaneManagement:
        details:
        - code: REVISION_READY
          details: 'Ready: asm-managed'
        state: ACTIVE
      dataPlaneManagement:
        details:
        - code: OK
          details: Service is running.
        state: ACTIVE
    state:
      code: OK
      description: 'Revision(s) ready for use: asm-managed.'
      updateTime: '2023-11-11T00:12:29.561475685Z'
name: projects/antoshmesh/locations/global/features/servicemesh
resourceState:
  state: ACTIVE
spec: {}
updateTime: '2023-11-11T00:12:36.683308023Z'
ananda_dwirahmawati313@cloudshell:~ (antoshmesh)$
```

Google Developer Groups

Demo

Link: https://s.id/anthosmeshguide01

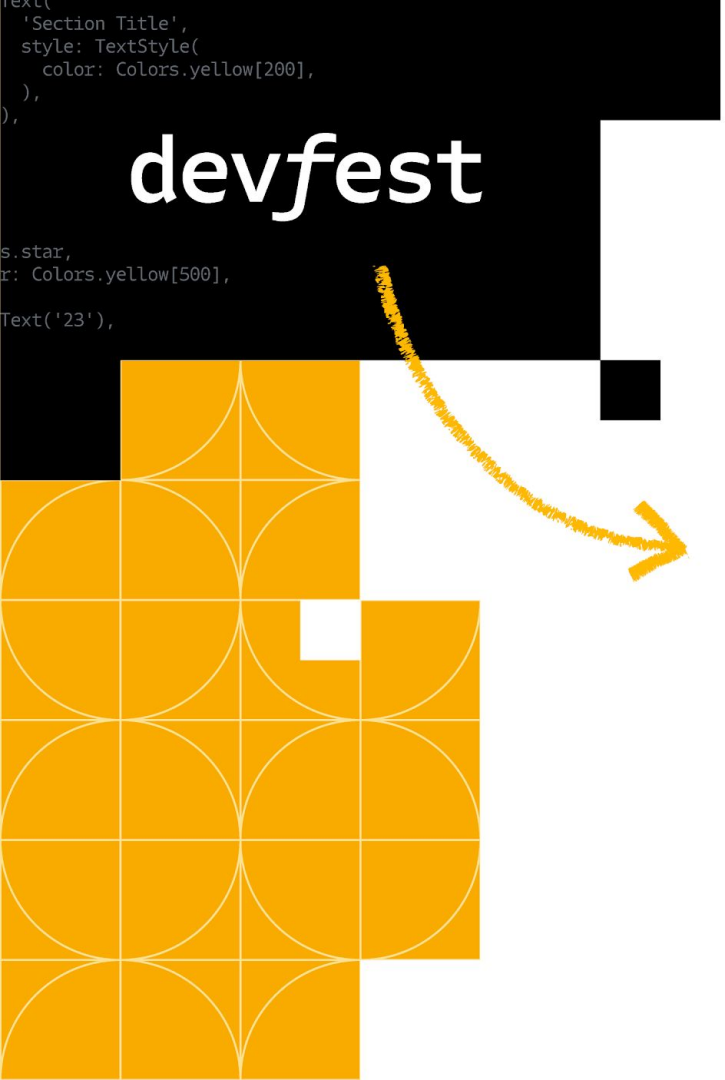


Google Developer Groups

References

References

- https://cloud.google.com/anthos/service-mesh
- https://cloud.google.com/service-mesh/docs/security/egress-gateway-gke-tutorial
- https://cloud.google.com/service-mesh/docs/unified-install/install-anthos-service-mesh
- https://cloud.google.com/service-mesh/docs/managed/enable-managed-anthos-service-mesh-optional-features#envoy_access_logs
- https://cloud.google.com/service-mesh/docs/downloading-istioctl
-



Google Developer Groups

Thank you!