**Istio Getting Started Tutorial Documentation**

**Introduction**

This document walks through the official Istio "Getting Started" guide, covering environment preparation, Istio installation, deploying a sample application, and exploring Istio's core features like traffic management, observability, and security. Each section includes step-by-step instructions, command explanations, and key notes for successful implementation.

**1. Prerequisites**

Before starting, ensure your environment meets the following requirements:

* A Kubernetes cluster (Minikube, Kind, EKS, GKE, or AKS). For this tutorial, Minikube is recommended for local testing.
* kubectl command-line tool configured to interact with your Kubernetes cluster.
* Minimum cluster resources: 2 CPU cores, 4GB RAM, and 10GB free disk space.
* Internet access to download Istio binaries and container images.

**Note:** If using Minikube, start the cluster with sufficient resources using minikube start --cpus=2 --memory=4g.

**2. Download and Install Istio**

**2.1 Download the Istio CLI**

The Istio CLI (istioctl) is used to manage Istio service mesh. Follow these steps to download it:

1. Download the latest Istio release (replace <version> with the latest version, e.g., 1.21.0):  
    curl -L https://istio.io/downloadIstio | sh -This command downloads the Istio package into a directory named istio-<version>.
2. Add the istioctl binary to your system path:  
    cd istio-<version>  
   export PATH=$PWD/bin:$PATH
3. Verify the installation:  
    istioctl versionThe output should display the istioctl version (cluster version will show "unavailable" until Istio is installed on the cluster).

**2.2 Install Istio on the Kubernetes Cluster**

Use istioctl install with the demo profile, which includes all Istio components for testing and demonstration:

|  |
| --- |
| bash istioctl install --set profile=demo -y |

**What the demo profile includes:**

* Istiod (control plane: service discovery, configuration, certificate management)
* Ingress Gateway (for external traffic entry)
* Egress Gateway (for controlled external service access)
* Prometheus, Grafana, Jaeger, and Kiali (observability tools)

Wait for all Istio pods to be in Running status:  
 kubectl get pods -n istio-systemExpected output shows pods like istiod-xxxxxxxxx-xxxxx, istio-ingressgateway-xxxxxxxxx-xxxxx, and observability tool pods.

**3. Deploy the Sample Application (Bookinfo)**

The Bookinfo application is a multi-service demo used to showcase Istio's features. It consists of four services: productpage, details, reviews (with three versions), and ratings.

**3.1 Label the Namespace for Sidecar Injection**

Istio uses sidecar proxies (Envoy) to manage traffic. To automatically inject sidecars into pods, label the namespace (we'll use default for simplicity):

|  |
| --- |
| bash kubectl label namespace default istio-injection=enabled |

Verify the label:  
 kubectl get namespace default -L istio-injection

**3.2 Deploy the Bookinfo Application**

Use Istio's sample manifest to deploy Bookinfo:

|  |
| --- |
| bash kubectl apply -f samples/bookinfo/platform/kube/bookinfo.yaml |

Check that all services and pods are running:  
 kubectl get services  
kubectl get podsEach pod should have two containers: the application container and the istio-proxy sidecar.

**3.3 Verify the Application is Running**

Test access to the productpage service from within the cluster:  
 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>"The output should be <title>Simple Bookstore App</title>, confirming the application is working.

**4. Expose the Application with Istio Ingress Gateway**

To make the Bookinfo application accessible from outside the cluster, create an Istio Gateway and VirtualService:

**4.1 Apply the Ingress Configuration**

|  |
| --- |
| bash kubectl apply -f samples/bookinfo/networking/bookinfo-gateway.yaml |

This configuration:  
 Creates a Gateway named bookinfo-gateway that listens on port 80.Creates a VirtualService that routes traffic from the gateway to the productpage service.

**4.2 Verify the Ingress Configuration**

|  |
| --- |
| bash istioctl analyze |

Ensure there are no errors in the configuration. The output should show "No issues found when analyzing namespace 'default'."

**4.3 Access the Application from Outside the Cluster**

Get the Istio Ingress Gateway URL:  
 export INGRESS\_HOST=$(minikube ip) # For Minikube; use appropriate command for other clusters  
export INGRESS\_PORT=$(kubectl -n istio-system get service istio-ingressgateway -o jsonpath='{.spec.ports[?(@.name=="http2")].nodePort}')  
export GATEWAY\_URL=$INGRESS\_HOST:$INGRESS\_PORTOpen http://$GATEWAY\_URL/productpage in your browser. You should see the Bookinfo product page.

**Note:** For cloud-based clusters (EKS/GKE/AKS), use kubectl -n istio-system get service istio-ingressgateway -o jsonpath='{.status.loadBalancer.ingress[0].hostname}' to get the INGRESS\_HOST.

**5. Explore Istio's Core Features**

**5.1 Traffic Management: Route to Specific Service Versions**

Configure Istio to route all traffic to reviews:v1 (no star ratings):  
 kubectl apply -f samples/bookinfo/networking/virtual-service-all-v1.yamlRefresh the Bookinfo product page—you should see "Reviews: There are no reviews yet." (no stars).

Now route traffic from a specific user (e.g., jason) to reviews:v2 (with black stars):  
 kubectl apply -f samples/bookinfo/networking/virtual-service-reviews-test-v2.yamlLog in as user jason (no password) on the product page—you’ll see black stars. Other users still see v1.

**5.2 Observability: Access Monitoring and Tracing Tools**

Istio integrates with Prometheus, Grafana, and Jaeger for observability. Use istioctl dashboard to access these tools:

* Grafana Dashboard (metrics visualization):  
   istioctl dashboard grafanaNavigate to "Istio Service Mesh Dashboard" to view metrics like request latency, success rate, and traffic volume.
* Jaeger Dashboard (distributed tracing):  
   istioctl dashboard jaegerRefresh the Bookinfo product page a few times, then search for traces in Jaeger to see the request flow across services.

**5.3 Security: Enable Mutual TLS**

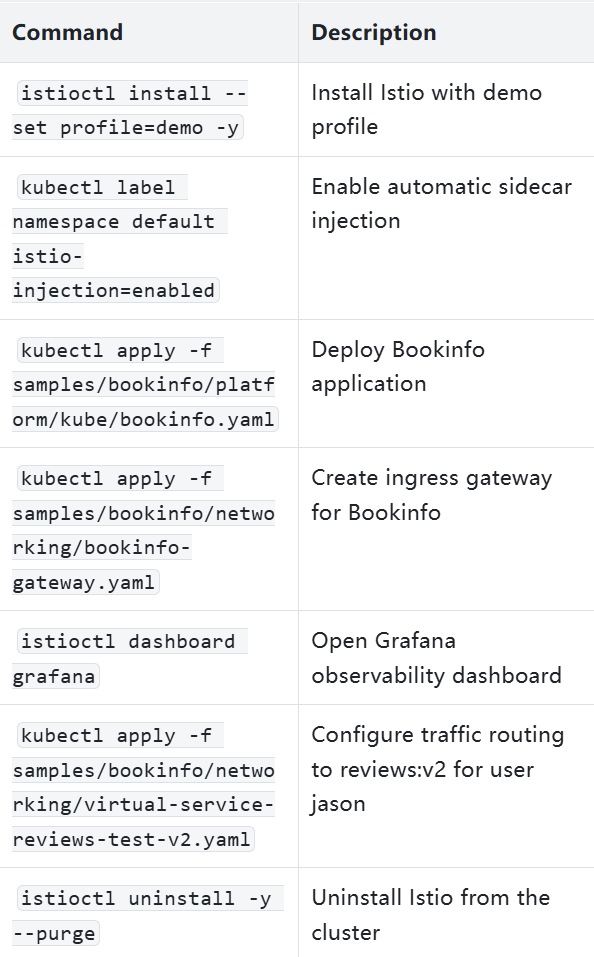
Istio can enforce mutual TLS (mTLS) between services, ensuring secure communication:  
 kubectl apply -f samples/bookinfo/networking/peerauthentication-default.yamlThis policy enables mTLS for all services in the default namespace. Verify with:  
 kubectl get peerauthentication -o yaml

**6. Cleanup Resources**

To remove the Bookinfo application and Istio from your cluster:

1. Delete the Bookinfo resources:  
    samples/bookinfo/platform/kube/cleanup.sh
2. Remove the namespace label for sidecar injection:  
    kubectl label namespace default istio-injection-
3. Uninstall Istio:  
    istioctl uninstall -y --purge  
   kubectl delete namespace istio-system
4. Stop Minikube (if used):  
    minikube stop  
   minikube delete

**Key Commands Summary**



**Conclusion**

This tutorial covered the basics of Istio, including installation, deploying a sample application, configuring ingress, managing traffic, and exploring observability and security features. For more advanced topics like fault injection, circuit breaking, or production-grade installation, refer to the [official Istio documentation](https://istio.io/latest/docs/).