Arun Ananthampalayam Follow
Nov 6, 2021 · 5 min read · Listen

Demystifying Istio and Anthos Service Mesh on GKE

What is Istio service mesh ?

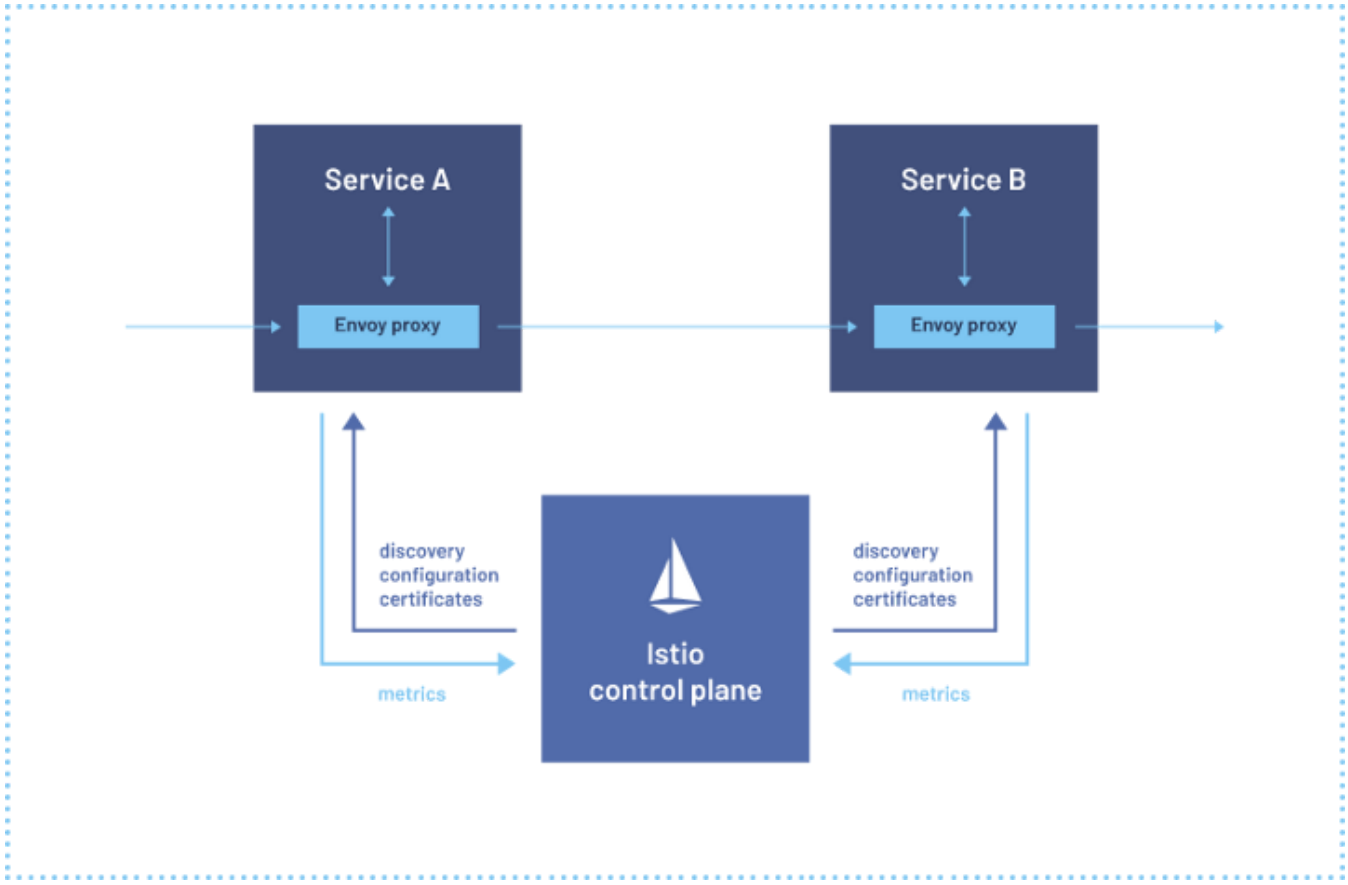
Istio is an open service mesh that provides a uniform way to connect, manage, and secure microservices. Istio extends Kubernetes to establish a programmable, application-aware network using the powerful Envoy service proxy. Working with both Kubernetes and traditional workloads, Istio brings standard, universal traffic management, telemetry, and security to complex deployments.

Istio gives you the following benefits:

- Automatic load balancing for HTTP, gRPC, WebSocket, MongoDB, and TCP traffic.
- Fine-grained control of traffic behavior with rich routing rules, retries, failovers, and fault injection.
- A configurable policy layer and API that supports 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.

You configure Istio access control, routing rules, and so on by using a custom Kubernetes API, either via kubectl or the Istio command-line tool istioctl, which

provides extra validation.



You can find out more about Istio in the open source documentation set at istio.io.

Hands-on demo Istio on minikube

Online Boutique is a cloud-native microservices demo application. Online Boutique consists of a 10-tier microservices application. The application is a web-based e-commerce app where users can browse items, add them to the cart, and purchase them. This application to demonstrate use of technologies like Kubernetes, Istio, GKE and Anthos.

1. Start the minikube cluster with custom CPU and memory resources and install istio.

```
$ minikube start — cpu 6 — memory 8192
```

```
arunkasi-macbookpro1:microservices-demo arunkasi$ minikube start --cpu 6 --memory 8192
Error: unknown flag: --cpu
See 'minikube start --help' for usage.
arunkasi-macbookpro1:microservices-demo arunkasi$ minikube start --cpus 6 --memory 8192
😄  minikube v1.23.2 on Darwin 11.6
✨  Using the docker driver based on existing profile

💣  Exiting due to PROVIDER_DOCKER_NOT_RUNNING: "docker version --format -" exit status 1: Cannot connect to the Docker daem
on at unix:///var/run/docker.sock. Is the docker daemon running?
💡  Suggestion: Start the Docker service
📘  Documentation: https://minikube.sigs.k8s.io/docs/drivers/docker/
```

```
$ curl -L https://istio.io/downloadIstioexport

$ PATH="$PATH:/Users/arunkasi/Documents/istio-1.11.4/bin" | sh -
```

```
arunkasi-macbookpro1:Documents arunkasi$ curl -L https://istio.io/downloadIstio | sh -
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   102  100   102    0     0    404      0 --:--:-- --:--:-- --:--:--   403
100  4549  100  4549    0     0  13498      0 --:--:-- --:--:-- --:--:-- 13498

Downloading istio-1.11.4 from https://github.com/istio/istio/releases/download/1.11.4/istio-1.11.4-osx.tar.gz ...
Istio 1.11.4 Download Complete!

Istio has been successfully downloaded into the istio-1.11.4 folder on your system.

Next Steps:
See https://istio.io/latest/docs/setup/install/ to add Istio to your Kubernetes cluster.

To configure the istioctl client tool for your workstation,
add the /Users/arunkasi/Documents/istio-1.11.4/bin directory to your environment path variable with:
         export PATH="$PATH:/Users/arunkasi/Documents/istio-1.11.4/bin"
```

```
arunkasi-macbookpro1:bin arunkasi$ istioctl install --set profile=demo -y
✔ Istio core installed
✔ Istiod installed
✔ Ingress gateways installed
✔ Egress gateways installed
✔ Installation complete
Thank you for installing Istio 1.11.  Please take a few minutes to tell us about your install/upgrade experience!
```

2. Add a namespace label to instruct Istio to automatically inject Envoy sidecar proxies when you deploy your application later

```
$ kubectl label namespace default istio-injection=enabled
```

```
arunkasi-macbookpro1:istio-1.11.4 arunkasi$ ls
LICENSE         README.md       bin             manifest.yaml   manifests       samples         tools
arunkasi-macbookpro1:istio-1.11.4 arunkasi$ kubectl apply -f samples/bookinfo/platform/kube/bookinfo.yaml
service/details created
serviceaccount/bookinfo-details created
deployment.apps/details-v1 created
service/ratings created
serviceaccount/bookinfo-ratings created
deployment.apps/ratings-v1 created
service/reviews created
serviceaccount/bookinfo-reviews created
deployment.apps/reviews-v1 created
deployment.apps/reviews-v2 created
deployment.apps/reviews-v3 created
service/productpage created
serviceaccount/bookinfo-productpage created
deployment.apps/productpage-v1 created
```

```
$ kubectl get services
```

```
arunkasi-macbookpro1:istio-1.11.4 arunkasi$ kubectl get services
NAME          TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)    AGE
details       ClusterIP   10.44.10.225   <none>        9080/TCP   35s
kubernetes    ClusterIP   10.44.0.1      <none>        443/TCP    4d9h
productpage   ClusterIP   10.44.10.185   <none>        9080/TCP   33s
ratings       ClusterIP   10.44.11.129   <none>        9080/TCP   34s
reviews       ClusterIP   10.44.14.43    <none>        9080/TCP   34s
```

```
$ kubectl get pods
```

```
arunkasi-macbookpro1:istio-1.11.4 arunkasi$ kubectl get pods
NAME                          READY   STATUS    RESTARTS   AGE
details-v1-79f774bdb9-zw5gv       2/2     Running   0          84s
productpage-v1-6b746f74dc-pvw4s   2/2     Running   0          82s
ratings-v1-b6994bb9-cqbzp         2/2     Running   0          83s
reviews-v1-545db77b95-8pw76       2/2     Running   0          82s
reviews-v2-7bf8c9648f-7w4k7       2/2     Running   0          82s
reviews-v3-84779c7bbc-drjbw       2/2     Running   0          82s
```

VERY IMPORTANT — → Istio sidecar proxy is injected.

```
$ kubectl describe pods details-v1-79f774bdb9-zw5gv
```

```
istio-proxy:
  Container ID:  containerd://f6cf8e60e6a9e521a0df5cb26f46a948772fcc85b3c2a681072b6c3fe21e79ae
  Image:         docker.io/istio/proxyv2:1.11.4
  Image ID:      docker.io/istio/proxyv2@sha256:68b1012e5ba209161671f0e76c92baad845cef810fa0a7cf1b105fee8f0d3e46
  Port:          15090/TCP
```

3.Verify everything is working correctly up to this point. Run this command to see if the app is running inside the cluster and serving HTML pages by checking for the page title in the response.

```
arunkasi-macbookpro1:istio-1.11.4 arunkasi$ kubectl exec "$(kubectl get pod -l app=ratings -o jsonpath='{.items[0].metadata.
name}')" -c ratings -- curl -sS productpage:9080/productpage | grep -o "<title>.*</title>"

<title>Simple Bookstore App</title>
```

4. The Bookinfo application is deployed but not accessible from the outside. To make it accessible, you need to create an Istio Ingress Gateway, which maps a path to a route at the edge of your mesh.

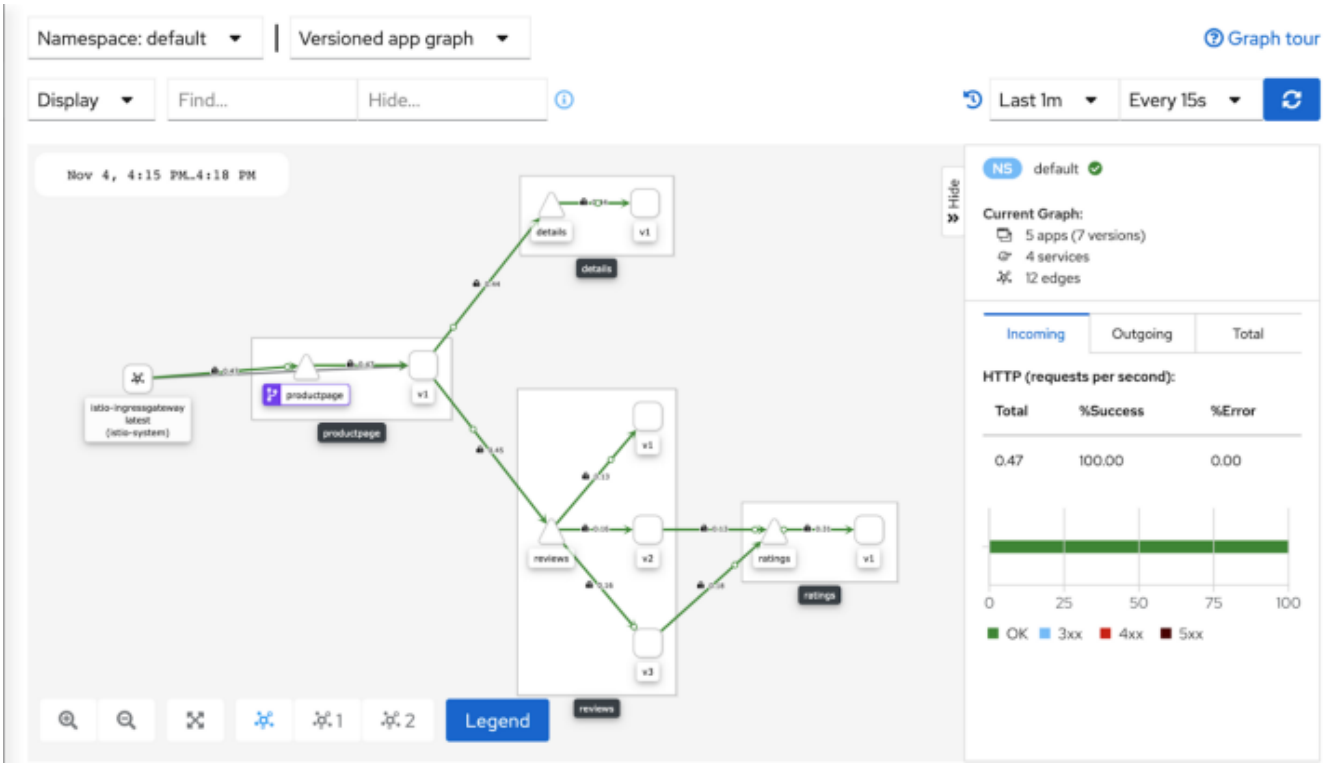
```
$ kubectl apply -f samples/bookinfo/networking/bookinfo-gateway.yaml
$ istioctl analyze
```

```
arunkasi-macbookpro1:istio-1.11.4 arunkasi$ kubectl exec "$(kubectl get pod -l app=ratings -o jsonpath='{.items[0].metadata
name}')" -c ratings -- curl -sS productpage:9080/productpage | grep -o "<title>.*</title>"

<title>Simple Bookstore App</title>
arunkasi-macbookpro1:istio-1.11.4 arunkasi$
arunkasi-macbookpro1:istio-1.11.4 arunkasi$ kubectl apply -f samples/bookinfo/networking/bookinfo-gateway.yaml
gateway.networking.istio.io/bookinfo-gateway created
virtualservice.networking.istio.io/bookinfo created
arunkasi-macbookpro1:istio-1.11.4 arunkasi$ istioctl analyze
```

5. Install addon from sample folder including Kiali

Kiali is an observability console for Istio with service mesh configuration and validation capabilities. It helps you understand the structure and health of your service mesh by monitoring traffic flow to infer the topology and report errors.

```
arunkasi-macbookpro1:istio-1.11.4 arunkasi$ kubectl apply -f samples/addons
kubectl rollout status deployment/kiali -n istio-system

serviceaccount/grafana created
configmap/grafana created
service/grafana created
deployment.apps/grafana created
configmap/istio-grafana-dashboards created
configmap/istio-services-grafana-dashboards created
deployment.apps/jaeger created
service/tracing created
service/zipkin created
service/jaeger-collector created
serviceaccount/kiali created
configmap/kiali created
clusterrole.rbac.authorization.k8s.io/kiali-viewer created
clusterrole.rbac.authorization.k8s.io/kiali created
clusterrolebinding.rbac.authorization.k8s.io/kiali created
role.rbac.authorization.k8s.io/kiali-controlplane created
rolebinding.rbac.authorization.k8s.io/kiali-controlplane created
service/kiali created
deployment.apps/kiali created
serviceaccount/prometheus created
configmap/prometheus created
clusterrole.rbac.authorization.k8s.io/prometheus created
clusterrolebinding.rbac.authorization.k8s.io/prometheus created
service/prometheus created
deployment.apps/prometheus created
arunkasi-macbookpro1:istio-1.11.4 arunkasi$ kubectl rollout status deployment/kiali -n istio-system
Waiting for deployment "kiali" rollout to finish: 0 of 1 updated replicas are available...
deployment "kiali" successfully rolled out
arunkasi-macbookpro1:istio-1.11.4 arunkasi$
arunkasi-macbookpro1:istio-1.11.4 arunkasi$ istioctl dashboard kiali
```



What is Anthos Service Mesh?

Anthos Service Mesh is a suite of tools that helps you monitor and manage a reliable service mesh on-premises or on Google Cloud. Anthos Service Mesh is Google's *fully-supported* distribution of Istio. Because Anthos Service Mesh is compatible with the Istio APIs, it provides all the benefits of the Istio service mesh and more:

Traffic Management:

- Create canary and blue-green deployments.
- Provide fine-grained control over specific routes for services.
- Configure load balancing between services.
- Set up circuit breakers.

Observability insights

- The Anthos Service Mesh pages in the Google Cloud Console provide the following insights into your service mesh:
- Service metrics and logs for HTTP traffic within your mesh's GKE cluster are automatically ingested to Google Cloud.
- Preconfigured service dashboards give you the information you need to understand your services.
- In-depth telemetry — powered by Cloud Monitoring, Cloud Logging, and Cloud Trace — lets you dig deep into your service metrics and logs. You can filter and slice your data on a wide variety of attributes.
- Service-to-service relationships at a glance help you understand who connects to each service and the services that each service depends on.
- Service level objectives (SLOs) give you insight into the health of your services. You can easily define an SLO and alert on your own standards of service health.

Security benefits

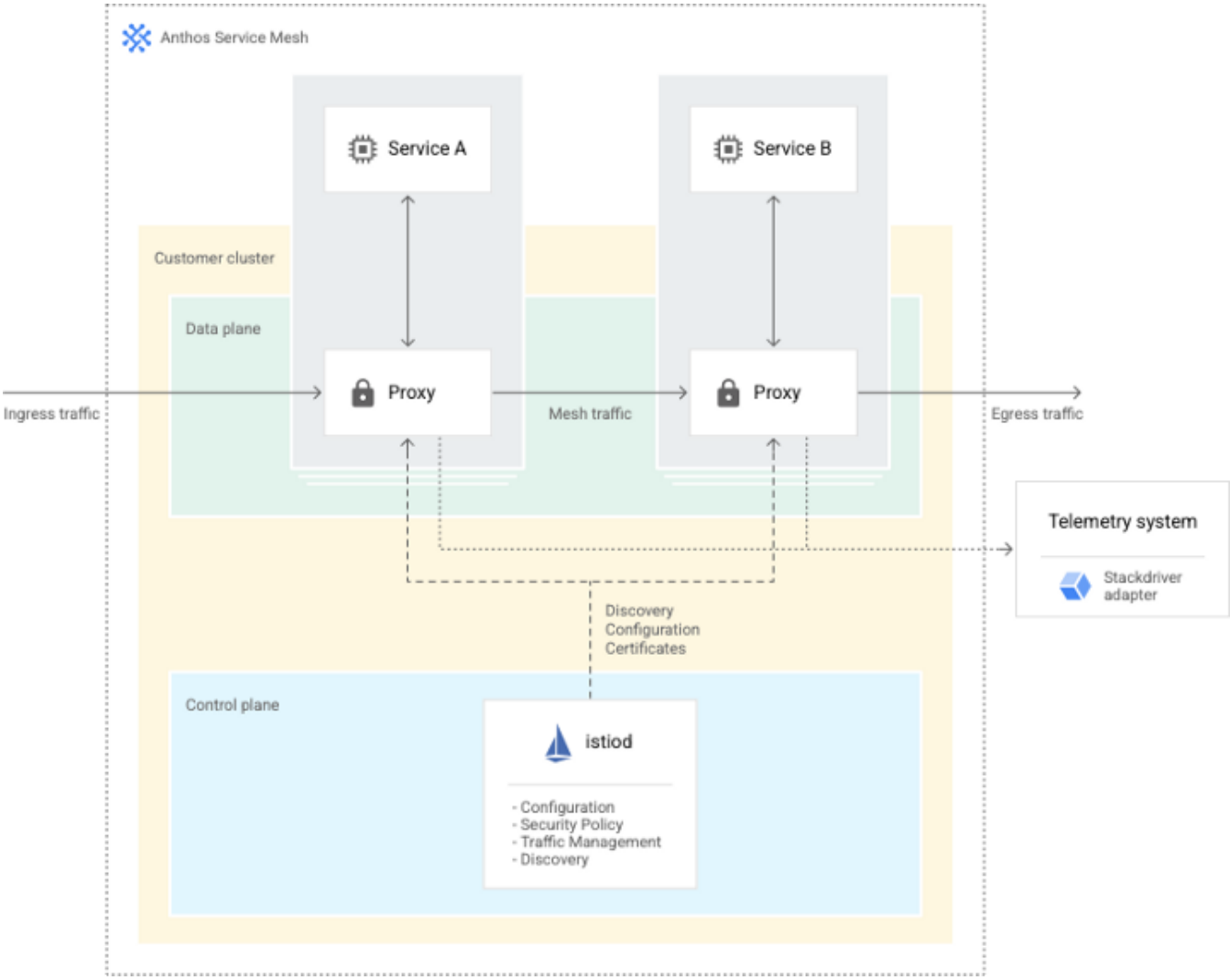
- Mitigates risk of replay or impersonation attacks that use stolen credentials. Anthos Service Mesh relies on mutual TLS (mTLS) certificates to authenticate peers, rather than bearer tokens such as JSON Web Tokens (JWT).
- Ensures encryption in transit. Using mTLS for authentication also ensures that all TCP communications are encrypted in transit.
- Ensures that only authorized clients can access a service with sensitive data, irrespective of the network location of the client and the application-level credentials.

- Mitigates the risk of user data breach within your production network. You can ensure that insiders can only access sensitive data through authorized clients.

Anthos Service Mesh is generally available for GKE as standalone services with pay-as-you-go pricing. GKE customers can now use Anthos Service Mesh to enable next-level security and networking on container-based microservices. Anthos Service Mesh is also generally available to support a hybrid mesh. This gives you the flexibility to have a common mesh that spans both your Google Cloud and on-prem deployments.

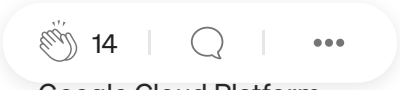
- Managed Anthos Service Mesh
- Include Compute Engine VMs in the service mesh.

I'll demonstrate the In-cluster control plane in the next article.



Reference

- Git Repo
- Istio documentation
- Anthos Service Mesh documentation



Istio Anthos Service Mesh Google Cloud Platform Service Mesh Envoy Proxy