

Request Routing



Before you begin

About this task

Route to version 1

Test the new routing configuration

Route based on user identity

Understanding what happened

Cleanup

See also

This task shows you how to route requests dynamically to multiple versions of a microservice.

Istio includes beta support for the Kubernetes Gateway API⁵ and intends to make it the default API for traffic management in the future⁶. The following instructions allow you to choose to use either the Gateway API or the Istio configuration API when configuring traffic management in the mesh. Follow instructions under either the Gateway API or Istio APIs tab, according to your preference.

Note that this document uses the Gateway API to configure internal mesh (east-west) traffic, i.e., not just ingress (north-south) traffic. Configuring internal mesh traffic is an experimental feature of the Gateway API, currently under development. If using the Gateway API instructions, before proceeding make sure to:

1. Install the **experimental version** of the Gateway API CRDs:

```
$ kubectl kustomize "github.com/kubernetes-sigs/gateway-api/config/crd/experimental?ref=v1.0.0" | kub
```

2. Configure Istio to read the alpha Gateway API resources by setting the PILOT_ENABLE_ALPHA_GATEWAY_API environment variable to true when installing Istio:

```
$ istioctl install --set values.pilot.env.PILOT_ENABLE_ALPHA_GATEWAY_API=true --set profile=minimal -
```

Before you begin

- Setup Istio by following the instructions in the Installation guide⁷.
- Deploy the Bookinfo⁸ sample application.
- Review the Traffic Management⁹ concepts doc.

About this task

The Istio Bookinfo⁸ sample consists of four separate microservices, each with multiple versions. Three different versions of one of the microservices, reviews, have been deployed and are running concurrently. To illustrate the problem this causes, access the Bookinfo app's /productpage in a browser and refresh several times. The URL is http://\$GATEWAY_URL/productpage, where \$GATEWAY_URL is the External IP address of the ingress, as explained in the Bookinfo doc.

You'll notice that sometimes the book review output contains star ratings and other times it does not. This is because without an explicit default service version to route to, Istio routes requests to all available versions in a round robin fashion.

The initial goal of this task is to apply rules that route all traffic to v1 (version 1) of the microservices. Later, you will apply a rule to route traffic based on the value of an HTTP request header.

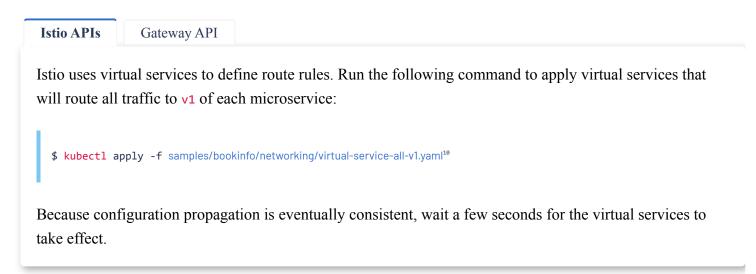
Route to version 1

To route to one version only, you configure route rules that send traffic to default versions for the microservices.



If you haven't already, follow the instructions in define the service versions.

1. Run the following command to create the route rules:



2. Display the defined routes with the following command:



```
$ kubectl get virtualservices -o yaml
- apiVersion: networking.istio.io/v1beta1
 kind: VirtualService
 spec:
   hosts:
   - details
   http:
    - route:
     - destination:
         host: details
         subset: v1
- apiVersion: networking.istio.io/v1beta1
 kind: VirtualService
 spec:
   hosts:
   - productpage
   http:
   - route:
     - destination:
         host: productpage
         subset: v1
- apiVersion: networking.istio.io/v1beta1
 kind: VirtualService
 . . .
 spec:
   hosts:
   - ratings
   http:
    - route:
     - destination:
         host: ratings
         subset: v1
- apiVersion: networking.istio.io/v1beta1
 kind: VirtualService
 spec:
   hosts:
   - reviews
   http:
    - route:
      - destination:
         host: reviews
          subset: v1
```

You can also display the corresponding subset definitions with the following command:

```
$ kubectl get destinationrules -o yaml
```

You have configured Istio to route to the v1 version of the Bookinfo microservices, most importantly the reviews service version 1.

Test the new routing configuration

You can easily test the new configuration by once again refreshing the /productpage of the Bookinfo app in your browser. Notice that the reviews part of the page displays with no rating stars, no matter how many times you refresh. This is because you configured Istio to route all traffic for the reviews service to the version reviews:v1 and this version of the service does not access the star ratings service.

You have successfully accomplished the first part of this task: route traffic to one version of a service.

Route based on user identity

Next, you will change the route configuration so that all traffic from a specific user is routed to a specific service version. In this case, all traffic from a user named Jason will be routed to the service reviews:v2.

This example is enabled by the fact that the productpage service adds a custom end-user header to all outbound HTTP requests to the reviews service.

Istio also supports routing based on strongly authenticated JWT on ingress gateway, refer to the JWT claim based routing¹¹ for more details.

Remember, reviews:v2 is the version that includes the star ratings feature.

1. Run the following command to enable user-based routing:

```
| $ kubectl apply -f samples/bookinfo/networking/virtual-service-reviews-test-v2.yaml<sup>12</sup>

You can confirm the rule is created using the following command:
```

```
$ kubectl get virtualservice reviews -o yaml
apiVersion: networking.istio.io/v1beta1
kind: VirtualService
spec:
 hosts:
  - reviews
 http:
  - match:
    - headers:
       end-user:
          exact: jason
   route:
    - destination:
        host: reviews
        subset: v2
  - route:
    - destination:
       host: reviews
        subset: v1
```

2. On the /productpage of the Bookinfo app, log in as user jason.

Refresh the browser. What do you see? The star ratings appear next to each review.

3. Log in as another user (pick any name you wish).

Refresh the browser. Now the stars are gone. This is because traffic is routed to reviews:v1 for all users except Jason.

You have successfully configured Istio to route traffic based on user identity.

Understanding what happened

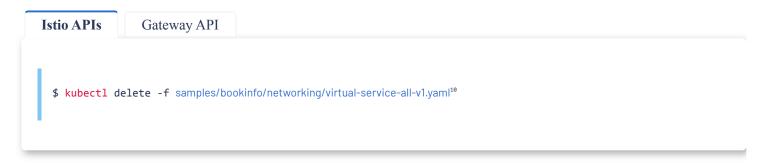
In this task, you used Istio to send 100% of the traffic to the v1 version of each of the Bookinfo services. You then set a rule to selectively send traffic to version v2 of the reviews service based on a custom end-user header added to the request by the productpage service.

Note that Kubernetes services, like the Bookinfo ones used in this task, must adhere to certain restrictions to take advantage of Istio's L7 routing features. Refer to the Requirements for Pods and Services¹³ for details.

In the traffic shifting¹⁴ task, you will follow the same basic pattern you learned here to configure route rules to gradually send traffic from one version of a service to another.

Cleanup

1. Remove the application route rules:



2. If you are not planning to explore any follow-on tasks, refer to the Bookinfo cleanup instructions to shutdown the application.

Links

- 1. https://istio.io/latest/docs/
- 2. https://istio.io/latest/docs/tasks/
- 3. https://istio.io/latest/docs/tasks/traffic-management/
- 4. https://github.com/istio/istio.io/tree/master/README.md#testing-document-content
- 5. https://gateway-api.sigs.k8s.io/
- 6. https://istio.io/latest/blog/2022/gateway-api-beta/
- 7. https://istio.io/latest/docs/setup/
- 8. https://istio.io/latest/docs/examples/bookinfo/
- 9. https://istio.io/latest/docs/concepts/traffic-management
- 10. https://raw.githubusercontent.com/istio/istio/release-1.20/samples/bookinfo/networking/virtual-service-all-v1.yaml
- 11. https://istio.io/latest/docs/tasks/security/authentication/jwt-route
- 12. https://raw.githubusercontent.com/istio/istio/release-1.20/samples/bookinfo/networking/virtual-service-reviews-test-v2.yaml
- 13. https://istio.io/latest/docs/ops/deployment/requirements/
- 14. https://istio.io/latest/docs/tasks/traffic-management/traffic-shifting