Working with Istio

This document contains the exercises for installing and using Istio 1.0.x. The units for these exercises are:

- Installing Istio
- Activating the bookinfo application
- Working with mixer configurations
- Configuring Istio networking
- Configuring external service
- Working with RBAC definitions

NOTE:

This set of lab exercises can be performed in either IBM Cloud Private 2.1.0.3 or IBM Cloud Public using the IBM Kubernetes Service. Where there is a difference, the command is given for each environment. Only use the command for your environment.

For IBM Cloud Private, some of the instructions are specific to version 2.1.0.3. If you are unsure of the version of your IBM Cloud Private version, please ask your instructor.

The lab instructions are written for the Istio 1.0.0 release, but they should work for any subsequent 1.0.x release.

Installing Istio

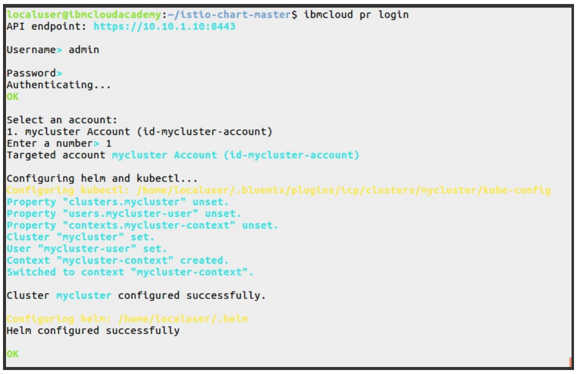
This exercise assumes that you have already:

- Installed the appropriate CLI for your Kubernetes environment
- Installed and initialized your environment for the `helm` command, including having `tiller` installed
- Set up the `kubectl` command line for accessing your cluster
- If you are using IBM Cloud Private, it is assumed that you have completed the steps included in the document "Initial Setup of IBM Cloud Private". If you do not have the document, please ask your instructor.

These steps will install Istio 1.0.0 into your Kubernetes environment:

1. Login and setup the environment for working with your cluster.

- If you are working in IBM Cloud Private, use the `ibmcloud pr login` command to login to the IBM Cloud Private CLI. The API endpoint is `https://10.10.1.10:8443`. The user id / password is `admin` / `passw0rd`. The account is `mycluster`.

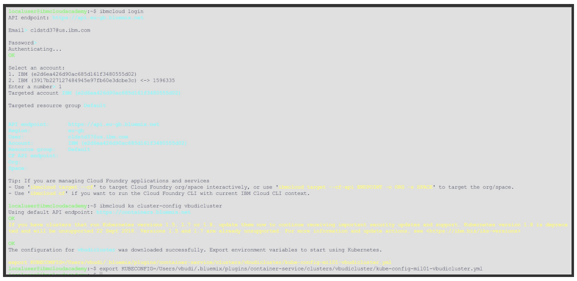


```
localuser@ibmcloudacademy:~/istio-chart-master$ ibmcloud pr login
API endpoint: https://10.10.1.10:8443

Username> admin

Password>
Authenticating...
OK

Select an account:
1. mycluster Account (id-mycluster-account)
Enter a number> 1
Targeted account mycluster Account (id-mycluster-account)

Configuring helm and kubectl...
Configuring kubectl: /home/localuser/.bluemix/plugins/icp/clusters/mycluster/kube-config
Property "clusters.mycluster" unset.
Property "users.mycluster-user" unset.
Property "contexts.mycluster-context" unset.
Cluster "mycluster" set.
User "mycluster-user" set.
Context "mycluster-context" created.
Switched to context "mycluster-context".

Cluster mycluster configured successfully.

Configuring helm: /home/localuser/.helm
Helm configured successfully

OK
```

- If you are working in IBM Cloud Public, login to the IBM Kubernetes Services using the `ibmcloud login` command with the IBM Cloud user ID and password you have been provided. Use the

  ```
  `ibmcloud ks cluster-config <clustername> | grep
  KUBECONFIG`
  ```

  command with the back-ticks to set the KUBECONFIG environment variable.



```
localuser@ibmcloudacademy:~$ ibmcloud login
API endpoint: https://api.eu-gb.bluemix.net

Email> cldstd37@us.ibm.com

Password>
Authenticating...
OK

Select an account:
1. IBM (e2d6ea426d90ac685d161f3480555d02)
2. IBM (3917b227127484945e97fb60e3dcbe3c) <-> 1596335
Enter a number> 1
Targeted account IBM (e2d6ea426d90ac685d161f3480555d02)

Targeted resource group Default


API endpoint:      https://api.eu-gb.bluemix.net
Region:            eu-gb
User:              cldstd37@us.ibm.com
Account:           IBM (e2d6ea426d90ac685d161f3480555d02)
Resource group:    Default
CF API endpoint:
Org:
Space:

Tip: If you are managing Cloud Foundry applications and services
- Use 'ibmcloud target --cf' to target Cloud Foundry org/space interactively, or use 'ibmcloud target --cf-api ENDPOINT -o ORG -s SPACE' to target the org/space.
- Use 'ibmcloud cf' if you want to run the Cloud Foundry CLI with current IBM Cloud CLI context.

localuser@ibmcloudacademy:~$ ibmcloud ks cluster-config vbudicluster
Using default API endpoint: https://containers.bluemix.net
OK
If you have clusters that run Kubernetes versions 1.5, 1.7 or 1.8, update them now to continue receiving important security updates and support. Kubernetes version 1.8 is depreca
ted and will be unsupported 22 Sept 2018. Versions 1.5 and 1.7 are already unsupported. For more information and update actions, see <https://ibm.biz/iks-versions>

OK
The configuration for vbudicluster was downloaded successfully. Export environment variables to start using Kubernetes.

export KUBECONFIG=/Users/vbudi/.bluemix/plugins/container-service/clusters/vbudicluster/kube-config-mil01-vbudicluster.yml
localuser@ibmcloudacademy:~$ export KUBECONFIG=/Users/vbudi/.bluemix/plugins/container-service/clusters/vbudicluster/kube-config-mil01-vbudicluster.yml
localuser@ibmcloudacademy:~$
```

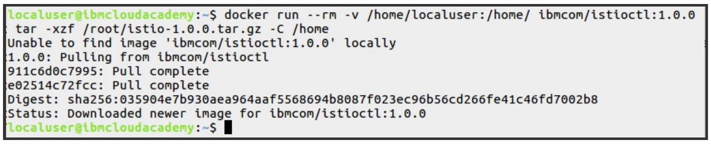
2. Download the Istio zip file for some definitions and executables that are needed to setup Istio later.

   Note: This download is only valid if you are using a Linux client. For other platforms, use the download from the `https://istio.io` page. The filename and path may differ slightly.

   If you are on IBM Cloud Private, run the following command.

   ```
   docker run --rm -v /home/localuser:/home ibmcom/istioctl:1.0.0 tar
   ```

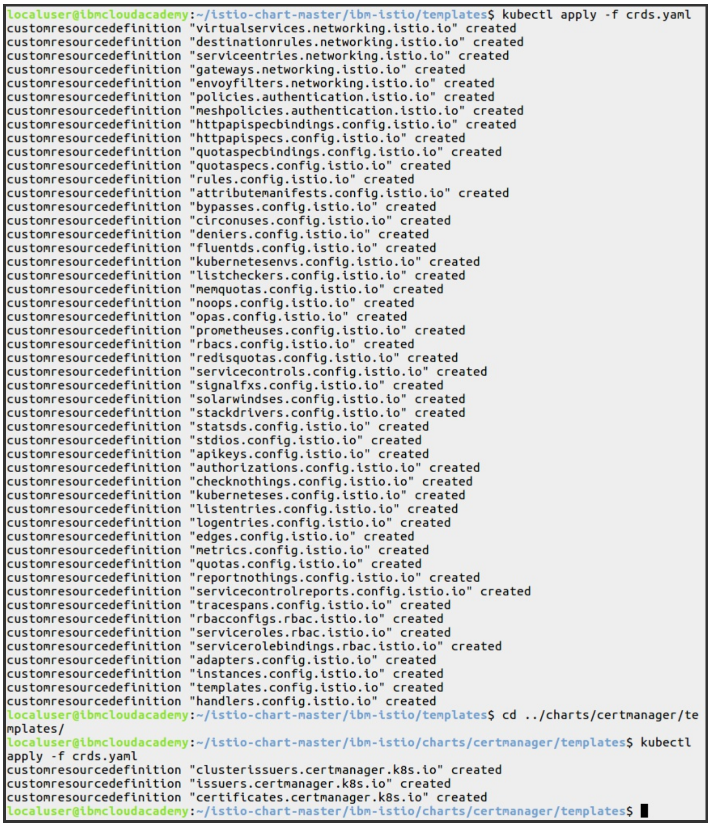
```
-xzf /root/istio-1.0.0.tar.gz -C /home
```



```
localuser@ibmcloudacademy:~$ docker run --rm -v /home/localuser:/home/ ibmcom/istioctl:1.0.0
 tar -xzf /root/istio-1.0.0.tar.gz -C /home
Unable to find image 'ibmcom/istioctl:1.0.0' locally
1.0.0: Pulling from ibmcom/istioctl
911c6d0c7995: Pull complete
e02514c72fcc: Pull complete
Digest: sha256:035904e7b930aea964aaf5568694b8087f023ec96b56cd266fe41c46fd7002b8
Status: Downloaded newer image for ibmcom/istioctl:1.0.0
localuser@ibmcloudacademy:~$ 
```

3. If you are using an IBM Cloud Private release below 3.1, setup the Kubernetes cluster for Istio (this is only needed for helm releases below 2.10).

```
 kubectl apply -f istio-
1.0.0/install/kubernetes/helm/istio/templates/crds.yaml
```



```
localuser@ibmcloudacademy:~/istio-chart-master/ibm-istio/templates$ kubectl apply -f crds.yaml
customresourcedefinition "virtualservices.networking.istio.io" created
customresourcedefinition "destinationrules.networking.istio.io" created
customresourcedefinition "serviceentries.networking.istio.io" created
customresourcedefinition "gateways.networking.istio.io" created
customresourcedefinition "envoyfilters.networking.istio.io" created
customresourcedefinition "policies.authentication.istio.io" created
customresourcedefinition "meshpolicies.authentication.istio.io" created
customresourcedefinition "httpapispecbindings.config.istio.io" created
customresourcedefinition "httpapispecs.config.istio.io" created
customresourcedefinition "quotaspecbindings.config.istio.io" created
customresourcedefinition "quotaspecs.config.istio.io" created
customresourcedefinition "rules.config.istio.io" created
customresourcedefinition "attributemanifests.config.istio.io" created
customresourcedefinition "bypasses.config.istio.io" created
customresourcedefinition "circonuses.config.istio.io" created
customresourcedefinition "deniers.config.istio.io" created
customresourcedefinition "fluentds.config.istio.io" created
customresourcedefinition "kubernetesenvs.config.istio.io" created
customresourcedefinition "listcheckers.config.istio.io" created
customresourcedefinition "memquotas.config.istio.io" created
customresourcedefinition "noops.config.istio.io" created
customresourcedefinition "opas.config.istio.io" created
customresourcedefinition "prometheuses.config.istio.io" created
customresourcedefinition "rbacs.config.istio.io" created
customresourcedefinition "redisquotas.config.istio.io" created
customresourcedefinition "servicecontrols.config.istio.io" created
customresourcedefinition "signalfxs.config.istio.io" created
customresourcedefinition "solarwindses.config.istio.io" created
customresourcedefinition "stackdrivers.config.istio.io" created
customresourcedefinition "statsds.config.istio.io" created
customresourcedefinition "stdios.config.istio.io" created
customresourcedefinition "apikeys.config.istio.io" created
customresourcedefinition "authorizations.config.istio.io" created
customresourcedefinition "checknothings.config.istio.io" created
customresourcedefinition "kuberneteses.config.istio.io" created
customresourcedefinition "listentries.config.istio.io" created
customresourcedefinition "logentries.config.istio.io" created
customresourcedefinition "edges.config.istio.io" created
customresourcedefinition "metrics.config.istio.io" created
customresourcedefinition "quotas.config.istio.io" created
customresourcedefinition "reportnothings.config.istio.io" created
customresourcedefinition "servicecontrolreports.config.istio.io" created
customresourcedefinition "tracespans.config.istio.io" created
customresourcedefinition "rbacconfigs.rbac.istio.io" created
customresourcedefinition "serviceroles.rbac.istio.io" created
customresourcedefinition "servicerolebindings.rbac.istio.io" created
customresourcedefinition "adapters.config.istio.io" created
customresourcedefinition "instances.config.istio.io" created
customresourcedefinition "templates.config.istio.io" created
customresourcedefinition "handlers.config.istio.io" created
localuser@ibmcloudacademy:~/istio-chart-master/ibm-istio/templates$ cd ../charts/certmanager/te
mplates/
localuser@ibmcloudacademy:~/istio-chart-master/ibm-istio/charts/certmanager/templates$ kubectl
apply -f crds.yaml
customresourcedefinition "clusterissuers.certmanager.k8s.io" created
customresourcedefinition "issuers.certmanager.k8s.io" created
customresourcedefinition "certificates.certmanager.k8s.io" created
localuser@ibmcloudacademy:~/istio-chart-master/ibm-istio/charts/certmanager/templates$ 
```

4. Create the istio-system namespace.

```
 kubectl create namespace istio-system
```

5. Setup the istio helm chart repository. We are using the IBM flavor of the helm chart.

```
 helm repo add ibm-charts
https://raw.githubusercontent.com/IBM/charts/master/repo/stable/
```

6. Deploy Istio using helm. Enable `tracing` and `grafana` as you will use them in this exercise. If you are working in IBM Cloud Private, issue the following command. **Note**: this is a long command that must be run in a single line.

```
 helm install ibm-charts/ibm-istio --tls --name istio --namespace
istio-system --set tracing.enabled=true --set grafana.enabled=true
--set pilot.resource.request.cpu=100m --set
pilot.resource.request.memory=1024Mi
```

If you are working in IBM Cloud Public using the IBM Kubernetes Service, issue the following command (same as the command above but missing the `--tls` option).

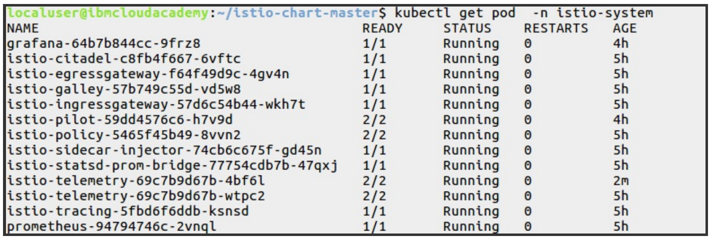
```
helm install ibm-charts/ibm-istio --name istio --namespace istio-
system --set tracing.enabled=true --set grafana.enabled=true --set
pilot.resource.request.cpu=100m --set
pilot.resource.request.memory=1024Mi
```

Note: If the deploy failed, you must clean the helm installation using the command `helm delete istio --purge --tls`

5. Check result of the deployment.

```
 kubectl get pod -n istio-system
```

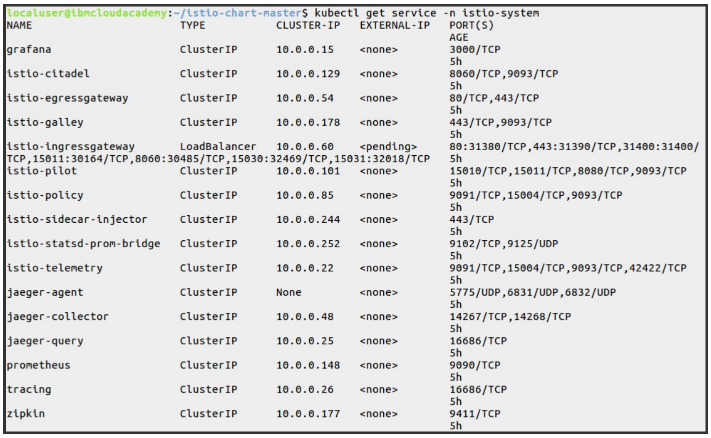
Make sure that the Istio system pods are running.



```
localuser@ibmcloudacademy:~/istio-chart-master$ kubectl get pod  -n istio-system
NAME                                         READY     STATUS    RESTARTS   AGE
grafana-64b7b844cc-9frz8                     1/1       Running   0          4h
istio-citadel-c8fb4f667-6vftc                1/1       Running   0          5h
istio-egressgateway-f64f49d9c-4gv4n          1/1       Running   0          5h
istio-galley-57b749c55d-vd5w8                1/1       Running   0          5h
istio-ingressgateway-57d6c54b44-wkh7t        1/1       Running   0          5h
istio-pilot-59dd4576c6-h7v9d                 2/2       Running   0          4h
istio-policy-5465f45b49-8vvn2                2/2       Running   0          5h
istio-sidecar-injector-74cb6c675f-gd45n      1/1       Running   0          5h
istio-statsd-prom-bridge-77754cdb7b-47qxj    1/1       Running   0          5h
istio-telemetry-69c7b9d67b-4bf6l             2/2       Running   0          2m
istio-telemetry-69c7b9d67b-wtpc2             2/2       Running   0          5h
istio-tracing-5fbd6f6ddb-ksnsd               1/1       Running   0          5h
prometheus-94794746c-2vnql                   1/1       Running   0          5h
```

6. Check the created services for Istio.

```
kubectl get service -n istio-system
```



```
localuser@ibmcloudacademy:~/istio-chart-master$ kubectl get service -n istio-system
NAME                       TYPE           CLUSTER-IP   EXTERNAL-IP   PORT(S)
                                                                     AGE
grafana                    ClusterIP      10.0.0.15    <none>        3000/TCP
                                                                     5h
istio-citadel              ClusterIP      10.0.0.129   <none>        8060/TCP,9093/TCP
                                                                     5h
istio-egressgateway        ClusterIP      10.0.0.54    <none>        80/TCP,443/TCP
                                                                     5h
istio-galley               ClusterIP      10.0.0.178   <none>        443/TCP,9093/TCP
                                                                     5h
istio-ingressgateway       LoadBalancer   10.0.0.60    <pending>     80:31380/TCP,443:31390/TCP,31400:31400/
TCP,15011:30164/TCP,8060:30485/TCP,15030:32469/TCP,15031:32018/TCP   5h
istio-pilot                ClusterIP      10.0.0.101   <none>        15010/TCP,15011/TCP,8080/TCP,9093/TCP
                                                                     5h
istio-policy               ClusterIP      10.0.0.85    <none>        9091/TCP,15004/TCP,9093/TCP
                                                                     5h
istio-sidecar-injector     ClusterIP      10.0.0.244   <none>        443/TCP
                                                                     5h
istio-statsd-prom-bridge   ClusterIP      10.0.0.252   <none>        9102/TCP,9125/UDP
                                                                     5h
istio-telemetry            ClusterIP      10.0.0.22    <none>        9091/TCP,15004/TCP,9093/TCP,42422/TCP
                                                                     5h
jaeger-agent               ClusterIP      None         <none>        5775/UDP,6831/UDP,6832/UDP
                                                                     5h
jaeger-collector           ClusterIP      10.0.0.48    <none>        14267/TCP,14268/TCP
                                                                     5h
jaeger-query               ClusterIP      10.0.0.25    <none>        16686/TCP
                                                                     5h
prometheus                 ClusterIP      10.0.0.148   <none>        9090/TCP
                                                                     5h
tracing                    ClusterIP      10.0.0.26    <none>        16686/TCP
                                                                     5h
zipkin                     ClusterIP      10.0.0.177   <none>        9411/TCP
                                                                     5h
```

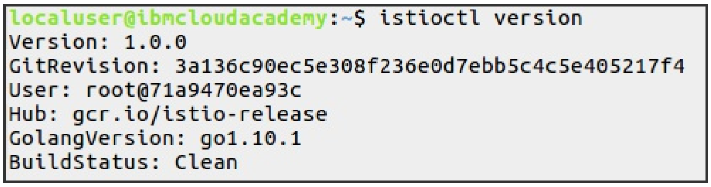
What ports are mapped to the Istio-ingressgateway?

- Port 80: `____________`
- Port 443: `____________`

7. Add `istioctl` to the program path.

```
sudo mv ~/istio-1.0.0/bin/istioctl /usr/local/bin/istioctl
```

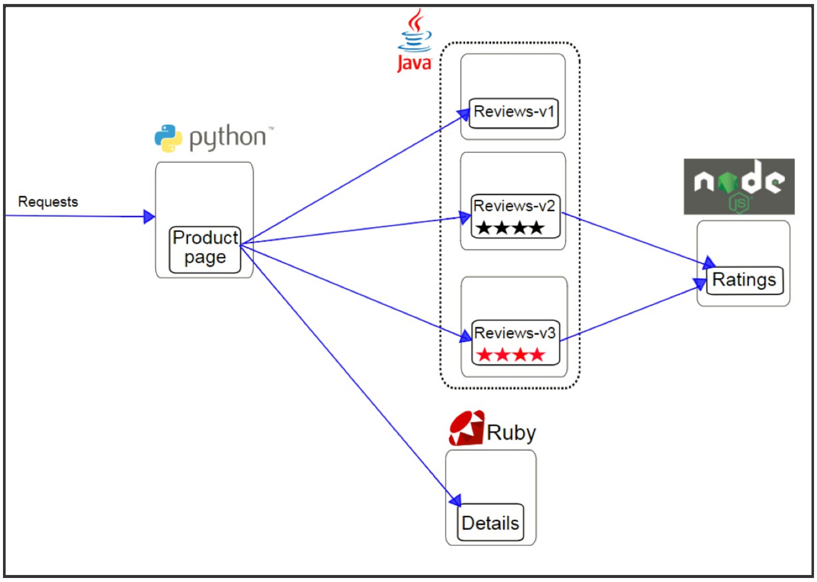
Check it using the command `istioctl version`.



```
localuser@ibmcloudacademy:~$ istioctl version
Version: 1.0.0
GitRevision: 3a136c90ec5e308f236e0d7ebb5c4c5e405217f4
User: root@71a9470ea93c
Hub: gcr.io/istio-release
GolangVersion: go1.10.1
BuildStatus: Clean
```

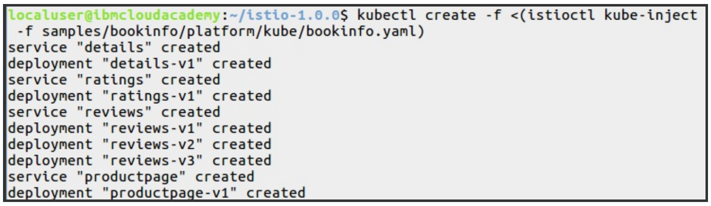
Activating the bookinfo application

We will use the bookinfo sample application. The application structure is as shown here:



1. The bookinfo sample application comes with the Istio distribution. Here are the steps to install it.

```
 cd ~/istio-1.0.0
 kubectl create -f <(istioctl kube-inject -f
samples/bookinfo/platform/kube/bookinfo.yaml)
```

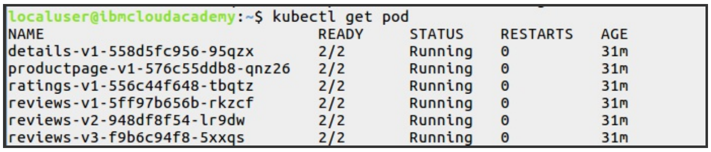


```
localuser@ibmcloudacademy:~/istio-1.0.0$ kubectl create -f <(istioctl kube-inject
 -f samples/bookinfo/platform/kube/bookinfo.yaml)
service "details" created
deployment "details-v1" created
service "ratings" created
deployment "ratings-v1" created
service "reviews" created
deployment "reviews-v1" created
deployment "reviews-v2" created
deployment "reviews-v3" created
service "productpage" created
deployment "productpage-v1" created
```

2. Check the deployed pods:

```
 kubectl get pod
```

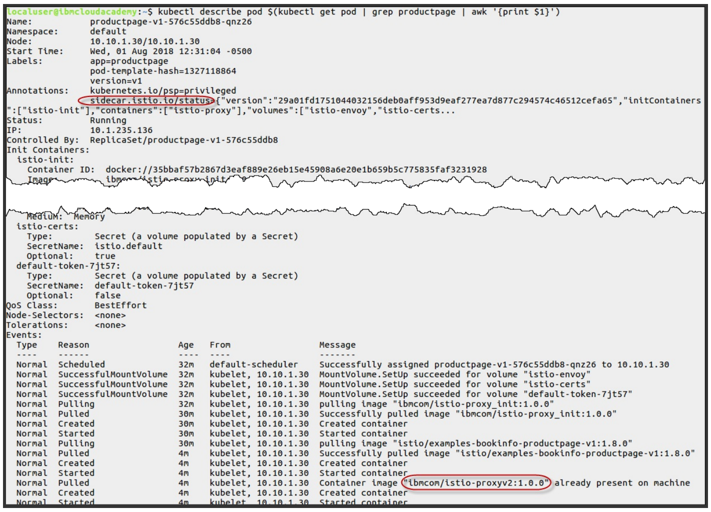
Make sure that the pods are running before continuing with the lab.



```
localuser@ibmcloudacademy:~$ kubectl get pod
NAME                                READY     STATUS    RESTARTS   AGE
details-v1-558d5fc956-95qzx         2/2       Running   0          31m
productpage-v1-576c55ddb8-qnz26     2/2       Running   0          31m
ratings-v1-556c44f648-tbqtz         2/2       Running   0          31m
reviews-v1-5ff97b656b-rkzcf         2/2       Running   0          31m
reviews-v2-948df8f54-lr9dw          2/2       Running   0          31m
reviews-v3-f9b6c94f8-5xxqs          2/2       Running   0          31m
```

3. Check whether the `istio-proxy` sidecars are running with the pods. As you deploy them with a single command, it is sufficient to check only one of them. This example checks the deployment of the `productpage` pod.

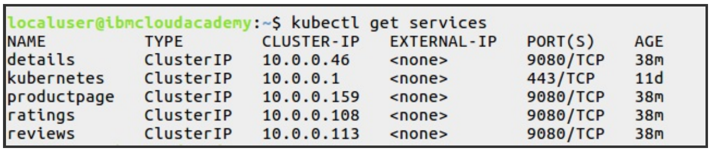
```
 kubectl describe pod $(kubectl get pod | grep productpage | awk
'{print $1}')
```



```
localuser@ibmcloudacademy:~$ kubectl describe pod $(kubectl get pod | grep productpage | awk '{print $1}')
Name:           productpage-v1-576c55ddb8-qnz26
Namespace:      default
Node:           10.10.1.30/10.10.1.30
Start Time:     Wed, 01 Aug 2018 12:31:04 -0500
Labels:         app=productpage
                pod-template-hash=1327118864
                version=v1
Annotations:    kubernetes.io/psp=privileged
                sidecar.istio.io/status={"version":"29a01fd1751044032156deb0aff953d9eaf277ea7d877c294574c46512cefa65","initContainers
":["istio-init"],"containers":["istio-proxy"],"volumes":["istio-envoy","istio-certs...
Status:         Running
IP:             10.1.235.136
Controlled By:  ReplicaSet/productpage-v1-576c55ddb8
Init Containers:
  istio-init:
    Container ID:  docker://35bbaf57b2867d3eaf889e26eb15e45908a6e20e1b659b5c775835faf3231928
    Image:         ibmcom/istio-proxy_init:1.0.0

    Medium:  Memory
  istio-certs:
    Type:        Secret (a volume populated by a Secret)
    SecretName:  istio.default
    Optional:    true
  default-token-7jt57:
    Type:        Secret (a volume populated by a Secret)
    SecretName:  default-token-7jt57
    Optional:    false
QoS Class:       BestEffort
Node-Selectors:  <none>
Tolerations:     <none>
Events:
  Type    Reason                 Age   From                 Message
  ----    ------                 ----  ----                 -------
  Normal  Scheduled              32m   default-scheduler    Successfully assigned productpage-v1-576c55ddb8-qnz26 to 10.10.1.30
  Normal  SuccessfulMountVolume  32m   kubelet, 10.10.1.30  MountVolume.SetUp succeeded for volume "istio-envoy"
  Normal  SuccessfulMountVolume  32m   kubelet, 10.10.1.30  MountVolume.SetUp succeeded for volume "istio-certs"
  Normal  SuccessfulMountVolume  32m   kubelet, 10.10.1.30  MountVolume.SetUp succeeded for volume "default-token-7jt57"
  Normal  Pulling                32m   kubelet, 10.10.1.30  pulling image "ibmcom/istio-proxy_init:1.0.0"
  Normal  Pulled                 30m   kubelet, 10.10.1.30  Successfully pulled image "ibmcom/istio-proxy_init:1.0.0"
  Normal  Created                30m   kubelet, 10.10.1.30  Created container
  Normal  Started                30m   kubelet, 10.10.1.30  Started container
  Normal  Pulling                30m   kubelet, 10.10.1.30  pulling image "istio/examples-bookinfo-productpage-v1:1.8.0"
  Normal  Pulled                 4m    kubelet, 10.10.1.30  Successfully pulled image "istio/examples-bookinfo-productpage-v1:1.8.0"
  Normal  Created                4m    kubelet, 10.10.1.30  Created container
  Normal  Started                4m    kubelet, 10.10.1.30  Started container
  Normal  Pulled                 4m    kubelet, 10.10.1.30  Container image "ibmcom/istio-proxyv2:1.0.0" already present on machine
  Normal  Created                4m    kubelet, 10.10.1.30  Created container
  Normal  Started                4m    kubelet, 10.10.1.30  Started container
```

4. Check the services created and note the port number for the productpage service.

```
 kubectl get services
```

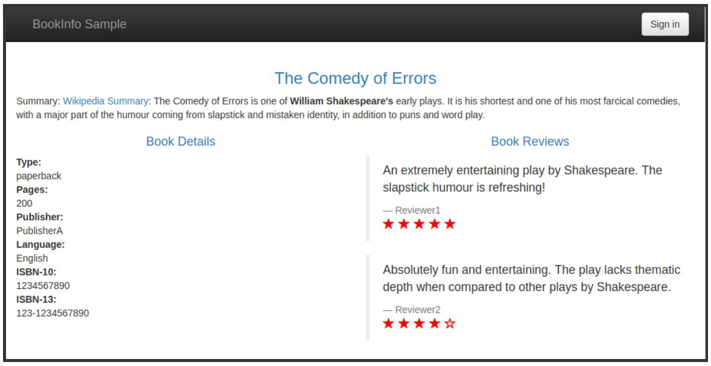


```
localuser@ibmcloudacademy:~$ kubectl get services
NAME          TYPE        CLUSTER-IP    EXTERNAL-IP   PORT(S)    AGE
details       ClusterIP   10.0.0.46     <none>        9080/TCP   38m
kubernetes    ClusterIP   10.0.0.1      <none>        443/TCP    11d
productpage   ClusterIP   10.0.0.159    <none>        9080/TCP   38m
ratings       ClusterIP   10.0.0.108    <none>        9080/TCP   38m
reviews       ClusterIP   10.0.0.113    <none>        9080/TCP   38m
```

5. To access the application without Istio, you can create a port-forwarding environment with `kubectl`

```
 kubectl port-forward $(kubectl get pod | grep productpage | awk
'{print $1}') 9000:9080
```

6. Make sure that you can access the forwarded port using a Web browser to `http://localhost:9000/productpage`. Try to refresh the page several times. You should be able to see that the reviews part has 3 different flavors (representing the different versions of the reviews application which all are wired to the same service). Refresh the pages several times.



The Comedy of Errors

Summary: Wikipedia Summary: The Comedy of Errors is one of **William Shakespeare's** early plays. It is his shortest and one of his most farcical comedies, with a major part of the humour coming from slapstick and mistaken identity, in addition to puns and word play.

Book Details

Type:
paperback
Pages:
200
Publisher:
PublisherA
Language:
English
ISBN-10:
1234567890
ISBN-13:
123-1234567890

Book Reviews

An extremely entertaining play by Shakespeare. The slapstick humour is refreshing!

— Reviewer1
★★★★★

Absolutely fun and entertaining. The play lacks thematic depth when compared to other plays by Shakespeare.

— Reviewer2
★★★★☆

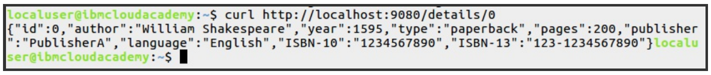
7. Stop the port-forward command using `Ctrl-C`.
8. Try accessing the details page using curl and port-forward.
   - Run the port-forwarding for the details pod:

     ```
     kubectl port-forward <details-pod> 9080:9080
     ```

   - Run the `curl` command to get the details:

     ```
     curl http://localhost:9080/details/0
     ```

   - Stop the port-forwarding using `Ctrl-C`



```
localuser@ibmcloudacademy:~$ curl http://localhost:9080/details/0
{"id":0,"author":"William Shakespeare","year":1595,"type":"paperback","pages":200,"publisher
":"PublisherA","language":"English","ISBN-10":"1234567890","ISBN-13":"123-1234567890"}localu
ser@ibmcloudacademy:~$ 
```

So far the bookinfo application is running without any management function from Istio. You did install the Istio sidecar with the pods. However, as there are no management directives yet, Istio is not doing anything.

Working with mixer definitions

Start working with the mixer definitions.

1. Mixer definitions are defined against the `istio-system` namespace. The out-of-the-box installation of Istio already has the necessary metrics collected. This first part of this exercise checks this collection. You will try to answer these questions in the following steps:
   - What feeds into the grafana dashboard?
   - What Istio record types are needed to create that feed?

____

- How can you check that those feeds exist?

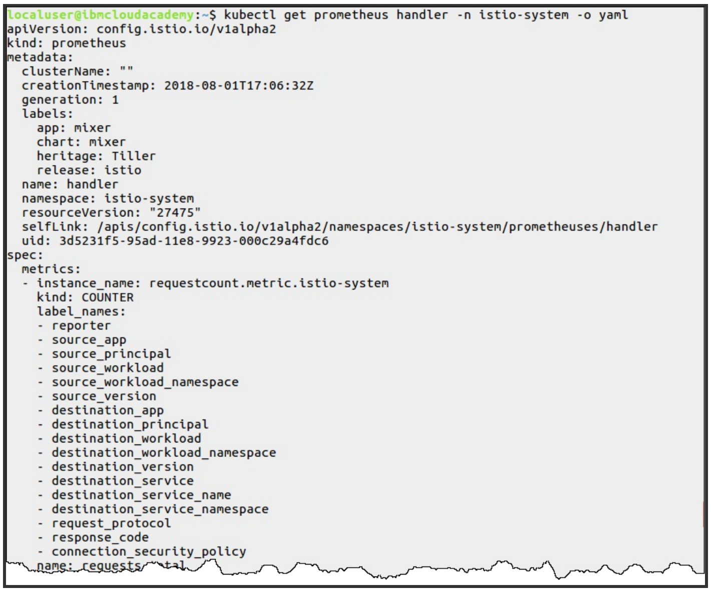
  ____

2. Perform the following command.

```
kubectl get prometheus -n istio-system
```

Use the `NAME` result from the command above.

```
kubectl get prometheus <NAME> -n istio-system -o yaml
```



```
localuser@ibmcloudacademy:~$ kubectl get prometheus handler -n istio-system -o yaml
apiVersion: config.istio.io/v1alpha2
kind: prometheus
metadata:
  clusterName: ""
  creationTimestamp: 2018-08-01T17:06:32Z
  generation: 1
  labels:
    app: mixer
    chart: mixer
    heritage: Tiller
    release: istio
  name: handler
  namespace: istio-system
  resourceVersion: "27475"
  selfLink: /apis/config.istio.io/v1alpha2/namespaces/istio-system/prometheuses/handler
  uid: 3d5231f5-95ad-11e8-9923-000c29a4fdc6
spec:
  metrics:
  - instance_name: requestcount.metric.istio-system
    kind: COUNTER
    label_names:
    - reporter
    - source_app
    - source_principal
    - source_workload
    - source_workload_namespace
    - source_version
    - destination_app
    - destination_principal
    - destination_workload
    - destination_workload_namespace
    - destination_version
    - destination_service
    - destination_service_name
    - destination_service_namespace
    - request_protocol
    - response_code
    - connection_security_policy
    name: requests_total
```

These metrics and collections are the ones to be loaded to prometheus, but what generates the metrics?

3. Perform the following command:

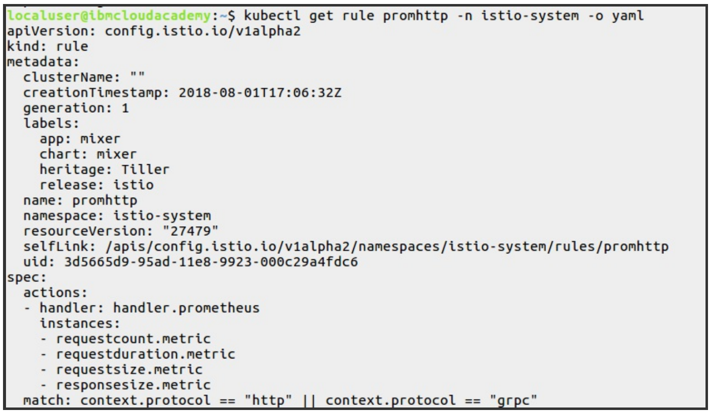
```
kubectl get rule -n istio-system
```

- How many rules do you get? ________________
- Which rules do you think are targeting the prometheus handler? ________________

4. Check the `promhttp` rule using the following command:

```
kubectl get rule promhttp -n istio-system -o yaml
```

What metrics are collected from the prometheus handler?

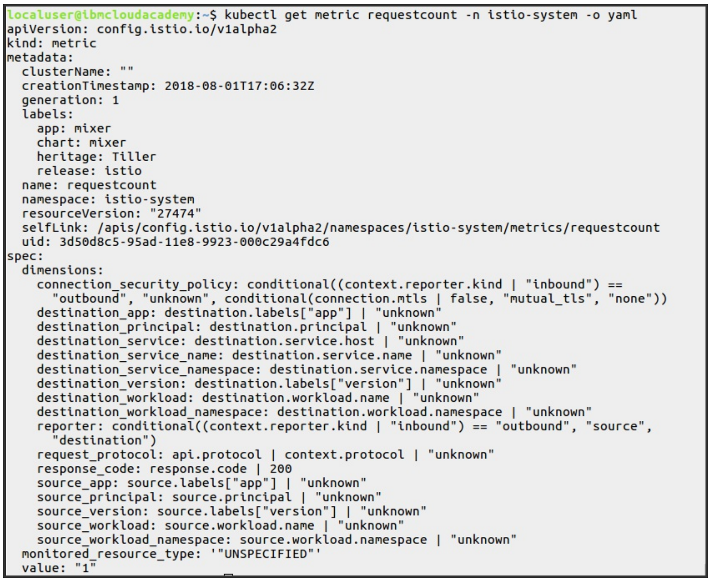


```
localuser@ibmcloudacademy:~$ kubectl get rule promhttp -n istio-system -o yaml
apiVersion: config.istio.io/v1alpha2
kind: rule
metadata:
  clusterName: ""
  creationTimestamp: 2018-08-01T17:06:32Z
  generation: 1
  labels:
    app: mixer
    chart: mixer
    heritage: Tiller
    release: istio
  name: promhttp
  namespace: istio-system
  resourceVersion: "27479"
  selfLink: /apis/config.istio.io/v1alpha2/namespaces/istio-system/rules/promhttp
  uid: 3d5665d9-95ad-11e8-9923-000c29a4fdc6
spec:
  actions:
  - handler: handler.prometheus
    instances:
    - requestcount.metric
    - requestduration.metric
    - requestsize.metric
    - responsesize.metric
  match: context.protocol == "http" || context.protocol == "grpc"
```

5. Lets look at the first instance, the `requestcount`.

```
kubectl get metric requestcount -n istio-system -o yaml
```

Do the collected metrics match with the loaded items?



```
localuser@ibmcloudacademy:~$ kubectl get metric requestcount -n istio-system -o yaml
apiVersion: config.istio.io/v1alpha2
kind: metric
metadata:
  clusterName: ""
  creationTimestamp: 2018-08-01T17:06:32Z
  generation: 1
  labels:
    app: mixer
    chart: mixer
    heritage: Tiller
    release: istio
  name: requestcount
  namespace: istio-system
  resourceVersion: "27474"
  selfLink: /apis/config.istio.io/v1alpha2/namespaces/istio-system/metrics/requestcount
  uid: 3d50d8c5-95ad-11e8-9923-000c29a4fdc6
spec:
  dimensions:
    connection_security_policy: conditional((context.reporter.kind | "inbound") ==
      "outbound", "unknown", conditional(connection.mtls | false, "mutual_tls", "none"))
    destination_app: destination.labels["app"] | "unknown"
    destination_principal: destination.principal | "unknown"
    destination_service: destination.service.host | "unknown"
    destination_service_name: destination.service.name | "unknown"
    destination_service_namespace: destination.service.namespace | "unknown"
    destination_version: destination.labels["version"] | "unknown"
    destination_workload: destination.workload.name | "unknown"
    destination_workload_namespace: destination.workload.namespace | "unknown"
    reporter: conditional((context.reporter.kind | "inbound") == "outbound", "source",
      "destination")
    request_protocol: api.protocol | context.protocol | "unknown"
    response_code: response.code | 200
    source_app: source.labels["app"] | "unknown"
    source_principal: source.principal | "unknown"
    source_version: source.labels["version"] | "unknown"
    source_workload: source.workload.name | "unknown"
    source_workload_namespace: source.workload.namespace | "unknown"
  monitored_resource_type: '"UNSPECIFIED"'
  value: "1"
```

6. Check the `grafana` dashboard. First you will do the port forwarding to allow access to `grafana`. Get the `grafana` pod name and port number:

```
kubectl get pod -n istio-system | grep grafana
kubectl get service -n istio-system | grep grafana
```

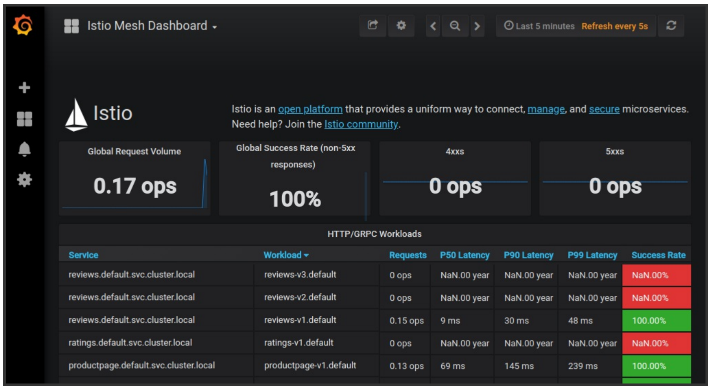
Use the pod name and port number to run port forwarding:

```
kubectl port-forward <podname> -n istio-system 10000:<portnumber>
```

7. Check the grafana dashboards at `http://localhost:10000`. Navigate through the various dashboards it has. You can change dashboards using the

   ⊞

   icon on the top left.

   Istio is an open platform that provides a uniform way to connect, manage, and secure microservices. Need help? Join the Istio community.

   Global Request Volume: 0.17 ops

   Global Success Rate (non-5xx responses): 100%

   4xxs: 0 ops

   5xxs: 0 ops

   HTTP/GRPC Workloads

Service	Workload	Requests	P50 Latency	P90 Latency	P99 Latency	Success Rate
reviews.default.svc.cluster.local	reviews-v3.default	0 ops	NaN.00 year	NaN.00 year	NaN.00 year	NaN.00%
reviews.default.svc.cluster.local	reviews-v2.default	0 ops	NaN.00 year	NaN.00 year	NaN.00 year	NaN.00%
reviews.default.svc.cluster.local	reviews-v1.default	0.15 ops	9 ms	30 ms	48 ms	100.00%
ratings.default.svc.cluster.local	ratings-v1.default	0 ops	NaN.00 year	NaN.00 year	NaN.00 year	NaN.00%
productpage.default.svc.cluster.local	productpage-v1.default	0.13 ops	69 ms	145 ms	239 ms	100.00%

   Stop the port forwarding when you are done.

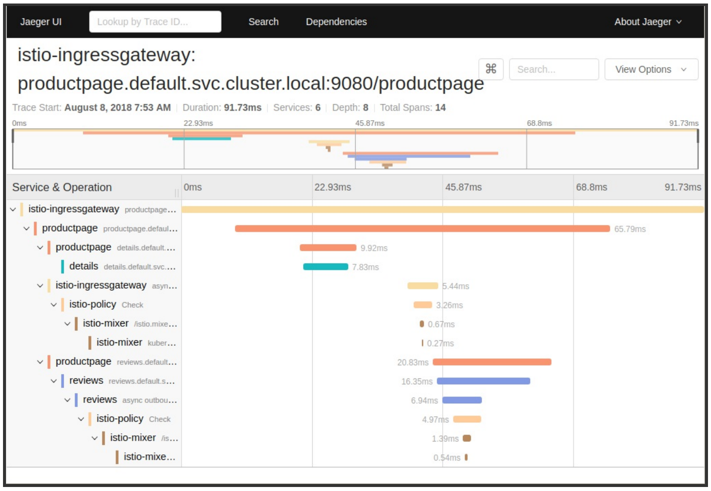
8. Look at the tracing data. Similar to the `grafana` procedure, first get the pod name and port number.

```
kubectl get pod -n istio-system | grep tracing
kubectl get service -n istio-system | grep tracing
```

Use the pod name and port number to run port forwarding:

```
kubectl port-forward <podname> -n istio-system 10000:<portnumber>
```

9. Open the jaeger dashboard at `http://localhost:10000`. Select the service `productpage` and then click **Find Traces** at the bottom of the left pane. It will show a bubble chart for the found instances. Click on one of the instances. You can see the time progression of your microservice application.



istio-ingressgateway: productpage.default.svc.cluster.local:9080/productpage

Trace Start: **August 8, 2018 7:53 AM** | Duration: **91.73ms** | Services: **6** | Depth: **8** | Total Spans: **14**

Jaeger UI
Lookup by Trace ID...
Search
Dependencies
About Jaeger
⌘
Search...
View Options
0ms
22.93ms
45.87ms
68.8ms
91.73ms
Service & Operation
istio-ingressgateway productpage...
productpage productpage.defaul... 65.79ms
productpage details.default.... 9.92ms
details details.default.svc.... 7.83ms
istio-ingressgateway asyn... 5.44ms
istio-policy Check 3.26ms
istio-mixer /istio.mixe... 0.67ms
istio-mixer kuber... 0.27ms
productpage reviews.default... 20.83ms
reviews reviews.default.s... 16.35ms
reviews async outbou... 6.94ms
istio-policy Check 4.97ms
istio-mixer /is... 1.39ms
istio-mixe... 0.54ms

Navigate and expand some of the components to understand the use of this application. Also note that the Istio overhead shown for the `istio-policy` and `istio-mixer` is quite small.

Stop the port forwarding when you are done.

Configuring Istio networking

The `bookinfo` application is currently running using the internal service in Kubernetes. There is no real direct access to the application. This exercise creates the necessary Istio resources for `bookinfo`.

Note: You can download most of the yaml files for the rest of the exercises with the command `git clone https://github.com/ibm-cloud-academy/istio-yaml`

After downloading the yaml files, change to the `istio-yaml` subdirectory using the `cd istio-yaml` command.

1. Edit the definition file called `bookinfo-gw.yaml`. This definition contains a Gateway object that is using the default `istio-ingressgateway`, using port `80`.

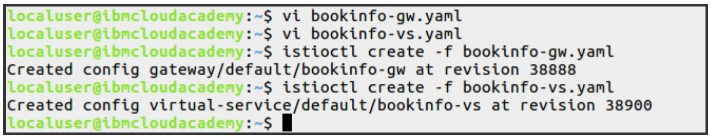
```
apiVersion: networking.istio.io/v1alpha3
kind: Gateway
metadata:
  name: bookinfo-gw
spec:
  selector:
    Istio: ingressgateway
  servers:
  - port:
      number: 80
      name: http
      protocol: HTTP
    hosts:
    - "*"
```

2. Edit the definition file called `bookinfo-vs.yaml`. This definition contains a virtual service object that is a member of `bookinfo-gw`. The application accepta the paths `/productpage`, `/login` and `/logoff`. The virtual service refers to the `productpage` host on port `9080`.

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: bookinfo
spec:
  hosts:
  - "*"
  gateways:
  - bookinfo-gw
  http:
  - match:
    - uri:
        exact: /productpage
    - uri:
        exact: /login
    - uri:
        exact: /logout
    route:
    - destination:
        host: productpage
        port:
          number: 9080
```

3. Activate both definitions:

```
istioctl create -f bookinfo-gw.yaml
istioctl create -f bookinfo-vs.yaml
```



```
localuser@ibmcloudacademy:~$ vi bookinfo-gw.yaml
localuser@ibmcloudacademy:~$ vi bookinfo-vs.yaml
localuser@ibmcloudacademy:~$ istioctl create -f bookinfo-gw.yaml
Created config gateway/default/bookinfo-gw at revision 38888
localuser@ibmcloudacademy:~$ istioctl create -f bookinfo-vs.yaml
Created config virtual-service/default/bookinfo-vs at revision 38900
localuser@ibmcloudacademy:~$ 
```

Note: Once the definition has been created, if you need to modify the yaml file, you can run `istioctl replace -f` command.

4. Try to access the bookinfo application. Remember the port mapping for http for the `istio-ingressgateway` service that you captured in a previous exercise.

   If you are working in IBM Cloud Private, enter the following in your browser.

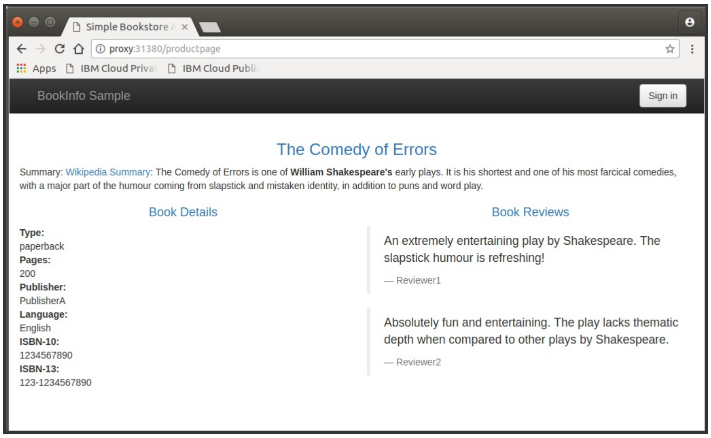
   `http://proxy:<ingressport>/productpage`

   If you are working in IBM Cloud Public, enter the following in your browser.

   `http://<Worker Node Public IP>:<ingressport>/productpage`

   The worker node Public IP address can be found with the command `ibmcloud ks workers <cluster name>`.

   You should get the same page as the one from the port-forward command.



The Comedy of Errors

Summary: Wikipedia Summary: The Comedy of Errors is one of **William Shakespeare's** early plays. It is his shortest and one of his most farcical comedies, with a major part of the humour coming from slapstick and mistaken identity, in addition to puns and word play.

Book Details

Type:
paperback
Pages:
200
Publisher:
PublisherA
Language:
English
ISBN-10:
1234567890
ISBN-13:
123-1234567890

Book Reviews

An extremely entertaining play by Shakespeare. The slapstick humour is refreshing!

— Reviewer1

Absolutely fun and entertaining. The play lacks thematic depth when compared to other plays by Shakespeare.

— Reviewer2

- Try to **Sign in** using user `foo` with the password of `bar`
- Try signing out

  Since you have not activated any security or other networking definitions using Istio, the application behaves exactly as it would in its default implementation. The next steps introduce different actions where you can modify the behavior of the application without changing the application itself.

6. Create definitions (`Virtual Service` and `Destination Rule`) for the `Reviews` page. The first few scenarios use the `reviews-dr.yaml` file.

   ```
   apiVersion: networking.istio.io/v1alpha3
   kind: DestinationRule
   metadata:
     name: reviews-dr
   spec:
     host: reviews
     subsets:
     - name: v1
       labels:
         version: v1
     - name: v2
       labels:
         version: v2
     - name: v3
       labels:
         version: v3
   ```

   Configure the file below such that only `reviews-v1` shows in the `productpage` every time:

   `reviews-vs.yaml`

   ```
   apiVersion: networking.istio.io/v1alpha3
   kind: VirtualService
   metadata:
     name: reviews-vs
   spec:
     hosts:
     - reviews
     http:
     - route:
       - destination:
           host: reviews
           subset: v1
   ```

   Activate the definition using:

   ```
   `istioctl create -f reviews-vs.yaml` <br>
   `istioctl create -f reviews-dr.yaml`
   ```

Check the product page now in your browser. Refresh the page. Confirm that the reviews never show the star rating information, even when you refresh it multiple times.

7. Modify the definitions to only show `reviews-v2`. This modification is performed in the `reviews-vs.yaml` file by changing the `subset` parameter to `subset: v2`. Apply the new definition:

   `istioctl replace -f reviews-vs.yaml`

   Refresh your `productpage` browser page several times. Confirm that now it just shows the ratings with black stars.

8. Modify the definition to perform traffic splitting so that it will show 50% of the requests `reviews-v2` and the other 50% `reviews-v3`. Modify the `reviews-vs.yaml` file to include the following route destinations.

   ```
   http:
   - route:
     - destination:
         host: reviews
         subset: v2
       weight: 50
     - destination:
         host: reviews
         subset: v3
       weight: 50
   ```

   Apply the new definition:

   `istioctl replace -f reviews-vs.yaml`

   Refresh your `productpage` browser page several times. Confirm that now it shows the ratings alternating between black stars and red stars. The color of the stars might not switch on each request. The configuration just causes 50% of the requests in the aggregate to choose each of the two versions.

9. Perform traffic steering such that only user `foo` will be sent to `reviews-v3` and all other users to `reviews-v2`. If the user is not logged in, it will go to `reviews-v1`. Change the `reviews-vs.yaml` file to include the following route definition.

```
http:
- match:
  - headers:
      end-user:
        exact: foo
  route:
  - destination:
      host: reviews
      subset: v3
- match:
  - headers:
      end-user:
        regex: ".*\\S+.*"
  route:
  - destination:
      host: reviews
      subset: v2
- route:
  - destination:
      host: reviews
      subset: v1
```

Apply the new definition:

`istioctl replace -f reviews-vs.yaml`

Test the definition using various user names in the `productpage` web page. There is no restriction on the users or passwords that you can use on the page.

10. Remove all the conditional routing and introduce latency for the user `foo` for 2 seconds. The `reviews-vs.yaml` file rules become as follows. Make these changes, apply the new definition, and test the result on `productpage`.

```
http:
- match:
  - headers:
      end-user:
        exact: foo
  fault:
    delay:
      percent: 100
      fixedDelay: 2s
  route:
  - destination:
      host: reviews
      subset: v2
- route:
  - destination:
      host: reviews
      subset: v3
```

11. Change from the delay into returning error 501 for `foo` to get reviews:

```
http:
- match:
  - headers:
      end-user:
        exact: foo
  fault:
    abort:
      percent: 100
      httpStatus: 501
  route:
  - destination:
      host: reviews
      subset: v2
- route:
  - destination:
      host: reviews
      subset: v3
```

12. Remove all of the fault injections in the `reviews-vs.yaml` file. Restore traffic steering for user `foo` to go to the `v2` subset:

```
http:
- match:
  - headers:
      end-user:
        exact: foo
  route:
  - destination:
      host: reviews
      subset: v2
- route:
  - destination:
      host: reviews
      subset: v3
```

Test to make sure that all fault injection has been removed and the steering works as expected.

13. Add a `ratings` virtual service and destination rule with a 2 second latency for `foo`.

    `ratings-vs.yaml`

    ```
    apiVersion: networking.istio.io/v1alpha3
    kind: VirtualService
    metadata:
      name: ratings-vs
    spec:
      hosts:
      - ratings
      http:
      - match:
        - headers:
            end-user:
              exact: foo
        fault:
          delay:
            percent: 100
            fixedDelay: 2s
        route:
        - destination:
            host: ratings
            subset: v1
      - route:
        - destination:
            host: ratings
            subset: v1
    ```

    `ratings-dr.yaml`

    ```
      apiVersion: networking.istio.io/v1alpha3
      kind: DestinationRule
      metadata:
        name: ratings-dr
      spec:
        host: ratings
        subsets:
        - name: v1
          labels:
            version: v1
    ```

    Activate the definition using:

`istioctl create -f ratings-vs.yaml`

`istioctl create -f ratings-dr.yaml`

Test the `productpage` and verify that the delays only happen for the user foo.

14. In the `reviews-dr.yaml` file, define a circuit breaker that is only to have a single connection in the `connectionPool`. Set the `outlierDetection` parameter to test every 5 seconds for any error. For any error, it shoudl eject the node for 5 minutes.

```
    trafficPolicy:
      connectionPool:
        http:
          http1MaxPendingRequests: 1
          maxRequestsPerConnection: 1
        tcp:
          maxConnections: 1
      outlierDetection:
        baseEjectionTime: 5m
        consecutiveErrors: 1
        interval: 5s
        maxEjectionPercent: 100
```

    - With a single connection, when you submit multiple requests in succession (use a couple of browser tabs); the first few requests should return after a couple of seconds. Afterwards, the request should just be failed quickly. This indicates that the circuit breaker in the `reviews` process has been tripped.
    - After the ejection time has passed (5 minutes), try to load the product page again to check whether it returns to the slow response time again.
    - Remove the traffic policy `from reviews-dr.yaml` and remove the fault from `ratings-vs.yaml`. Reapply the files using `istioctl replace -f <file>`.

15. Finish creating the `VirtualServices` and `DestinationRules` for the `details` and `productpage` components as follows. You will need these definitions for the RBAC testing later.

`productpage-vs.yaml`

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: productpage-vs
spec:
  hosts:
  - productpage
  http:
  - route:
    - destination:
        host: productpage
        subset: v1
```

`productpage-dr.yaml`

```
apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: productpage-dr
spec:
  host: productpage
  subsets:
  - name: v1
    labels:
      version: v1
```

`details-vs.yaml`

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: details-vs
spec:
  hosts:
  - details
  http:
  - route:
    - destination:
        host: details
        subset: v1
```

`details-dr.yaml`

```
apiVersion: networking.istio.io/v1alpha3
```

```
kind: DestinationRule
metadata:
  name: details-dr
spec:
  host: details
  subsets:
  - name: v1
    labels:
      version: v1
```

Load each of these definition files using the `istioctl create -f` command, as in the previous examples.

Working with an external service

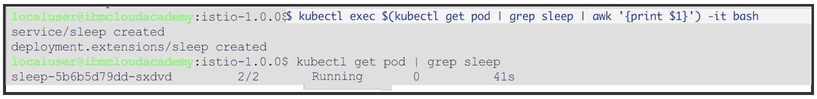
The external service that you will work with is represented by a simple `curl` command. This method simplifies the testing without the need to build a specialized application accessing an external API endpoint.

1. Deploy the `sleep` pod.

```
 cd istio-1.0.0
 kubectl apply -f <(istioctl kube-inject -f
samples/sleep/sleep.yaml)
```

Check that the pod got deployed correctly.

```
kubectl get pod | grep sleep
```



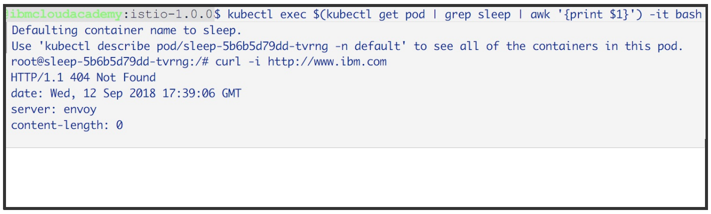
```
localuser@ibmcloudacademy:istio-1.0.0$ kubectl exec $(kubectl get pod | grep sleep | awk '{print $1}') -it bash
service/sleep created
deployment.extensions/sleep created
localuser@ibmcloudacademy:istio-1.0.0$ kubectl get pod | grep sleep
sleep-5b6b5d79dd-sxdvd         2/2       Running      0          41s
```

2. Open another terminal session to connect to the `sleep` pod. If you are on IBM Cloud Public and have opened a new terminal session, enter this command first `` `ibmcloud ks cluster-config <cluster name> | grep KUBECONFIG` `` (include the back-ticks).

```
kubectl exec $(kubectl get pod | grep sleep | awk '{print $1}') -it
bash
```

3. Check whether you can access any external web-site from the sleep pod. Let's use `http://www.ibm.com` as an example.

   ```
   curl -i http://www.ibm.com
   ```

   ```
   ibmcloudacademy:istio-1.0.0$ kubectl exec $(kubectl get pod | grep sleep | awk '{print $1}') -it bash
   Defaulting container name to sleep.
   Use 'kubectl describe pod/sleep-5b6b5d79dd-tvrng -n default' to see all of the containers in this pod.
   root@sleep-5b6b5d79dd-tvrng:/# curl -i http://www.ibm.com
   HTTP/1.1 404 Not Found
   date: Wed, 12 Sep 2018 17:39:06 GMT
   server: envoy
   content-length: 0
   ```

   You notice that it cannot find the URL, and `envoy` is the one that returns the 404 code.

4. Create a `serviceEntry` for `www.ibm.com` in the `ibmse.yaml` file.

   ```
   apiVersion: networking.istio.io/v1alpha3
   kind: ServiceEntry
   metadata:
     name: ibm-se
   spec:
     hosts:
     - www.ibm.com
     ports:
     - number: 80
       name: http
       protocol: http
     resolution: DNS
     location: MESH_EXTERNAL
   ```

   Apply the `serviceEntry` using the command `istioctl create -f istio-yaml/ibmse.yaml` from your first terminal window (not the one with the bash session into the `sleep` container).

   - Test your connection to `www.ibm.com` from the bash session into the `sleep` container.

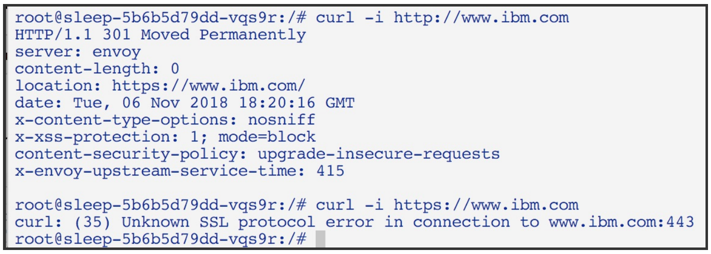
     ```
     curl -i http://www.ibm.com
     ```

     You should get some output and a HTTP response of 301, which redirects to the secure https site.

- Now try to connect to the secure https://www.ibm.com

  ```
  curl -i https://www.ibm.com
  ```

  You should get an error citing `Unknown SSL protocol error` indicating that it does not know how to handle https. That error is caused because the only port that is open is port 80, the http port, from the ServiceEntry definition.



```
root@sleep-5b6b5d79dd-vqs9r:/# curl -i http://www.ibm.com
HTTP/1.1 301 Moved Permanently
server: envoy
content-length: 0
location: https://www.ibm.com/
date: Tue, 06 Nov 2018 18:20:16 GMT
x-content-type-options: nosniff
x-xss-protection: 1; mode=block
content-security-policy: upgrade-insecure-requests
x-envoy-upstream-service-time: 415

root@sleep-5b6b5d79dd-vqs9r:/# curl -i https://www.ibm.com
curl: (35) Unknown SSL protocol error in connection to www.ibm.com:443
root@sleep-5b6b5d79dd-vqs9r:/#
```

5. To resolve this issue, you must create a TLS termination that accesses the HTTPS site.

   - If you are running in the IBM Kubernetes service in IBM Cloud Public, modify the `ibmse.yaml` to add port 443 to the port list as follows:

     ```
       ports:
       - number: 80
         name: http
         protocol: HTTP
       - number: 443
         name: https
         protocol: HTTPS
       resolution: DNS
       location: MESH_EXTERNAL
     ```

     Activate the modified definition using the command `istioctl replace -f istio-yaml/ibmse.yaml`

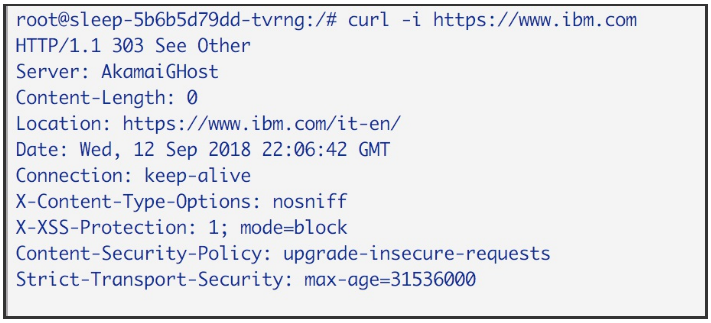
   - If you are running in an IBM Cloud Private environment, create a virtual service to route to `www.ibm.com` using `tls` in a file called `ibmvs.yaml`.

     ```
     apiVersion: networking.istio.io/v1alpha3
     kind: VirtualService
     metadata:
     ```

```
  name: ibm-vs
spec:
  hosts:
  - www.ibm.com
  tls:
  - match:
    - port: 443
      sni_hosts:
      - www.ibm.com
    route:
    - destination:
        host: www.ibm.com
        port:
          number: 443
      weight: 100
```

Activate the virtual service using the command `istioctl create -f istio-yaml/ibmvs.yaml`.

6. Test the connection with `curl -i https://www.ibm.com` from the bash session to the `sleep` container. Make sure that you can connect to the destination. In the testing that we performed, it redirects you (HTTP 303) to another page, but the communication works.



```
root@sleep-5b6b5d79dd-tvrng:/# curl -i https://www.ibm.com
HTTP/1.1 303 See Other
Server: AkamaiGHost
Content-Length: 0
Location: https://www.ibm.com/it-en/
Date: Wed, 12 Sep 2018 22:06:42 GMT
Connection: keep-alive
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Content-Security-Policy: upgrade-insecure-requests
Strict-Transport-Security: max-age=31536000
```

Defining RBAC security

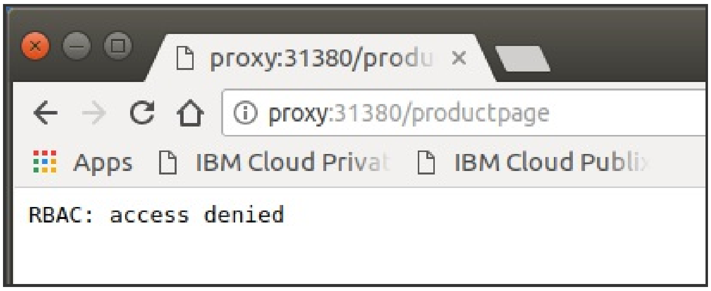
In this exercise, you will work with RBAC (Role Based Access Control) security for the `bookinfo` application.

1. Create the global RbacConfig definition to enable RBAC for the whole default namespace. Edit a file called `rbacconfig.yaml` and define the following:

```
apiVersion: rbac.istio.io/v1alpha1
kind: RbacConfig
metadata:
  name: default
spec:
  mode: 'ON_WITH_INCLUSION'
  inclusion:
    namespaces: ["default"]
```

This definition activates RBAC in the default namespace. Change your current directory to the `istio-yaml` with the command `cd istio-yaml`. Apply this definition with the command `istioctl create -f rbacconfig.yaml`.

2. In your browser, check now whether you can access the `productpage`. It will reject the request because you have not defined any role and access to the role.

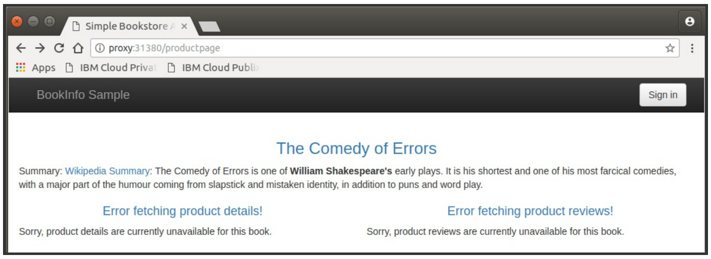


```
RBAC: access denied
```

3. Create an access rule for the `productpage` viewer (GET action). Create a file called `productpage-viewer.yaml` as follows:

```
apiVersion: rbac.istio.io/v1alpha1
kind: ServiceRole
metadata:
  name: productpage-viewer
  namespace: default
spec:
  rules:
  - services: ["productpage.default.svc.cluster.local"]
    methods: ["GET"]
---
apiVersion: rbac.istio.io/v1alpha1
kind: ServiceRoleBinding
metadata:
  name: bind-productpage-viewer
  namespace: default
spec:
  subjects:
  - user: "*"
  roleRef:
    kind: ServiceRole
    name: "productpage-viewer"
```

4. Activate the definition using the command `istioctl create -f productpage-viewer.yaml` and check whether you can access the product page. Do you see any error(s) on the page?
   Why? ____



The Comedy of Errors

Summary: Wikipedia Summary: The Comedy of Errors is one of **William Shakespeare's** early plays. It is his shortest and one of his most farcical comedies, with a major part of the humour coming from slapstick and mistaken identity, in addition to puns and word play.

Error fetching product details!

Sorry, product details are currently unavailable for this book.

Error fetching product reviews!

Sorry, product reviews are currently unavailable for this book.

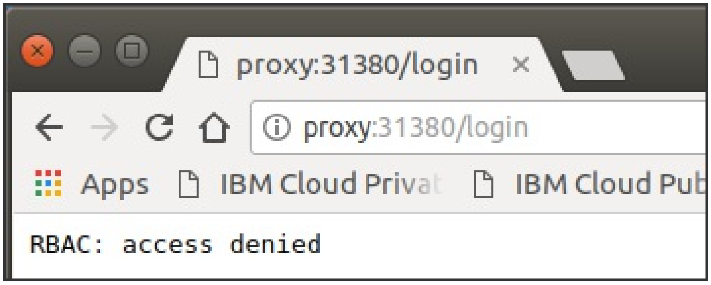
5. Fix the error by allowing all access to the `details` and `reviews` pages (you will restrict that later). Edit the files `details-viewer.yaml`, `ratings-viewer.yaml` and `reviews-viewer.yaml`. You will see that they are each similar to the `productpage-viewer.yaml` file, just with the object and service names changed to the appropriate values.

   Activate the definitions.

```
istioctl create -f details-viewer.yaml
istioctl create -f reviews-viewer.yaml
istioctl create -f ratings-viewer.yaml
```

   Verify that the product page can now be accessed without error.

6. Check whether you can **Sign in** to the product page. Specify a username and click **Sign in**.
   Why? ____



```
RBAC: access denied
```

7. Modify the `productpage-viewer.yaml` file to include the `POST` method for the `ServiceRole` record. Activate it using `istioctl replace -f productpage-viewer.yaml`.

```
apiVersion: rbac.istio.io/v1alpha1
kind: ServiceRole
```

```
metadata:
  name: productpage-viewer
  namespace: default
spec:
  rules:
  - services: ["productpage.default.svc.cluster.local"]
    methods: ["GET", "POST"]
```

Check that now you should be able to **Sign in**.

8. The current status is that all permissions are wide-open. For the details-viewer.yaml to only allow access from the productpage, answer the following questions:

    - What record do you update? `________________`
    - What property can you use for checking access? `____________________`
    - Can you do that now? `______________`

    Note: You should update the `ServiceRoleBinding` stanza. The `ServiceRole` defines the destination and the constraints that label that specific destination. The `Binding` defines which source can access the destination including specific identifying properties. The main property to identify a source is the `user` or `source.principal`. You cannot actually make this change yet because without mTLS, you do not really know which source generated the request except for its IP address.

9. Configure mutual TLS and a service account. First, enable a policy record that enables `mTLS` for the `default` namespace. Create or edit a yaml file called `default-policy.yaml`.

    ```
    apiVersion: authentication.istio.io/v1alpha1
    kind: Policy
    metadata:
      name: default
      namespace: default
    spec:
      peers:
      - mtls: {}
    ```

10. Load the policy using the command `istioctl create -f default-policy.yaml`.

    Can you access the product page? `_________`

The product page now requires `mTLS` access. Your request from the gateway has not specified any TLS attributes, and therefore it fails.

11. Activate `mTLS` in the destination rules. Add the following stanza in each of the destination rules files: `productpage-dr.yaml`, `reviews-dr.yaml`, `ratings-dr.yaml` and `details-dr.yaml`). Make sure the `trafficPolicy:` parameter is indented 2 spaces such that it aligns with the `host:` parameter.

    ```
      trafficPolicy:
        tls:
          mode: ISTIO_MUTUAL
    ```

    Activate the definition for each of the 4 files using the `istioctl replace -f <file>` command.

    Note: If you get an error on any of these yaml files saying the destination rule is not found, run the command `istioctl create -f <file>` for that file.

    Can you access the `productpage` now? ______________________________

12. Create a service account for the `bookinfo` components. Create or edit the `bookinfo-sa.yaml` file as follows:

    ```
    apiVersion: v1
    kind: ServiceAccount
    metadata:
      name: productpage-sa
    ---
    apiVersion: v1
    kind: ServiceAccount
    metadata:
      name: details-sa
    ---
    apiVersion: v1
    kind: ServiceAccount
    metadata:
      name: reviews-sa
    ```

    Load the definition using the command `kubectl create -f bookinfo-sa.yaml`. Do you think we missed one service account? `______`

Note: No, the `ratings-sa` is intentionally left out as the ratings service does not call any other service, hence it does not need to be identified using a service account.

13. Modify the serviceAccount for the bookinfo deployment units. Edit the definition for bookinfo deployments from `~/istio-1.0.0/samples/bookinfo/platform/kube/bookinfo.yaml`

14. Edit `bookinfo.yaml` and add the appropriate `serviceAccountName` stanzas, following the examples above, under `spec` and just before `containers` for each of the deployments for `bookinfo`. Use the related serviceAccounts for each of the components. Load the definition using the command:

    ```
    kubectl apply -f <(istioctl kube-inject -f bookinfo.yaml)
    ```

    Wait a while so that the new deployment can become active. Make sure that after the modification, the application is still running.

15. Now let's add restrictions on which applications can access the `bookinfo` components. Edit the `details-viewer.yaml` file and modify the `ServiceRoleBinding` to look like the following stanza:

    ```
      subjects:
      - user: "cluster.local/ns/default/sa/details-sa"
    ```

    Load the definition with the command `istioctl replace -f details-viewer.yaml`. Check the `productpage` web page. Is it working? Why? ______________________________

    Note: Which application accesses `details`? It is the `productpage`. If you modified the yaml file correctly, then the `productpage` is run using the `productpage-sa` user, not the `details-sa` user.

16. Fix any resulting errors. The `source.principal` parameter should be `cluster.local/ns/default/sa/productpage-sa`. Make this change and replace the `details-viewer.yaml` file. Check that you can load the `productpage` correctly now.

End of exercise