**Install istioctl**

Install the istioctl binary with curl:

1. Download the latest release with the command:

$ curl -sL https://istio.io/downloadIstioctl | sh -

1. Add the istioctl client to your path, on a macOS or Linux system:

$ export PATH=$PATH:$HOME/.istioctl/bin

**Download Istio**

1. Go to the [Istio release](https://github.com/istio/istio/releases/tag/1.7.2) page to download the installation file for your OS, or download and extract the latest release automatically (Linux or macOS):

$ curl -L https://istio.io/downloadIstio | sh -

The command above downloads the latest release (numerically) of Istio. You can pass variables on the command line to download a specific version or to override the processor architecture. For example, to download Istio 1.6.8 for the x86\_64 architecture, run curl -L https://istio.io/downloadIstio | ISTIO\_VERSION=1.6.8 TARGET\_ARCH=x86\_64 sh -.

1. Move to the Istio package directory. For example, if the package is istio-1.7.2:

$ cd istio-1.7.2

The installation directory contains:

* + Sample applications in samples/
  + The [istioctl](https://istio.io/latest/docs/reference/commands/istioctl) client binary in the bin/ directory.

1. Add the istioctl client to your path (Linux or macOS):

$ export PATH=$PWD/bin:$PATH

**Install Istio**

1. For this installation, we use the demo [configuration profile](https://istio.io/latest/docs/setup/additional-setup/config-profiles/). It’s selected to have a good set of defaults for testing, but there are other profiles for production or performance testing.

$ istioctl profile list

$ istioctl profile dump demo

$ istioctl install --set profile=demo

*✔ Istio core installed*

*✔ Istiod installed*

*✔ Egress gateways installed*

*✔ Ingress gateways installed*

*✔ Installation complete*

**Istio Sidecar Injection**

**Manual**

**Automatic**

1. 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

*namespace/default labeled*

**Deploy the sample application**

1. Deploy the [Bookinfo sample application](https://istio.io/latest/docs/examples/bookinfo/):

$ kubectl apply -f [samples/bookinfo/platform/kube/bookinfo.yaml](https://raw.githubusercontent.com/istio/istio/release-1.7/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*

1. The application will start. As each pod becomes ready, the Istio sidecar will be deployed along with it.

$ kubectl get services

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *NAME* | *TYPE* | CLUSTER-IP | EXTERNAL-IP | PORT(S) | AGE |
| details | ClusterIP | 10.0.0.212 | <none> | 9080/TCP | 29s |
| kubernetes | ClusterIP | 10.0.0.1 | <none> | 443/TCP | 25m |
| productpage | ClusterIP | 10.0.0.57 | <none> | 9080/TCP | 28s |
| ratings | ClusterIP | 10.0.0.33 | <none> | 9080/TCP | 29s |
| reviews | ClusterIP | 10.0.0.28 | <none> | 9080/TCP | 29s |

and

$ kubectl get pods

*NAME READY STATUS RESTARTS AGE*

*details-v1-558b8b4b76-2llld 2/2 Running 0 2m41s*

*productpage-v1-6987489c74-lpkgl 2/2 Running 0 2m40s*

*ratings-v1-7dc98c7588-vzftc 2/2 Running 0 2m41s*

*reviews-v1-7f99cc4496-gdxfn 2/2 Running 0 2m41s*

*reviews-v2-7d79d5bd5d-8zzqd 2/2 Running 0 2m41s*

*reviews-v3-7dbcdcbc56-m8dph 2/2 Running 0 2m41s*

Re-run the previous command and wait until all pods report READY 2 / 2 and STATUS Running before you go to the next step. This might take a few minutes depending on your platform.

1. 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:
2. $ kubectl exec "$(kubectl get pod -l app=ratings -o jsonpath='{.items[0].metadata.name}')" -c ratings -- curl -s productpage:9080/productpage | grep -o "<title>.\*</title>"
3. *<title>Simple Bookstore App</title>*

**Open the application to outside traffic**

The Bookinfo application is deployed but not accessible from the outside. To make it accessible, you need to create an [Istio Ingress Gateway](https://istio.io/latest/docs/concepts/traffic-management/#gateways), which maps a path to a route at the edge of your mesh.

1. Associate this application with the Istio gateway:
2. $ kubectl apply -f [samples/bookinfo/networking/bookinfo-gateway.yaml](https://raw.githubusercontent.com/istio/istio/release-1.7/samples/bookinfo/networking/bookinfo-gateway.yaml)
3. *gateway.networking.istio.io/bookinfo-gateway created*
4. *virtualservice.networking.istio.io/bookinfo created*
5. Ensure that there are no issues with the configuration:
6. $ istioctl analyze
7. *✔ No validation issues found when analyzing namespace: default.*

**Determining the ingress IP and ports**

Follow these instructions to set the INGRESS\_HOST and INGRESS\_PORT variables for accessing the gateway. Use the tabs to choose the instructions for your chosen platform:

MinikubeOther platforms

Set the ingress ports:

$ export INGRESS\_PORT=$(kubectl -n istio-system get service istio-ingressgateway -o jsonpath='{.spec.ports[?(@.name=="http2")].nodePort}')

$ export SECURE\_INGRESS\_PORT=$(kubectl -n istio-system get service istio-ingressgateway -o jsonpath='{.spec.ports[?(@.name=="https")].nodePort}')

Ensure a port was successfully assigned to each environment variable:

$ echo "$INGRESS\_PORT"

*32194*

$ echo "$SECURE\_INGRESS\_PORT"

*31632*

Set the ingress IP:

$ export INGRESS\_HOST=$(minikube ip)

Ensure an IP address was successfully assigned to the environment variable:

$ echo "$INGRESS\_HOST"

*192.168.4.102*

Run this command in a new terminal window to start a Minikube tunnel that sends traffic to your Istio Ingress Gateway:

$ minikube tunnel

1. Set GATEWAY\_URL:
2. $ export GATEWAY\_URL=$INGRESS\_HOST:$INGRESS\_PORT
3. Ensure an IP address and port were successfully assigned to the environment variable:
4. $ echo "$GATEWAY\_URL"
5. *192.168.99.100:32194*

**Verify external access**

Confirm that the Bookinfo application is accessible from outside by viewing the Bookinfo product page using a browser.

1. Run the following command to retrieve the external address of the Bookinfo application.
2. $ echo http://"$GATEWAY\_URL/productpage"
3. Paste the output from the previous command into your web browser and confirm that the Bookinfo product page is displayed.

**View the dashboard**

Istio integrates with [several](https://istio.io/latest/docs/ops/integrations) different telemetry applications. These can help you gain an understanding of the structure of your service mesh, display the topology of the mesh, and analyze the health of your mesh.

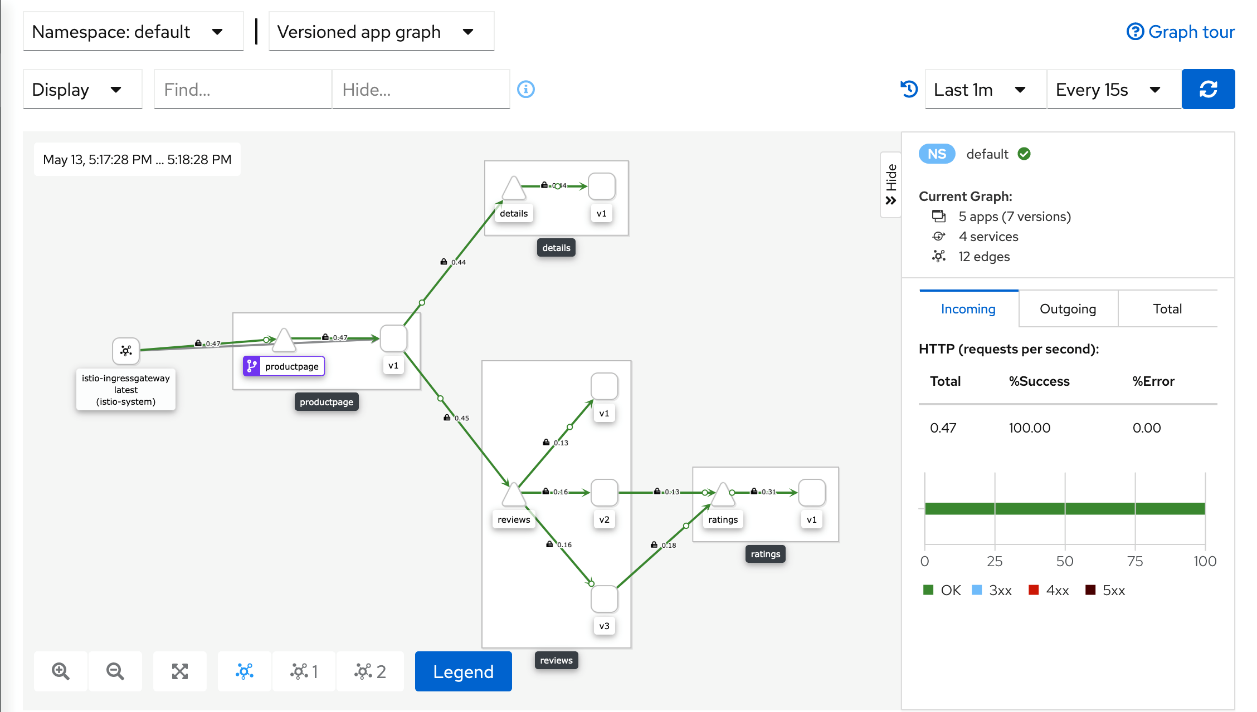
Use the following instructions to deploy the [Kiali](https://istio.io/latest/docs/ops/integrations/kiali/) dashboard, along with [Prometheus](https://istio.io/latest/docs/ops/integrations/prometheus/), [Grafana](https://istio.io/latest/docs/ops/integrations/grafana), and [Jaeger](https://istio.io/latest/docs/ops/integrations/jaeger/).

1. Install Kiali and wait for it to be deployed.
2. $ kubectl apply -f [samples/addons](https://raw.githubusercontent.com/istio/istio/release-1.7/samples/addons)
3. $ while ! kubectl wait --for=condition=available --timeout=600s deployment/kiali -n istio-system; do sleep 1; done

If there are errors trying to install the addons, try running the command again. There may be some timing issues which will be resolved when the command is run again.

1. Access the Kiali dashboard.
2. $ istioctl dashboard kiali
3. In the left navigation menu, select *Graph* and in the *Namespace* drop down, select *default*.

The Kiali dashboard shows an overview of your mesh with the relationships between the services in the Bookinfo sample application. It also provides filters to visualize the traffic flow.

[](https://istio.io/latest/docs/setup/getting-started/kiali-example2.png)

Kiali Dashboard