Provision managed Anthos Service Mesh

Overview

Managed Anthos Service Mesh is a Google-managed service mesh that you simply enable. Google handles the reliability, upgrades, scaling, and security for you in a backward-compatible manner.

This page shows you how to use the fleet (/anthos/multicluster-management/fleets) feature API to set up managed Anthos Service Mesh.

When you enable managed Anthos Service Mesh using the fleet API:

- Google applies the recommended control plane configuration
- Google enables automatic data plane management (#managed-data-plane)
- Your cluster is enrolled in an Anthos Service Mesh release channel (/service-mesh/docs/managed/release-channels#available_release_channels) based on your Google Kubernetes Engine (GKE) cluster's release channel, and your control plane and data plane are kept up to date with each new release.
- Google enables endpoint discovery (#configure_endpoint_discovery_only_for_multi-cluster_installations) and cross cluster load balancing throughout your service mesh with default settings, although you must create firewall rules.

Use this onboarding path if you want:

- To use `gcloud` to configure managed Anthos Service Mesh using Google Cloud APIs and IAM.
- To configure Anthos Service Mesh using the same APIs as other fleet features.
- To automatically get the recommended configuration of Anthos Service Mesh for each of your clusters.

Note: We recommend using the fleet feature API to automatically configure managed Anthos Service Mesh, but you can also configure managed Anthos Service Mesh using the `asmcli` tool (/service-mesh/docs/managed/provision-managed-anthos-service-mesh-asmcli). You must use that manual configuration to use VPC-SC, Private CA, or to choose an Anthos Service Mesh channel that is different from your Google Kubernetes Engine (GKE) channel.

Prerequisites

As a starting point, this guide assumes that you have:

- A Cloud project (/resource-manager/docs/creating-managing-projects)
- A Cloud Billing account (/billing/docs/how-to/manage-billing-account)
- Obtained the required permissions (/service-mesh/docs/installation-permissions) to provision managed Anthos Service Mesh
- Enabled Workload Identity (/kubernetes-engine/docs/how-to/workload-identity) on your clusters.

Requirements

- One or more clusters with a supported version of GKE (/service-mesh/docs/managed/supported-features-mcp#environments), in one of the supported regions (/service-mesh/docs/managed/supported-features-mcp#regions).
- Ensure that the client machine that you provision managed Anthos Service Mesh from has network connectivity to the API server.
- Your clusters must be registered to a fleet (/anthos/multicluster-management/connect/registering-a-cluster). This is included in the instructions, or can be done separately prior to the provision.
- Your project must have the Service Mesh fleet feature enabled. This is included in the instructions or can be done separately.
- GKE Autopilot is only supported with GKE version 1.21.3+. CNI will be installed and managed by Google.
- Managed Anthos Service Mesh can use multiple GKE clusters in a single-project single-network environment or a multi-project single-network environment.

Limitations

Caution: Note the following fleet-specific limitations: - All Anthos Service Mesh clusters for one mesh must be registered to the same fleet at all times to use Anthos Service Mesh. Other clusters in the project of an Anthos Service Mesh cluster must not be registered to a different fleet.

We recommend that you review the list of managed Anthos Service Mesh supported features and limitations (/service-mesh/docs/managed/supported-features-mcp). In particular, note the following:

- The `IstioOperator` API isn't supported since its main purpose is to control in-cluster components.

★ **Note:** You can still enable optional features without using the IstioOperator API, see Enabling optional features on managed Anthos Service Mesh (/service-mesh/docs/managed-service-mesh-optional-features). You can also use the migration tool included with **`asmcli`** to automatically convert other **`IstioOperator`** optional features to be compatible with managed control plane. For more information, see Migrate from **`IstioOperator`** (/service-mesh/docs/migrate-istio-operator).

- Enabling managed Anthos Service Mesh with the fleet API will use Mesh CA (/service-mesh/docs/security/security-overview#features). If your service mesh deployment requires Certificate Authority Service (CA Service), then follow Provision managed Anthos Service Mesh using **`asmcli`** (/service-mesh/docs/managed/provision-managed-anthos-service-mesh-asmcli).
- Migrations from managed Anthos Service Mesh with **`asmcli`** (/service-mesh/docs/managed/provision-managed-anthos-service-mesh-asmcli) to Anthos Service Mesh with fleet API (/service-mesh/docs/managed/provision-managed-anthos-service-mesh) are not supported. Similarly, configuring managed Anthos Service Mesh with fleet API from **`--management manual`** to **`--management automatic`** is not supported.
- For GKE Autopilot clusters, cross-project setup is only supported with GKE 1.23 or later.
- For GKE Autopilot clusters, in order to adapt to the GKE Autopilot resource limit (/kubernetes-engine/docs/concepts/autopilot-overview#default_contaizner_resource_requests), the default proxy resource requests and limits are set to 500m CPU and 512 Mb memory. You can override the default values using custom injection (#custom-injection).
- The actual features available to managed Anthos Service Mesh depend on the release channel (/service-mesh/docs/managed/select-a-release-channel). For more information,

review the full list of managed Anthos Service Mesh supported features and limitations (/service-mesh/docs/supported-features-mcp).

- During the provisioning process for a managed control plane, Istio CRDs corresponding to the selected channel are provisioned in the specified cluster. If there are existing Istio CRDs in the cluster, they will be overwritten.

Before you begin

1. In the Google Cloud console, on the project selector page, select or create a Google Cloud project (/resource-manager/docs/creating-managing-projects).

   Go to project selector (https://console.cloud.google.com/projectselector2/home/dashboard)

2. Make sure that billing is enabled for your Cloud project. Learn how to check if billing is enabled on a project (/billing/docs/how-to/verify-billing-enabled).

3. Enable the required APIs on your fleet host project.

```
gcloud services enable mesh.googleapis.com \
    --project=FLEET_PROJECT_ID
```

where:

- ***FLEET_PROJECT_ID*** is the ID of your Fleet Host project.

Enabling mesh.googleapis.com enables the following APIs:

API
meshconfig.googleapis.com
meshca.googleapis.com
container.googleapis.com
monitoring.googleapis.com
gkehub.googleapis.com
stackdriver.googleapis.com

opsconfigmonitoring.googleapis.com
connectgateway.googleapis.com

Enable the Anthos Service Mesh fleet feature

Enable Anthos Service Mesh on the fleet project. Note that if you plan to register multiple clusters, enabling Anthos Service Mesh happens at the fleet-level so you only have to run this command once.

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

Configure each cluster

Use the following steps to configure managed Anthos Service Mesh for each cluster in your mesh.

Configure gcloud

Do the following steps even if you are using Cloud Shell.

1. Authenticate with the Google Cloud CLI:

   ```
   gcloud auth login --project PROJECT_ID
   ```

2. Update the components:

   ```
   gcloud components update
   ```

3. Configure `kubectl` to point to the cluster.

★ **Note:** Use `--region` instead of `--zone`, if the cluster is a regional cluster.

```
gcloud container clusters get-credentials CLUSTER_NAME \
     --zone CLUSTER_LOCATION \
     --project PROJECT_ID
```

Register clusters to a fleet

1. Register a GKE cluster (/anthos/fleet-management/docs/register/gke) using Workload Identity to a fleet:

   ```
   gcloud container fleet memberships register MEMBERSHIP_NAME \
     --gke-uri=GKE_URI \
     --enable-workload-identity \
     --project FLEET_PROJECT_ID
   ```

   where:

   - *MEMBERSHIP_NAME* is the membership name that you choose to uniquely represent the cluster being registered to the fleet.
   - *GKE_URI* is the URI of the GKE cluster. For example: `https://container.googleapis.com/v1/projects/my-gke-project/locations/us-central1-a/clusters/my-gke-cluster`. You can obtain the URI by running `gcloud container clusters list --uri`.

2. Verify your cluster is registered:

   ```
   gcloud container fleet memberships list --project FLEET_PROJECT_ID
   ```

3. If your cluster's project differs from your fleet host project, you must allow Anthos Service Mesh service accounts in the fleet project to access the cluster project, and enable required APIs on the cluster project. You only need to do this once for each cluster project.

   If you previously used `asmcli` to configure managed Anthos Service Mesh for this combination of cluster and fleet projects, then these changes have already been applied and you do not have to run the following commands.

   Grant service accounts in the fleet project permission to access the cluster project:

```
gcloud projects add-iam-policy-binding "PROJECT_ID "  \
  --member "serviceAccount:service-FLEET_PROJECT_NUMBER @gcp-sa-servicem
  --role roles/anthosservicemesh.serviceAgent
```

Enable the Mesh API on the cluster's project:

```
gcloud services enable mesh.googleapis.com \
  --project=PROJECT_ID
```

Apply the mesh_id label

Zonal ClusterRegional Cluster...
(#zonal-cluster)

```
gcloud container clusters update  --project PROJECT_ID  CLUSTER_NAME
  --zone CLUSTER_LOCATION  --update-labels mesh_id=proj-FLEET_PROJECT_
```

where:

- *PROJECT_ID* is the unique identifier of your cluster project.
- *CLUSTER_NAME* is the name of your cluster.
- *CLUSTER_LOCATION* is the compute zone or region for your cluster.
- *FLEET_PROJECT_NUMBER* is the unique identifier for your fleet host project.

Enable automatic management

Run the following command to enable automatic management:

Note: You can apply this change to multiple clusters at a time by passing a comma-separated list of membership names as the **`--memberships`** parameter.

```
gcloud container fleet mesh update \
   --management automatic \
   --memberships MEMBERSHIP_NAME  \
  --project FLEET_PROJECT_ID
```

Note that an ingress gateway isn't automatically deployed with the control plane. Decoupling the deployment of the ingress gateway and control plane allows you to more easily manage your gateways in a production environment. If the cluster needs an ingress gateway or an egress gateway, see Deploy gateways (/service-mesh/docs/gateways#deploy_gateways). To enable other optional features, see Enabling optional features on managed Anthos Service Mesh (/service-mesh/docs/managed-service-mesh-optional-features).

Verify the control plane has been provisioned

1. After a few minutes, verify that the control plane status is `ACTIVE`:

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

   The output is similar to:

   ```
   ...
   membershipSpecs:
     projects/746296320118/locations/global/memberships/demo-cluster-1:
       mesh:
         management: MANAGEMENT_AUTOMATIC
   membershipStates:
     projects/746296320118/locations/global/memberships/demo-cluster-1:
       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.'
   ...
   ```

Take note of the revision label in the `details` field, for example, `asm-managed` in the provided output. If you are using revision labels, then you need to set this label before you Deploy applications (#deploy_applications). If you are using default injection labels, then you do not need to set this label.

Managed data plane

If you use managed Anthos Service Mesh with fleet API, Google will fully manage upgrades of your proxies, unless you opt out at the namespace or workload level.

If enabled, the sidecar proxies and injected gateways are automatically updated in conjunction with the managed control plane by restarting workloads to re-inject new versions of the proxy. This normally completes 1-2 weeks after the managed control plane is upgraded.

If disabled, proxy management is driven by the natural lifecycle of the pods in the cluster and must be manually triggered by the user to control the update rate.

The managed data plane upgrades proxies by evicting pods that are running older versions of the proxy. The evictions are done gradually, honoring the pod disruption budget and controlling the rate of change.

Note that the managed data plane requires the Istio Container Network Interface (CNI) plugin, which is enabled by default when you deploy the managed control plane.

Note: If the cluster has a GKE maintenance window (/kubernetes-engine/docs/concepts/maintenance-windows-and-exclusions) defined, pod evictions will commence at the start of the next available maintenance window and continue without pause until all managed pods have been updated.

The managed data plane doesn't manage the following:

- Uninjected pods
- Manually injected pods
- Jobs
- StatefulSets
- DaemonSets

Disable the managed data plane (optional)

You can disable the managed data plane for individual namespaces or pods.

To disable the managed data plane for a namespace:

```
kubectl annotate --overwrite namespace NAMESPACE \
  mesh.cloud.google.com/proxy='{"managed":"false"}'
```

To disable the managed data plane for a pod:

```
kubectl annotate --overwrite pod POD_NAME \
  mesh.cloud.google.com/proxy='{"managed":"false"}'
```

Enable maintenance notifications

You can request to be notified about upcoming managed data plane maintenance up to week before maintenance is scheduled. Maintenance notifications are not sent by default. You must also Configure a GKE maintenance window (/kubernetes-engine/docs/how-to/maintenance-windows-and-exclusions) before you can receive notifications.

To opt in to managed data plane maintenance notifications:

1. Go to the **Communication** page.

   Go to the Communication page (https://console.cloud.google.com/user-preferences/communica

2. In the **Anthos Service Mesh Upgrade** row, under the **Email** column, select the radio button to turn maintenance notifications **ON**.

Each user that wants to receive notifications must opt in separately. If you want to set an email filter for these notifications, the subject line is:

```
Upcoming upgrade for your Anthos Service Mesh cluster
"CLUSTER_LOCATION/CLUSTER_NAME".
```

The following example shows a typical managed data plane maintenance notification:

> Subject Line: Upcoming upgrade for your ASM cluster "`<location/cluster-name>`"
>
> Dear Anthos Service Mesh user,

> The Anthos Service Mesh components in your cluster ${instance_id} (https://console.cloud.google.com/kubernetes/clusters/details/${instance_id}/details?project=${project_id}) are scheduled to upgrade on ${scheduled_date_human_readable} at ${scheduled_time_human_readable}.
>
> You can check the release notes (https://cloud.google.com/service-mesh/docs/release-notes) to learn about the new update.
>
> In the event that this maintenance gets canceled, you'll receive another email.
>
> Sincerely,
>
> The Anthos Service Mesh Team
>
> (c) 2022 Google LLC 1600 Amphitheater Parkway, Mountain View, CA 94043 You have received this announcement to update you about important changes to Google Cloud Platform or your account. You can opt out of maintenance window notifications by editing your user preferences: https://console.cloud.google.com/user-preferences/communication?project=${project_id}

Configure endpoint discovery (only for multi-cluster installations)

Note: If your mesh has only one cluster, skip these multi-cluster steps and proceed to Deploy applications (#deploy_applications) or Migrate applications (#migrate_applications_to_managed_anthos_service_mesh).

Before you continue, you should have already configured managed Anthos Service Mesh on each cluster as described in the previous steps. There is no need to indicate that a cluster is a primary cluster, this is the default behavior.

Additionally, ensure you have downloaded `asmcli` (/service-mesh/docs/unified-install/install-dependent-tools#download_asmcli) (only if you wish to verify your configuration with the sample application) and set the project and cluster variables (/service-mesh/docs/unified-install/gke-install-multi-cluster#setting_project_and_cluster_variables).

Note: For more information on endpoint discovery, refer to Endpoint discovery with multiple control planes

(https://istio.io/v1.15/docs/ops/deployment/deployment-models/#endpoint-discovery-with-multiple-control-planes)

.

Public clustersPrivate clusters...
(#public-clusters)

Configure endpoint discovery between public clusters

Enabling managed Anthos Service Mesh with the fleet API will enable endpoint discovery for this cluster. However, you must open firewall ports (/service-mesh/docs/unified-install/gke-install-multi-cluster#create_firewall_rule). To disable endpoint discovery for one or more clusters, see the instructions to disable it in Endpoint discovery between public clusters with declarative API (/service-mesh/docs/unified-install/gke-install-multi-cluster#endpoint-discovery-declarative-api)

.

Note: The **HelloWorld service** example uses Docker Hub (https://hub.docker.com/). In a private cluster, the container runtime can pull container images from Artifact Registry (/artifact-registry/docs) by default. The container runtime cannot pull images from any other container image registry on the internet. You can download the image and push to Artifact Registry or use Cloud NAT (/nat/docs/overview) to provide outbound internet access for certain private nodes. For more information, see Migrate external containers (/container-registry/docs/migrate-external-containers) and Creating a private cluster (/kubernetes-engine/docs/how-to/private-clusters#pulling_images).

Deploy applications

To deploy applications, use either the label corresponding to the channel you configured during installation or `istio-injection=enabled` if you are using default injection labels.

Default injection labelRevision label (#revision-label)
(#default-injection-label)

```
kubectl label namespace NAMESPACE istio-injection=enabled istio.io/re
```

At this point, you have successfully configured Anthos Service Mesh managed control plane. If you have any existing workloads in labeled namespaces, then restart them so they get proxies injected.

You are now ready to deploy your applications, or you can deploy the Bookinfo sample application (/service-mesh/docs/deploy-bookinfo).

If you deploy an application in a multi-cluster setup, replicate the Kubernetes and control plane configuration in all clusters, unless you plan to limit that particular config to a subset of clusters. The configuration applied to a particular cluster is the source of truth for that cluster.

Customize injection (optional)

Caution: Overriding and customizing injection settings can lead to unforeseen configuration errors and resulting issues with sidecar containers.

Per-pod configuration is available to override these options on individual pods. This is done by adding an `istio-proxy` container to your pod. The sidecar injection will treat any configuration defined here as an override to the default injection template.

For example, the following configuration customizes a variety of settings, including lowering the CPU requests, adding a volume mount, and adding a `preStop` hook:

```
apiVersion: v1
kind: Pod
metadata:
  name: example
spec:
  containers:
  - name: hello
    image: alpine
  - name: istio-proxy
    image: auto
    resources:
      requests:
        cpu: "200m"
        memory: "256Mi"
      limites:
        cpu: "200m"
        memory: "256Mi"
    volumeMounts:
    - mountPath: /etc/certs
      name: certs
    lifecycle:
      preStop:
        exec:
          command: ["sleep", "10"]
  volumes:
  - name: certs
    secret:
      secretName: istio-certs
```

In general, any field in a pod can be set. However, care must be taken for certain fields:

- Kubernetes requires the `image` field to be set before the injection has run. While you can set a specific image to override the default one, it is recommended to set the `image` to `auto` which will cause the sidecar injector to automatically select the image to use.
- Some fields in `containers` are dependent on related settings. For example, CPU request must be less than CPU limit. If both fields are not properly configured, the pod may fail to start.
- Kubernetes lets you set both `requests` and `limits` for resources in your `PodSpec`. **GKE Autopilot** only considers `requests`. For more information, see Setting resource limits in Autopilot (/kubernetes-engine/docs/concepts/autopilot-resource-requests#resource-limits).

Additionally, certain fields are configurable by annotations on the pod, although it is recommended to use the above approach to customizing settings. Additional care must be taken for certain annotations:

- For GKE Standard, if `sidecar.istio.io/proxyCPU` is set, make sure to explicitly set `sidecar.istio.io/proxyCPULimit`. Otherwise the sidecar's CPU limit will be set as unlimited.
- For GKE Standard, if `sidecar.istio.io/proxyMemory` is set, make sure to explicitly set `sidecar.istio.io/proxyMemoryLimit`. Otherwise the sidecar's memory limit will be set as unlimited.
- For GKE Autopilot, configuring resource `requests` and `limits` using annotations might overprovision resources. Use the image template approach to avoid. See Resource modification examples in Autopilot (/kubernetes-engine/docs/concepts/autopilot-resource-requests#examples).

For example, see the below resources annotation configuration:

```
spec:
  template:
    metadata:
      annotations:
        sidecar.istio.io/proxyCPU: "200m"
        sidecar.istio.io/proxyCPULimit: "200m"
        sidecar.istio.io/proxyMemory: "256Mi"
        sidecar.istio.io/proxyMemoryLimit: "256Mi"
```

Verify control plane metrics

You can view the version of the control plane and data plane in Metrics Explorer.

To verify that your configuration works correctly:

1. In the Google Cloud console, view the control plane metrics:

   Go to Metrics Explorer (https://console.cloud.google.com/monitoring/metrics-explorer?pageStat

2. Choose your workspace and add a custom query using the following parameters:

   - **Resource type**: Kubernetes Container
   - **Metric**: Proxy Clients
   - **Filter**: `container_name="cr-`*`REVISION_LABEL`*`"`
   - **Group By**: revision label and proxy_version label
   - **Aggregator** sum
   - **Period**: 1 minute

   When you run Anthos Service Mesh with both a Google-managed and an in-cluster control plane, you can tell the metrics apart by their container name. For example, managed metrics have `container_name="cr-asm-managed"`, while unmanaged metrics have `container_name="discovery"`. To display metrics from both, remove the Filter on `container_name="cr-asm-managed"`.

3. Verify the control plane version and proxy version by inspecting the following fields in Metrics Explorer:

   - The **revision** field indicates the control plane version.

- The **proxy_version** field indicates the `proxy_version`.
- The **value** field indicates the number of connected proxies.

For the current channel to Anthos Service Mesh version mapping, see Anthos Service Mesh versions per channel (/service-mesh/docs/managed/select-a-release-channel#anthos_service_mesh_versions_per_channel)
.

Migrate applications to managed Anthos Service Mesh

Warning: During the migration process, Istio CRDs corresponding to the selected channel are installed in the specified cluster. If there are existing Istio CRDs in the cluster, they will be overwritten.

To migrate applications from in-cluster Anthos Service Mesh to managed Anthos Service Mesh, perform the following steps:

Note: Ensure you have provisioned managed Anthos Service Mesh by following all of the preceding sections of this guide.

1. Replace the current namespace label. The steps required depend on whether you wish to use default injection labels (/service-mesh/docs/managed/select-a-release-channel#how-to-select-a-release-channel) (for example, `istio-injection enabled`) or the revision label (/service-mesh/docs/glossary#revision).

Default injection labelRevision label (#revision-label)
(#default-injection-label)

a. Run the following command to move the default tag to the managed revision:

```
istioctl tag set default --revision REVISION_LABEL
```

b. Run the following command to label the namespace using `istio-injection=enabled`, if it wasn't already:

```
kubectl label namespace NAMESPACE istio-injection=enabled i
--overwrite
```

2. Perform a rolling upgrade of deployments in the namespace:

```
kubectl rollout restart deployment -n NAMESPACE
```

3. Test your application to verify that the workloads function correctly.

4. If you have workloads in other namespaces, repeat the previous steps for each namespace.

5. If you deployed the application in a multi-cluster setup, replicate the Kubernetes and Istio configuration in all clusters, unless there is a desire to limit that configuration to a subset of clusters only. The configuration applied to a particular cluster is the source of truth for that cluster.

6. Check that the metrics appear as expected by following the steps in Verify control plane metrics (#verify_control_plane_metrics).

If you are satisfied that your application works as expected, you can remove the in-cluster `istiod` after you switch all namespaces to the managed control plane, or keep them as a backup - `istiod` will automatically scale down to use fewer resources. To remove, skip to Delete old control plane (#delete-old-control-plane).

If you encounter problems, you can identify and resolve them by using the information in Resolving managed control plane issues (/service-mesh/docs/troubleshooting/troubleshoot-managed-service-mesh) and if necessary, roll back to the previous version.

Delete old control plane (#delete-old-control-plane) Roll back (#roll-back)

After you install and confirm that all namespaces use the Google-managed control plane, you can delete the old control plane.

```
kubectl delete Service,Deployment,HorizontalPodAutoscaler,PodDisruptionE
```

If you used **`istioctl kube-inject`** instead of automatic injection, or if you installed additional gateways, check the metrics for the control plane, and verify that the number of connected endpoints is zero.

Disable automatic management

Disabling automatic management does not deprovision any resources. All resources are left in the cluster for you to manually manage or remove, similar to the result of provisioning managed Anthos Service Mesh with **`asmcli`** (/service-mesh/docs/managed/provision-managed-anthos-service-mesh-asmcli).

1. Run the following command to disable automatic management:

```
gcloud container fleet mesh update \
   --management manual \
   --memberships MEMBERSHIP_NAME \
   --project FLEET_PROJECT_ID
```

2. After a few minutes, verify the status of control plane management is **`DISABLED`**:

```
gcloud container fleet mesh describe --project PROJECT_ID
```

The output is similar to:

```
...
membershipSpecs:
  projects/projectid/locations/global/memberships/cluster-name:
    mesh:
      controlPlane: management
membershipStates:
  projects/projectid/locations/global/memberships/cluster-name:
    servicemesh:
      controlPlaneManagement:
        state: DISABLED
    state:
      code: OK
      description: 'Revision(s) ready for use: asm-managed.'
...
```

Uninstall Anthos Service Mesh

To completely uninstall Anthos Service Mesh, see Uninstall Anthos Service Mesh (/service-mesh/docs/uninstall).

Troubleshooting

To identify and resolve problems when using managed control plane, see Resolving managed control plane issues (/service-mesh/docs/troubleshooting/troubleshoot-managed-service-mesh).

What's next

- Enable optional features on managed Anthos Service Mesh (/service-mesh/docs/managed/optional-features).
- Configuring transport security (/service-mesh/docs/security/configuring-mtls) to secure your mesh.

Last updated 2023-01-25 UTC.