IBM开源技术微讲堂

Istio系列

第2讲

Istiod -回归单体

IBM 开放技术研究院

- 专注于开源技术和开放标准的开发
  - 公众号： ibmopentech
- 微讲堂系列活动
  - 每周四晚8点
  - WebEX和哔哩哔哩同步直播
  - 课程主页： https://developer.ibm.com/cn/os-academy-istio/

Istio 系列

5月28日	Istio overview
6月4日	Istiod - from microservices to monolith
6月11日	Istio Hands-on
6月18日	Use WSAM to extend your envoy proxy
6月25日	Prow and Istio

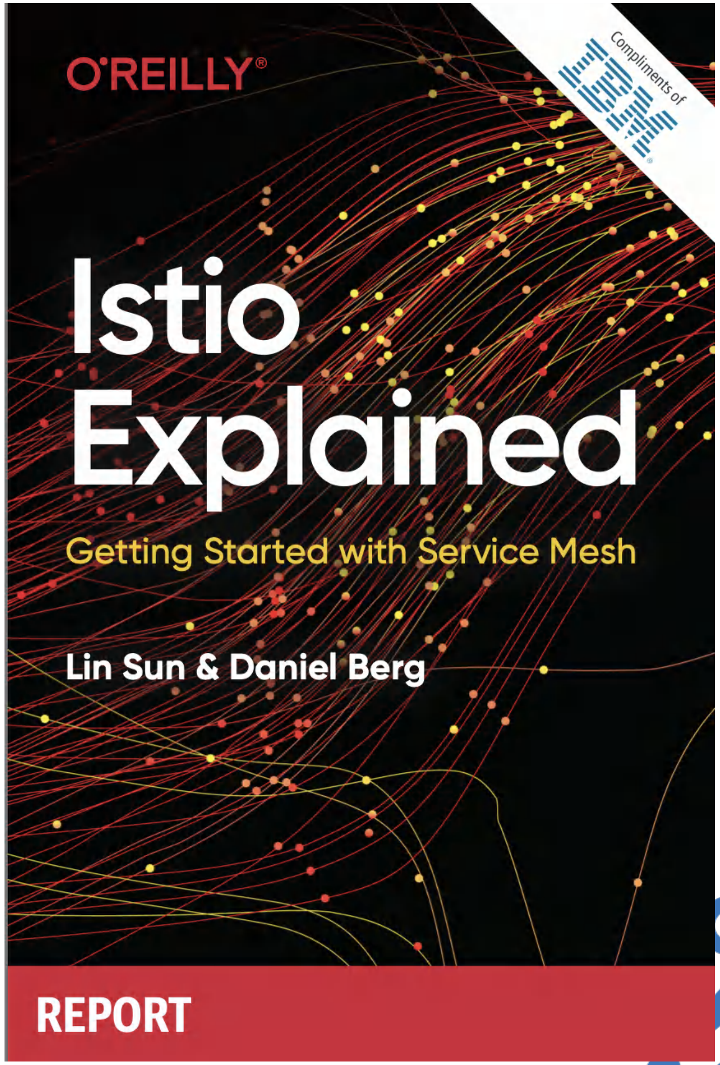
IBM

Lin Sun

IBM Senior Technical Staff Member

@linsun_unc

Members and co-founders of Istio open source project



O'REILLY®

Compliments of IBM

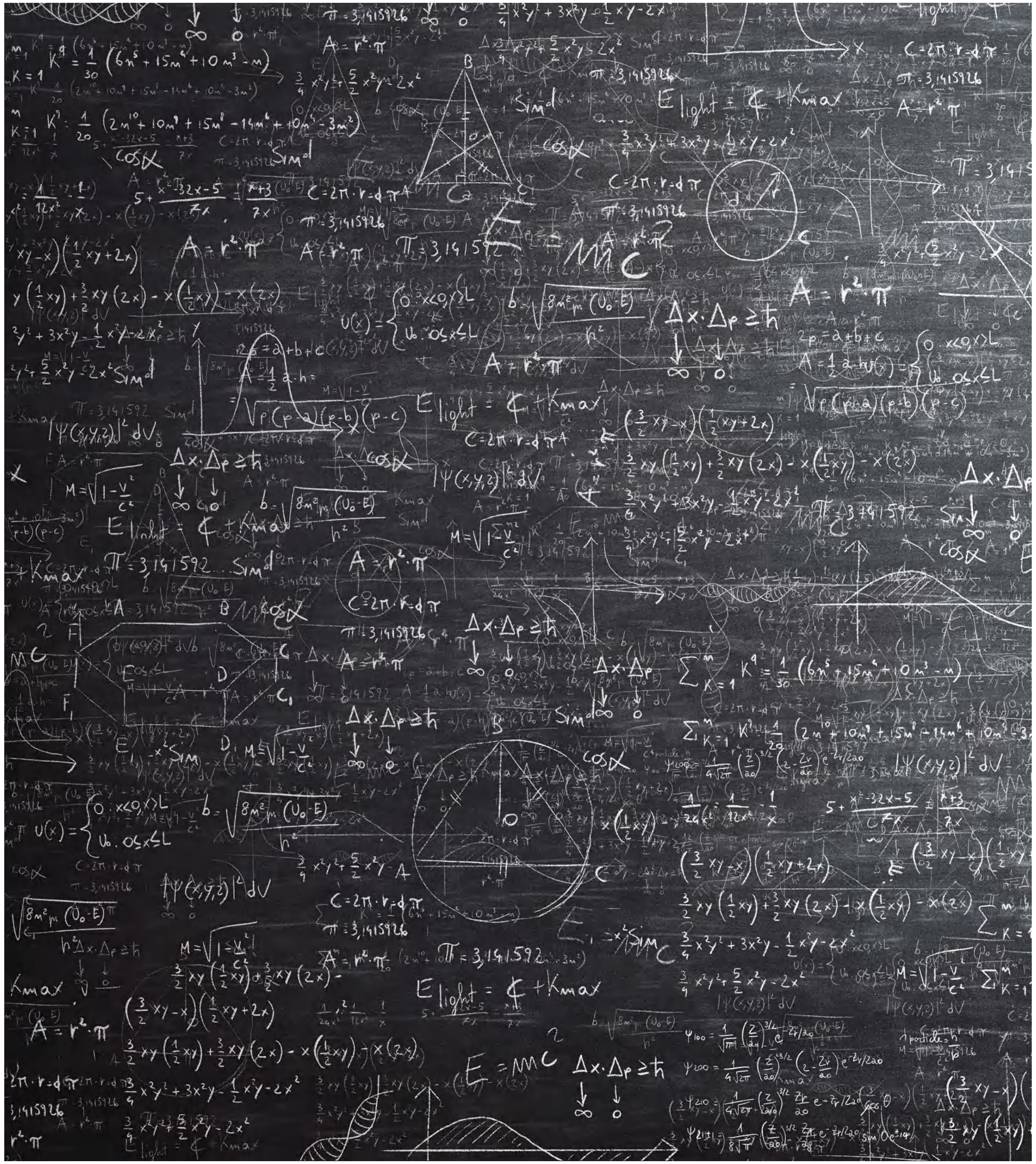
Istio Explained

Getting Started with Service Mesh

Lin Sun & Daniel Berg

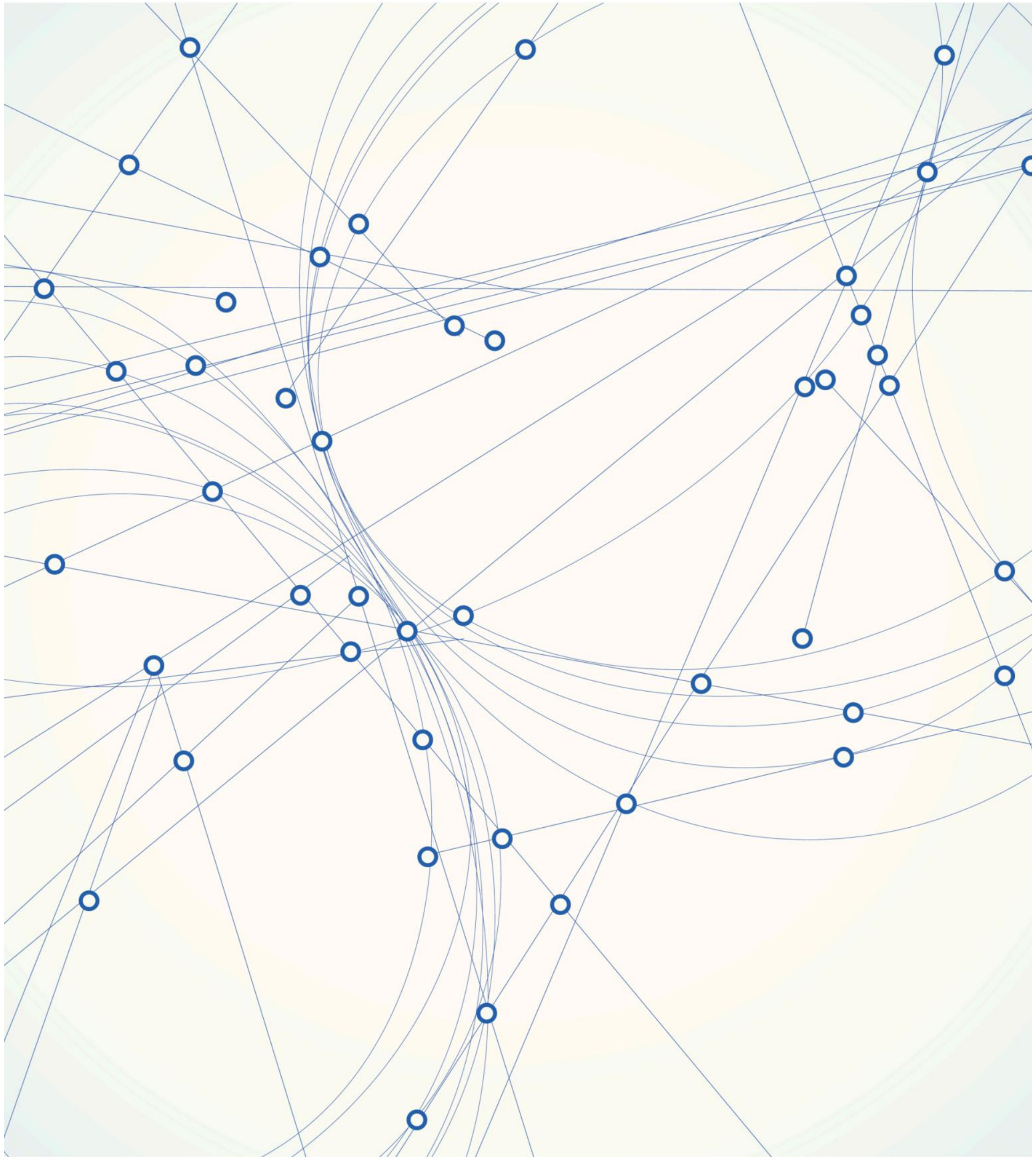
REPORT

Managing microservices doesn’t need to be complicated

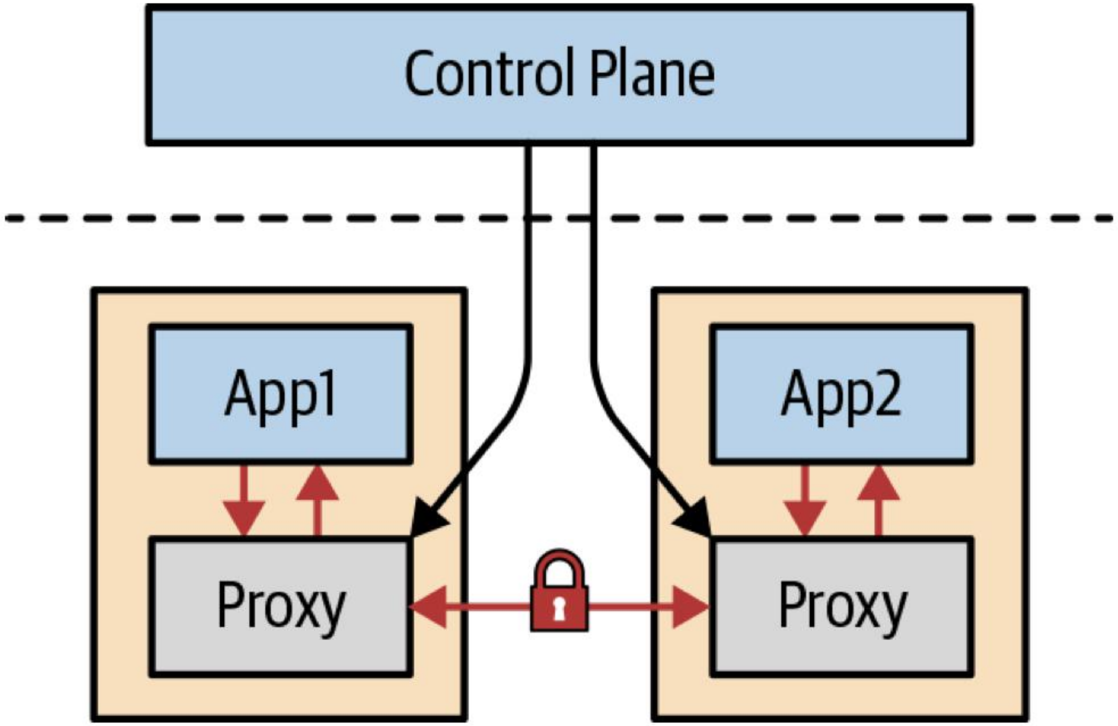


What is a service mesh?

- A service mesh is a **programmable** framework that allows you to **observe**, **secure**, and **connect** microservices.



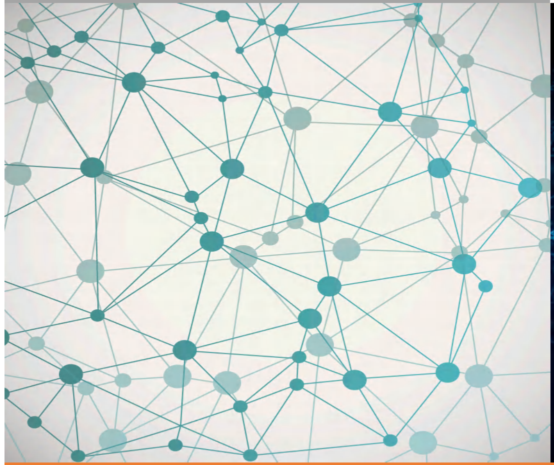
How does a service mesh work?



IBM

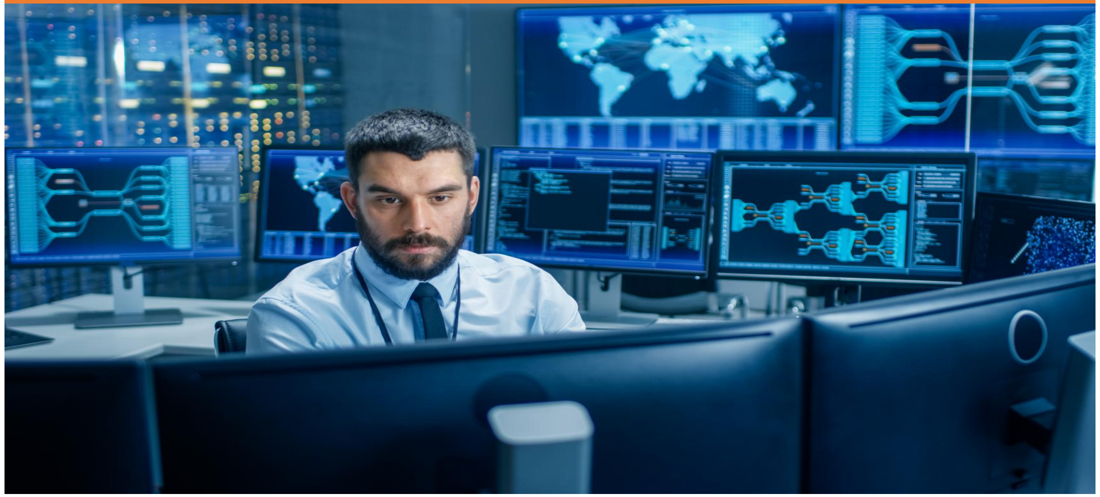
Istio

- Connect

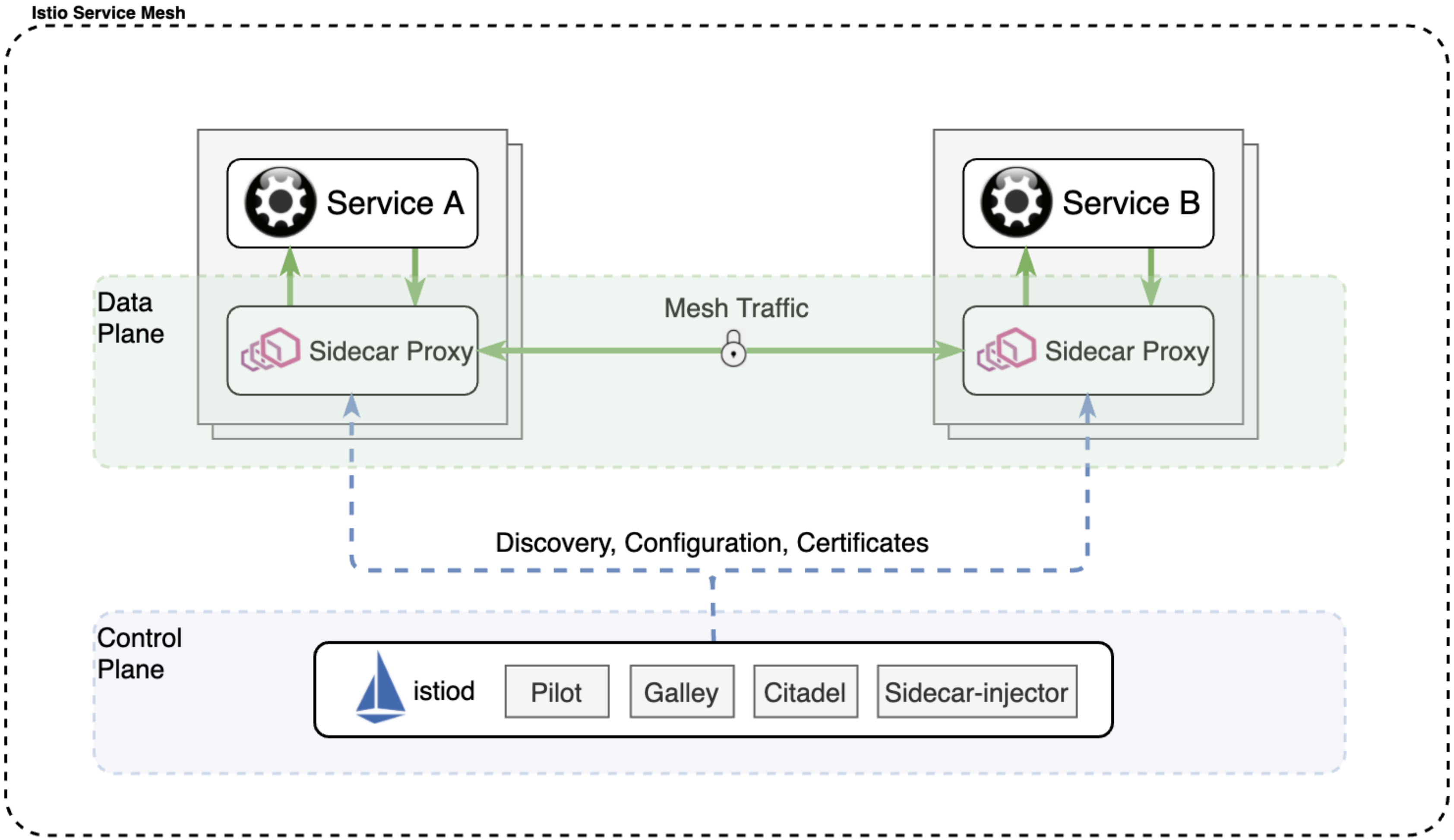


- Secure

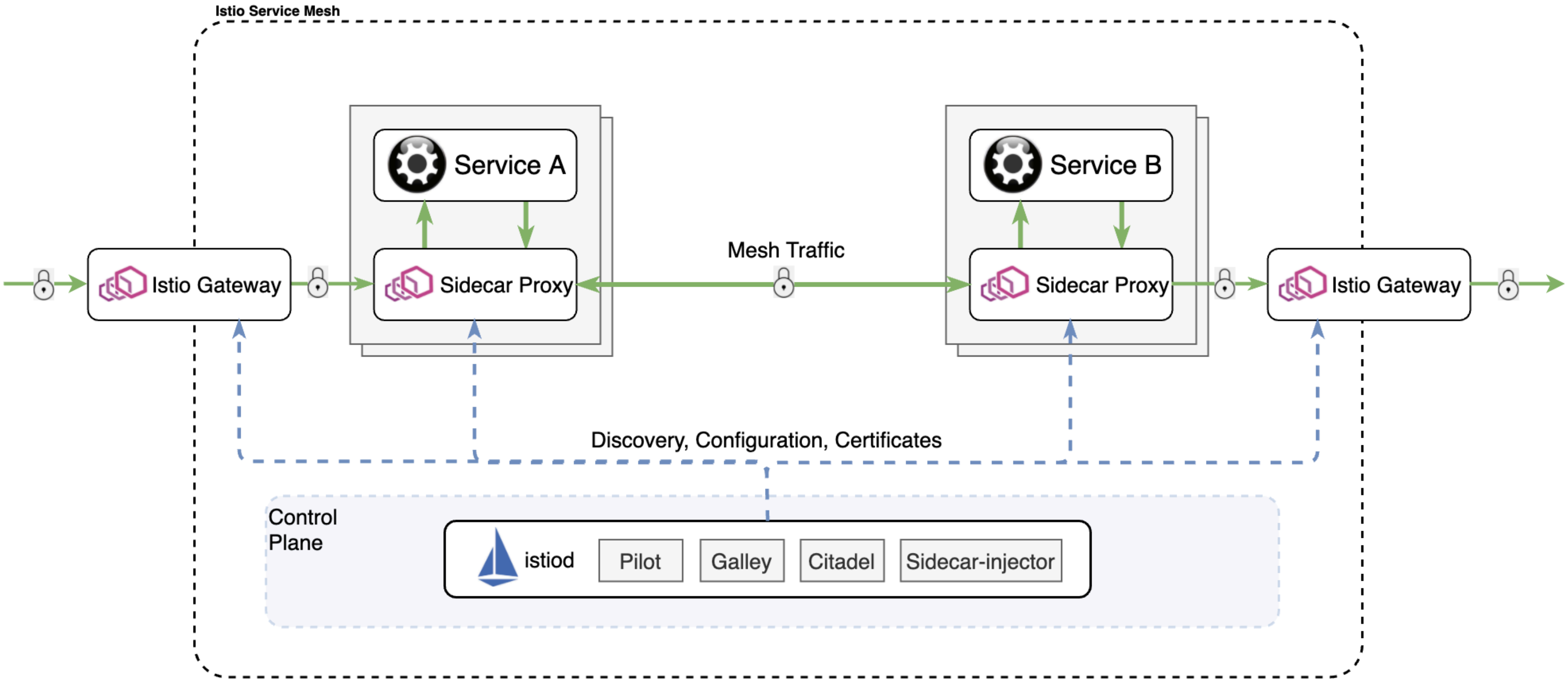
- Observe



Istio Architecture



Istio Architecture



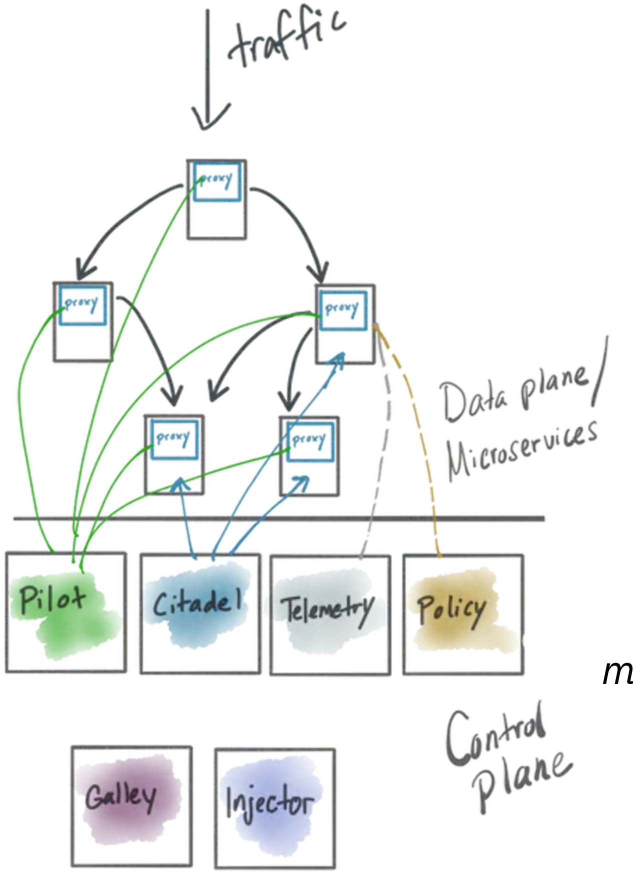
Istio 1.5 Highlight

- Simplified architecture and operations
  - istiod binary
  - Improvements in operator
- Improved extensibility
  - Mixer-less
  - Telemetry v2
- SDS (secret discovery service) by default
- New Authentication Policy

Simplified Architecture - istiod

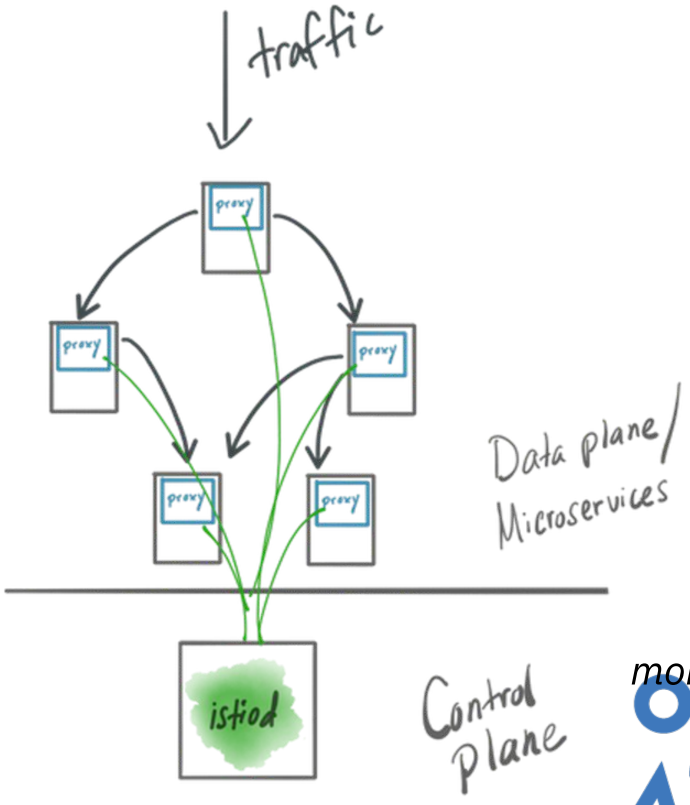
Images from Christian Posta

Istio 1.4 <



microservices

Istio 1.5 >



monolith

Why microservices prior to 1.5?

- Leverage Istio sidecar to secure control plane microservices
- Leverage Istio sidecar to observe control plane microservices.
- Ability to operate pilot differently than mixer or citadel
- Possible ability to develop and release Istio control plane components independently.
- Eat our own dog food ☺

Advantages of microservices

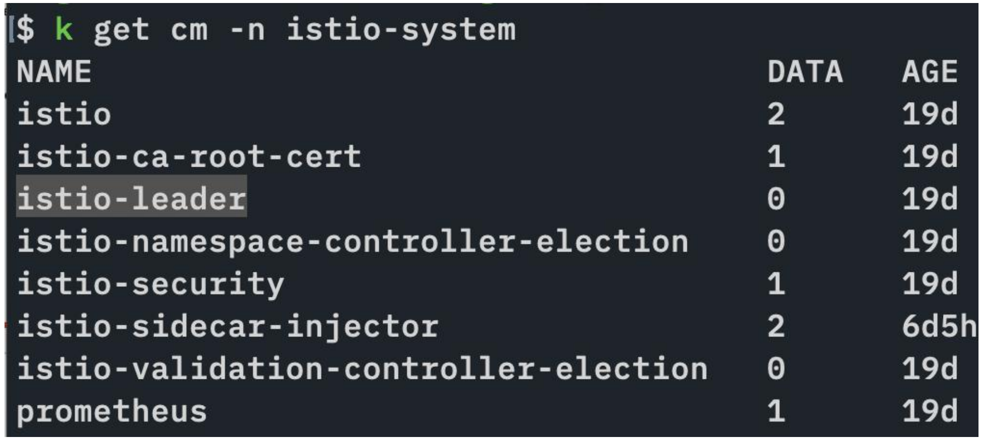
- Different programming languages for services
- Different team for managing services individually
- Different releases for services at different times
- Scale components independently
- Maintain security boundaries among your services

Why istiod?

- Simplify Istio install experience
- One single delivery, single deployment and service to install, manage and upgrade.
- Simplify Istio configuration experience
  - No longer need PodSecurityPolicy
- VMs and Multiclusters become easier
- Scalability becomes easier
- Debugging becomes easier
- Startup time improves
  - Component dependency is removed
- Performance improves
  - Communication among components is simpler and reliable

How Istiod works?

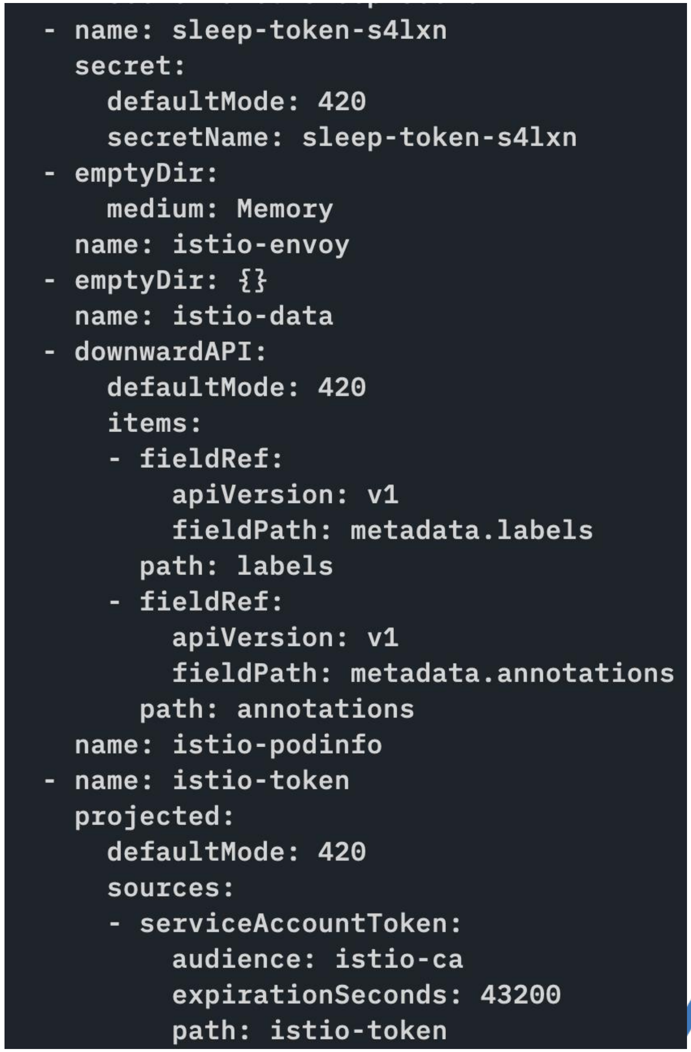
- certProvider has 2 choices:
    - istiod (default)
    - kubernetes
- istio configmap contains runtime mesh configuration for istiod
- Istio-leader defines which istiod is the leader
- Istio-sidecar-injector contains config for sidecar injectors
- Istio-ca-root-cert is created for each namespace
- prometheus contains Prometheus scrape configs and jobs



```
$ k get cm -n istio-system
NAME                                   DATA   AGE
istio                                  2      19d
istio-ca-root-cert                     1      19d
istio-leader                           0      19d
istio-namespace-controller-election    0      19d
istio-security                         1      19d
istio-sidecar-injector                 2      6d5h
istio-validation-controller-election   0      19d
prometheus                             1      19d
```

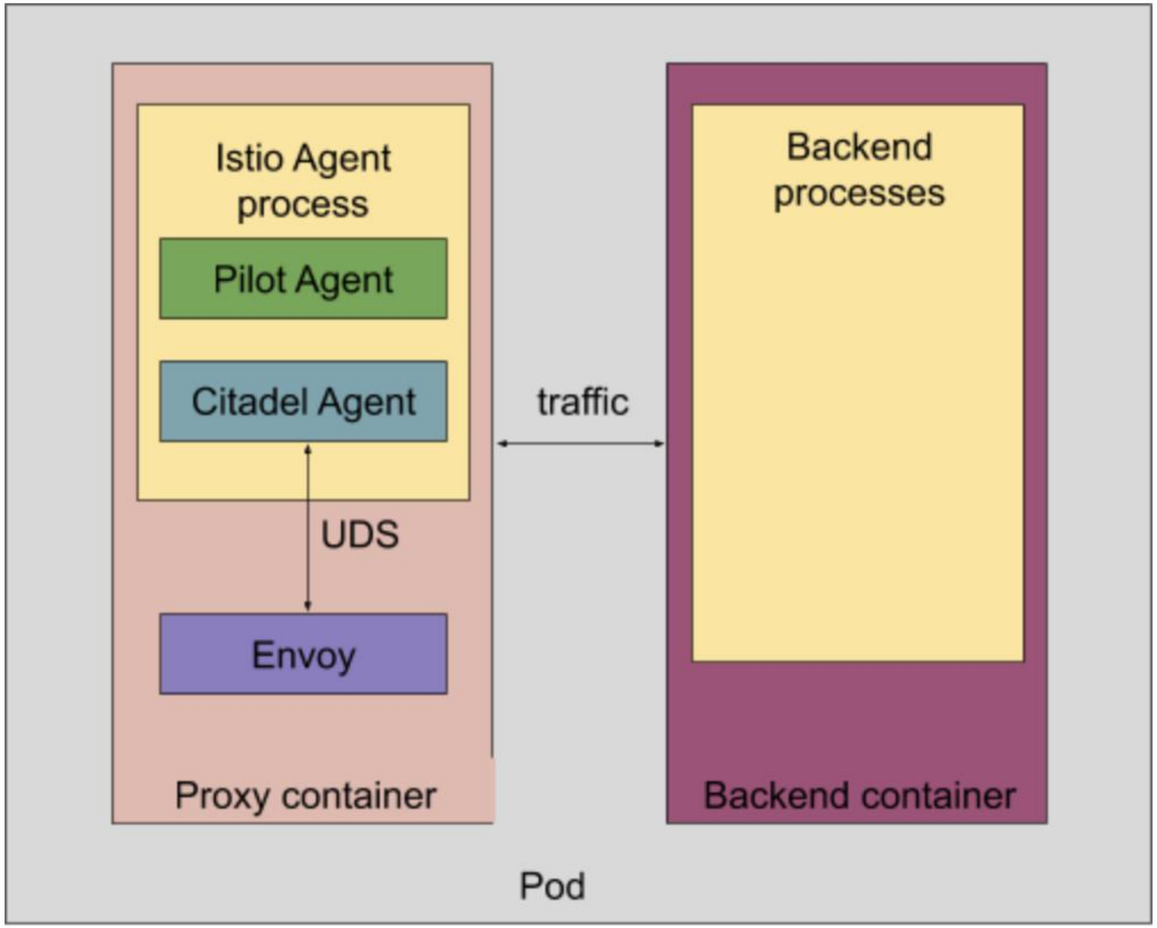
A service is added to the mesh

- Istio-ca-root-cert configmap must be created for each namespace by istiod
- Istio-ca-root-cert will be mounted to the pod
- The service's service account token & istio-token will be used to by istio-agent for generating the certificate signing request.
- Istiod does the authn and authz check on the CSR request and issue the signed certificate and key to istio-agent.



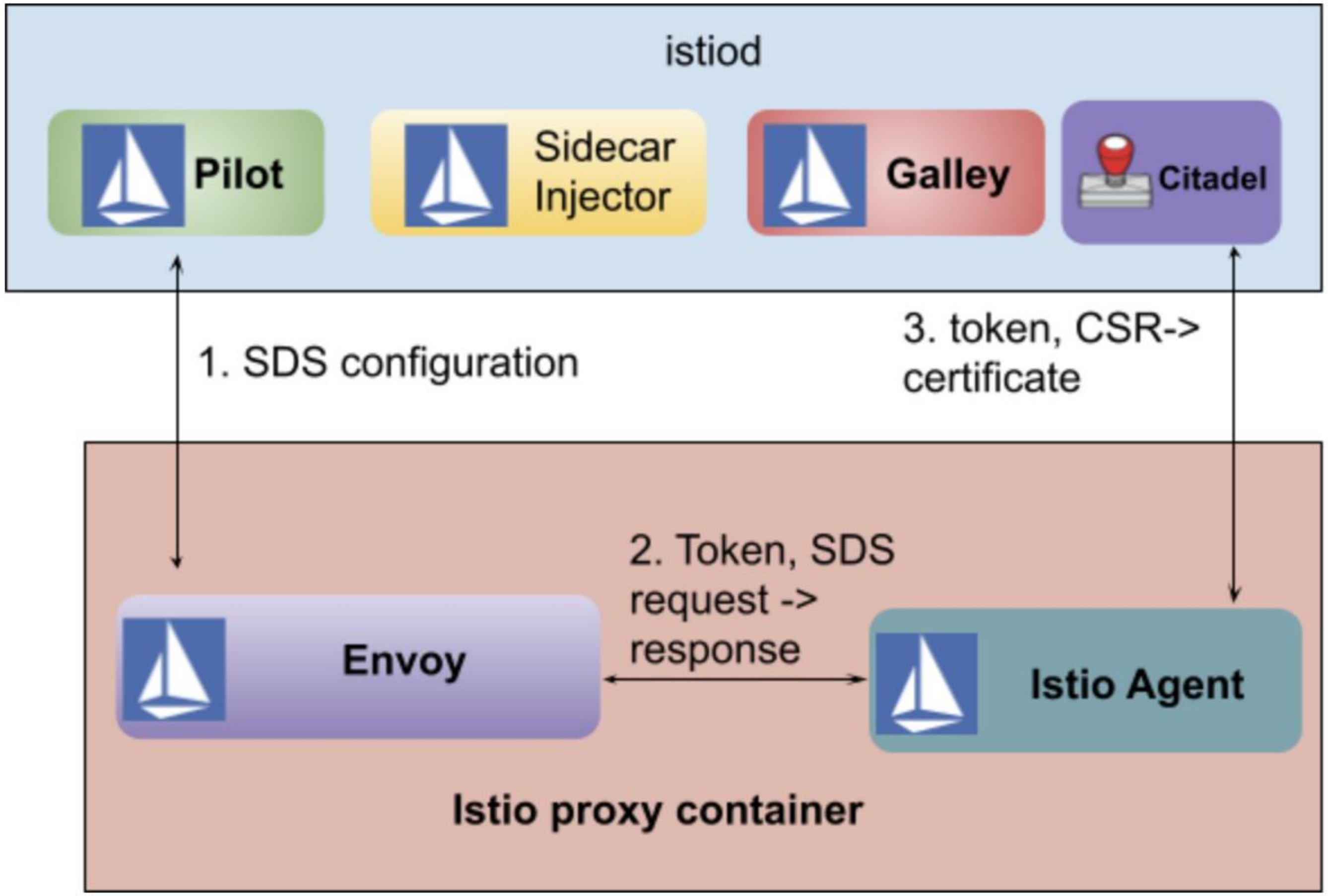
```
- name: sleep-token-s4lxn
  secret:
    defaultMode: 420
    secretName: sleep-token-s4lxn
- emptyDir:
    medium: Memory
  name: istio-envoy
- emptyDir: {}
  name: istio-data
- downwardAPI:
    defaultMode: 420
    items:
    - fieldRef:
        apiVersion: v1
        fieldPath: metadata.labels
      path: labels
    - fieldRef:
        apiVersion: v1
        fieldPath: metadata.annotations
      path: annotations
  name: istio-podinfo
- name: istio-token
  projected:
    defaultMode: 420
    sources:
    - serviceAccountToken:
        audience: istio-ca
        expirationSeconds: 43200
        path: istio-token
```

Istio-agent



https://docs.google.com/document/d/10I_BchyXP0ZEGkuCBM3_q4ny6Ji5ygH5nZdlUJ_PNss

Identity architecture for data plane



istiod

Pilot

Sidecar Injector

Galley

Citadel

1. SDS configuration

3. token, CSR-> certificate

2. Token, SDS request -> response

Envoy

Istio Agent

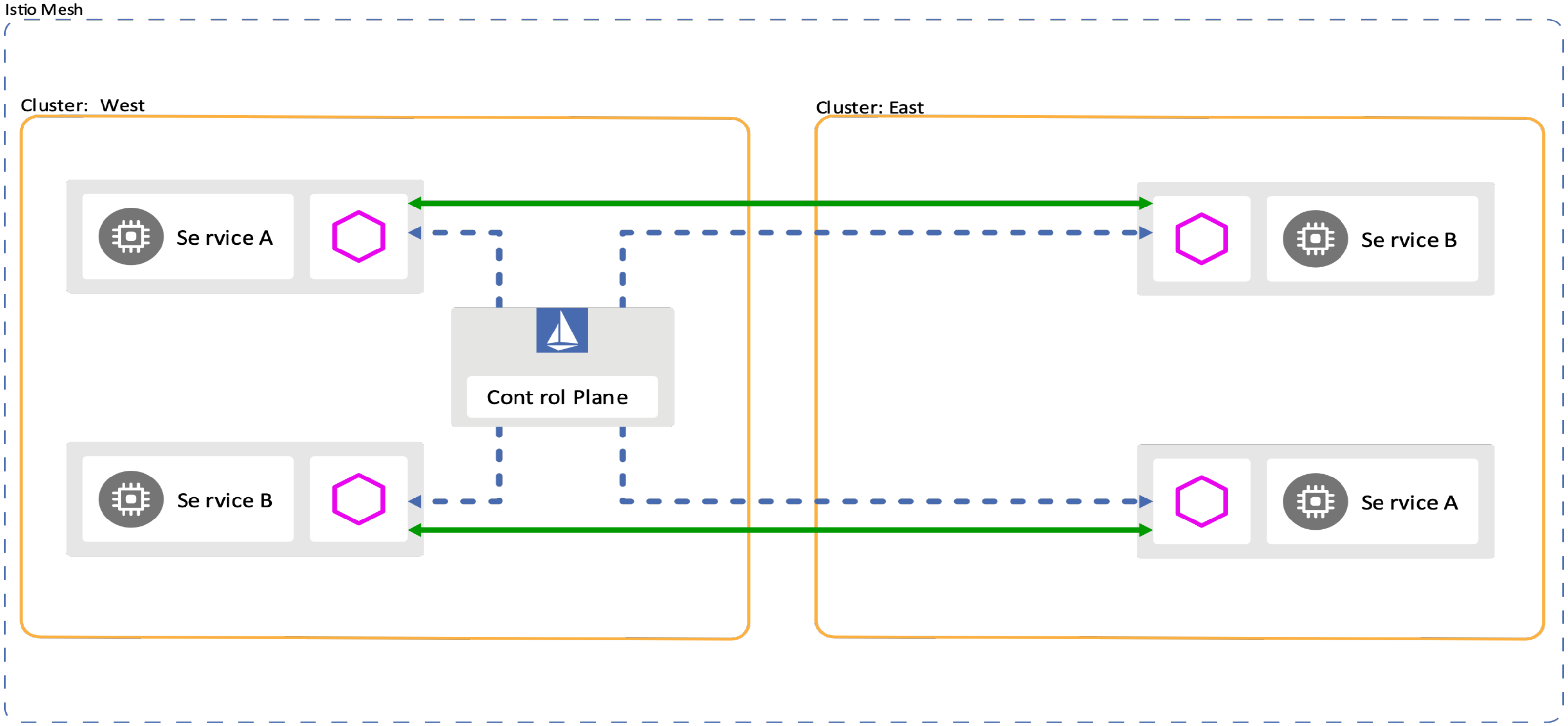
Istio proxy container

https://docs.google.com/document/d/10I_BchyXP0ZEGkuCBM3_q4ny6Ji5ygH5nZdlUJ_PNss

What istiod means for users?

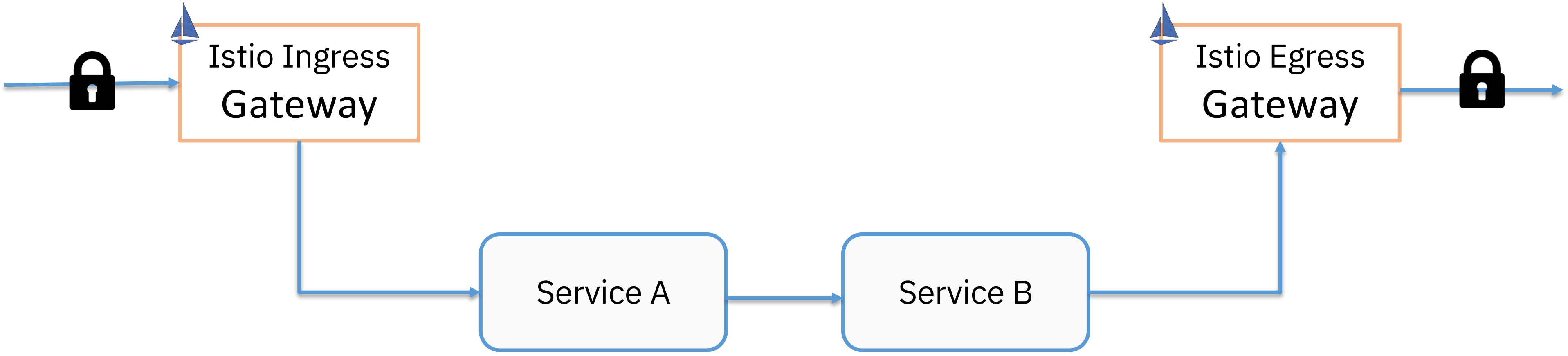
- Operators:
  - Easier to install, easier to manage or upgrade control plane.
  - Easier to configure multiclusters
  - Easier to check for logs
  - Easier to monitor control plane
- Users:
  - **No direct impacts**

Multi-Cluster Deployments – Single Control Plane



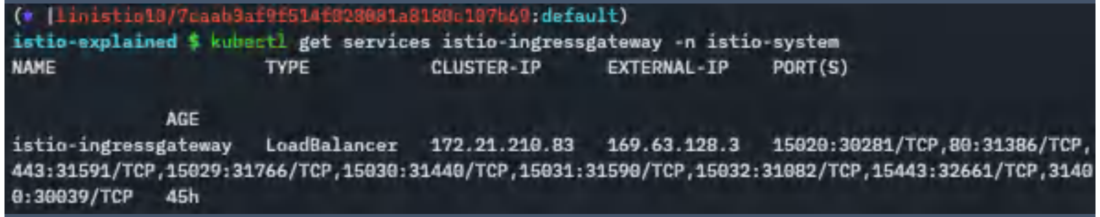
Adoption Approaches: Gateway Only

- Ingress and Egress (optional) gateways
- No sidecar proxies (no mesh)
- Observability only at gateway
- Control inbound and outbound traffic
- Support beyond the Application-Layer

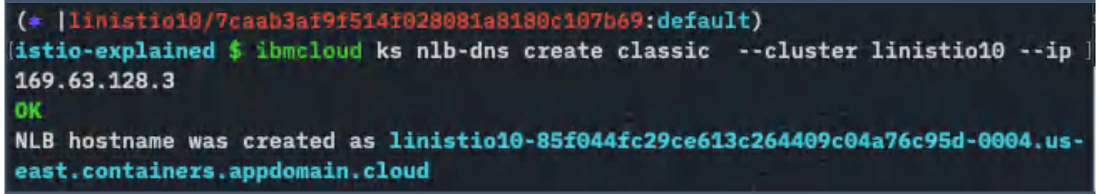


Securing inbound traffic

- Connect Istio Ingress gateway with IBM Cloud NLB

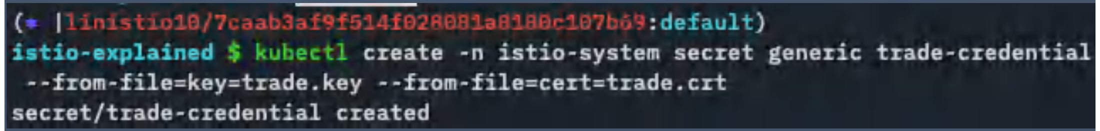


```
(⎈ |linistio10/7caab3af9f514f028081a8180c107b69:default)
istio-explained $ kubectl get services istio-ingressgateway -n istio-system
NAME                   TYPE           CLUSTER-IP      EXTERNAL-IP    PORT(S)
                                                                                    
                 AGE
istio-ingressgateway   LoadBalancer   172.21.210.83   169.63.128.3   15020:30281/TCP,80:31386/TCP,
443:31591/TCP,15029:31766/TCP,15030:31440/TCP,15031:31590/TCP,15032:31082/TCP,15443:32661/TCP,3140
0:30039/TCP   45h
```



```
(⎈ |linistio10/7caab3af9f514f028081a8180c107b69:default)
istio-explained $ ibmcloud ks nlb-dns create classic  --cluster linistio10 --ip 
169.63.128.3
OK
NLB hostname was created as linistio10-85f044fc29ce613c264409c04a76c95d-0004.us-
east.containers.appdomain.cloud
```

- Create credential based on key and cert from NLB hostname

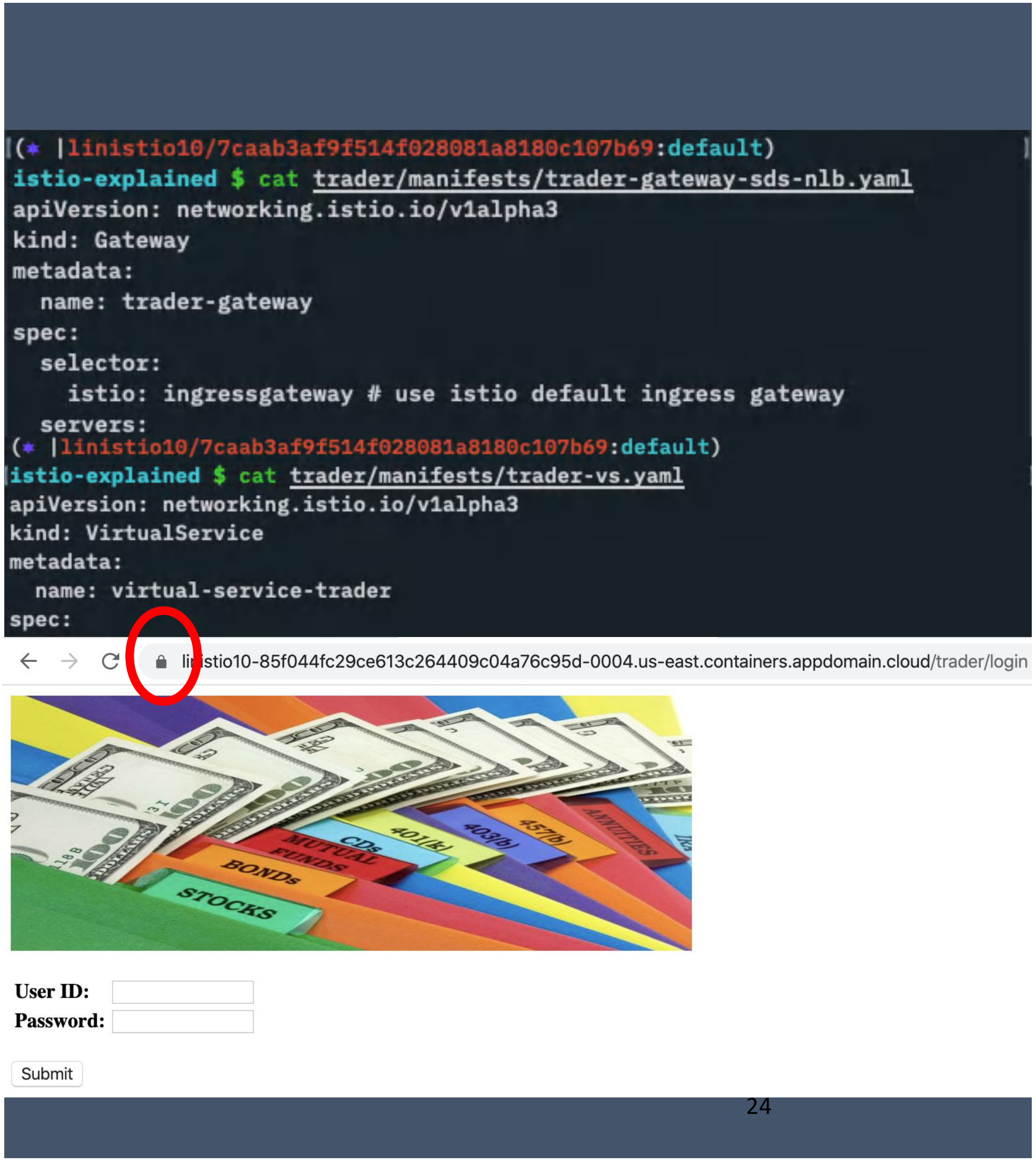


```
(⎈ |linistio10/7caab3af9f514f028081a8180c107b69:default)
istio-explained $ kubectl create -n istio-system secret generic trade-credential
 --from-file=key=trade.key --from-file=cert=trade.crt
secret/trade-credential created
```

Securing inbound traffic

- Secure Gateway configuration

- Access the application inbound traffic securely



```
(* |linistio10/7caab3af9f514f028081a8180c107b69:default)
istio-explained $ cat trader/manifests/trader-gateway-sds-nlb.yaml
apiVersion: networking.istio.io/v1alpha3
kind: Gateway
metadata:
  name: trader-gateway
spec:
  selector:
    istio: ingressgateway # use istio default ingress gateway
  servers:
(* |linistio10/7caab3af9f514f028081a8180c107b69:default)
istio-explained $ cat trader/manifests/trader-vs.yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: virtual-service-trader
spec:
```

linistio10-85f044fc29ce613c264409c04a76c95d-0004.us-east.containers.appdomain.cloud/trader/login

User ID:

Password:

Adoption Approaches: Mesh Observability

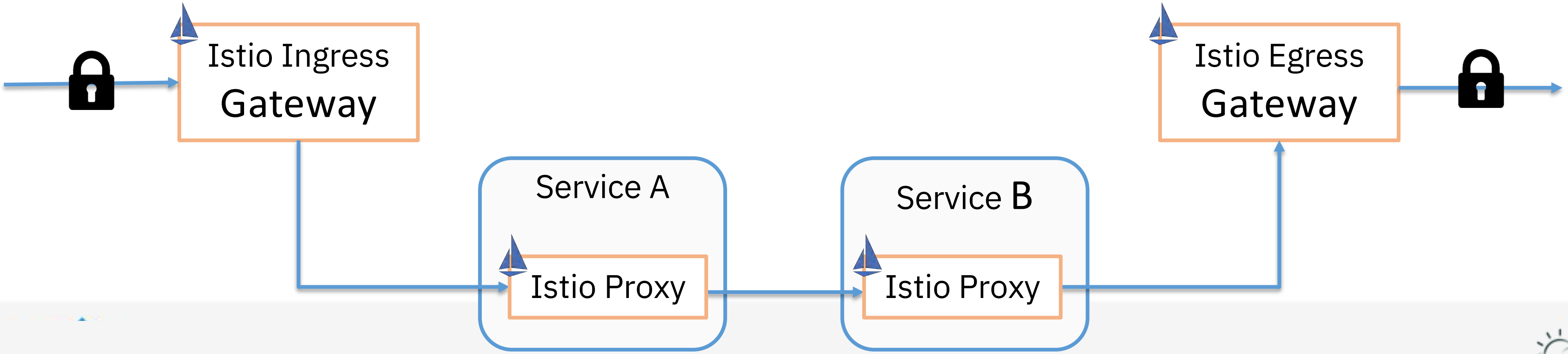
- Incrementally add services to the mesh
- Automatic visibility of interactions between services

33 ops

100%

0 ops

0 ops



Istio Ingress Gateway

Service A

Istio Proxy

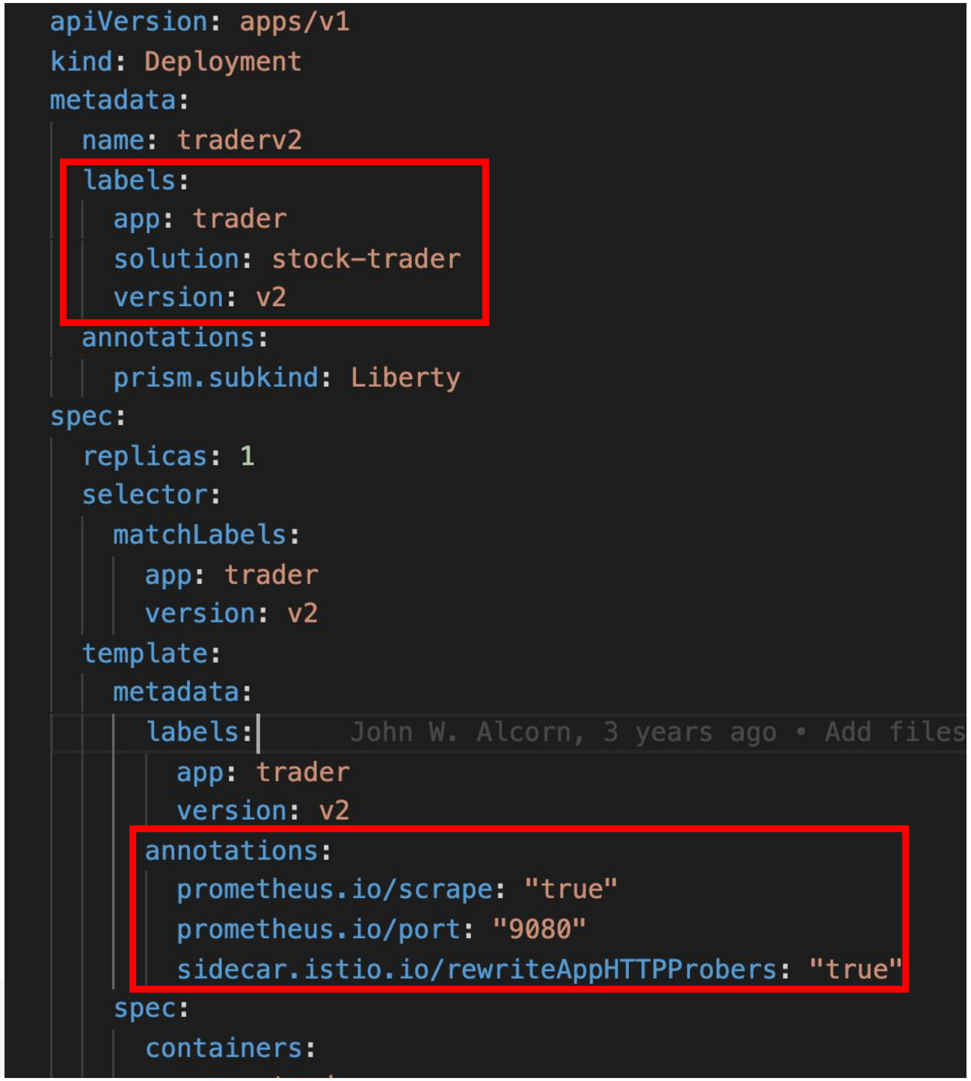
Service B

Istio Proxy

Istio Egress Gateway

Deploy pods and services to the mesh

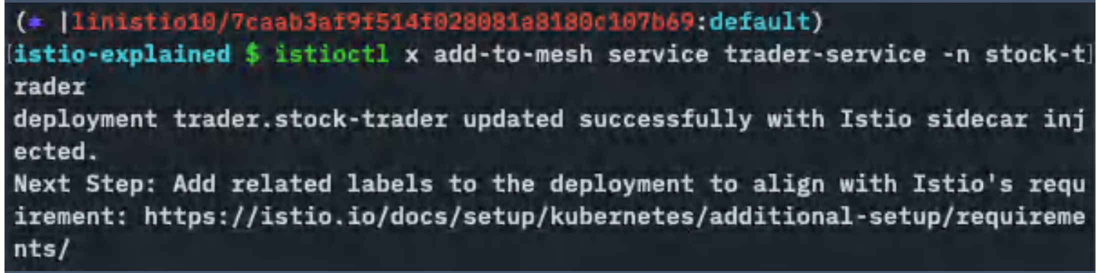
- Add named service port for **each service port**
- Pod must have a service associated
- Label deployments with app and version
- Don't use UID 1337
- Do you have **NET_ADMIN privilege?**
- https://istio.io/docs/setup/kubernetes/prepare/requirements/



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: traderv2
  labels:
    app: trader
    solution: stock-trader
    version: v2
  annotations:
    prism.subkind: Liberty
spec:
  replicas: 1
  selector:
    matchLabels:
      app: trader
      version: v2
  template:
    metadata:
      labels:      John W. Alcorn, 3 years ago • Add files
        app: trader
        version: v2
      annotations:
        prometheus.io/scrape: "true"
        prometheus.io/port: "9080"
        sidecar.istio.io/rewriteAppHTTPProbers: "true"
    spec:
      containers:
```

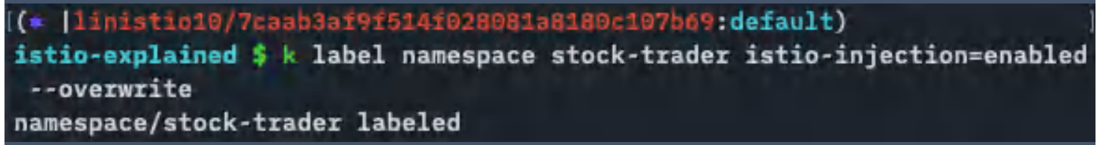
Simplify adding services to the mesh

- Existing Service: add services to the mesh



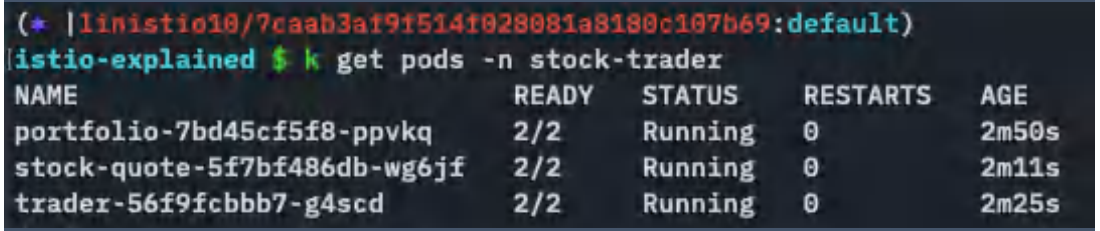
```
(⎈ |linistio10/7caab3af9f514f028081a8180c107b69:default)
istio-explained $ istioctl x add-to-mesh service trader-service -n stock-t
rader
deployment trader.stock-trader updated successfully with Istio sidecar inj
ected.
Next Step: Add related labels to the deployment to align with Istio's requ
irement: https://istio.io/docs/setup/kubernetes/additional-setup/requireme
nts/
```

- New Service: Deploy the services as usual
- Annotate the namespace with istio-injection enabled



```
(⎈ |linistio10/7caab3af9f514f028081a8180c107b69:default)
istio-explained $ k label namespace stock-trader istio-injection=enabled
 --overwrite
namespace/stock-trader labeled
```

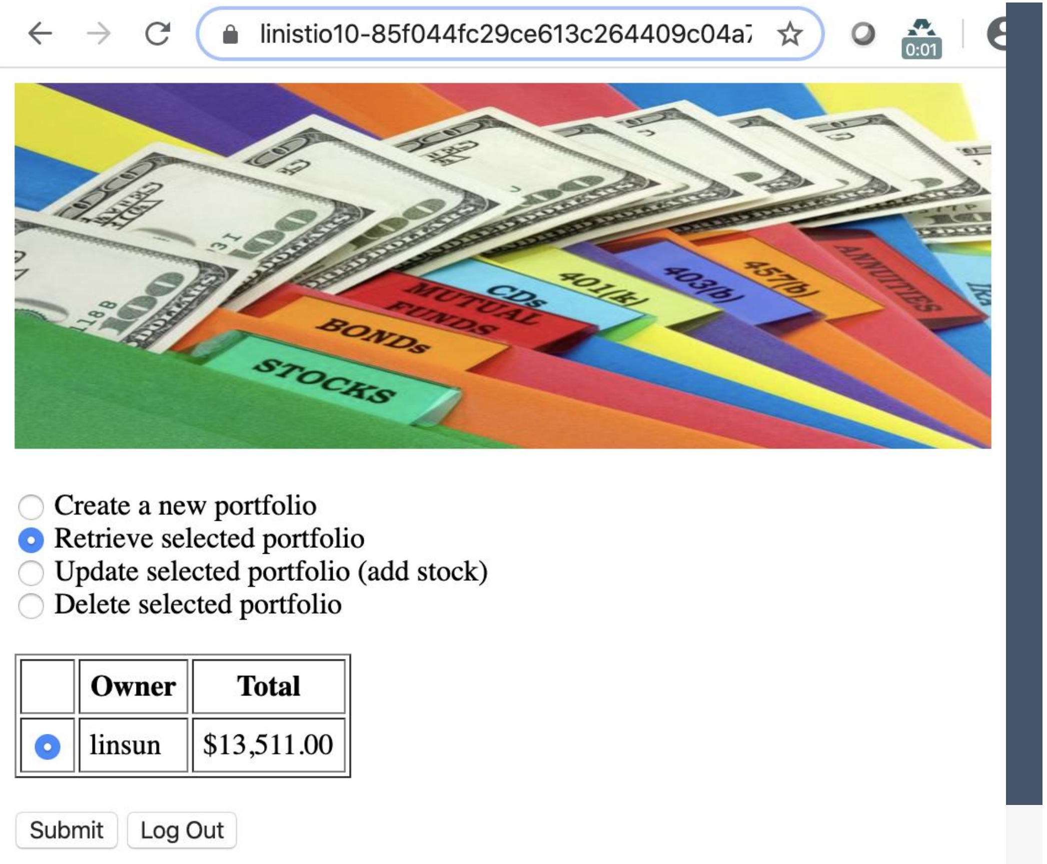
- Confirm pods are running with sidecars



```
(⎈ |linistio10/7caab3af9f514f028081a8180c107b69:default)
istio-explained $ k get pods -n stock-trader
NAME                             READY   STATUS    RESTARTS   AGE
portfolio-7bd45cf5f8-ppvkq       2/2     Running   0          2m50s
stock-quote-5f7bf486db-wg6jf     2/2     Running   0          2m11s
trader-56f9fcbbb7-g4scd          2/2     Running   0          2m25s
```

IBM Observe traffic communication with no change

- Generate some load on the stock trader application

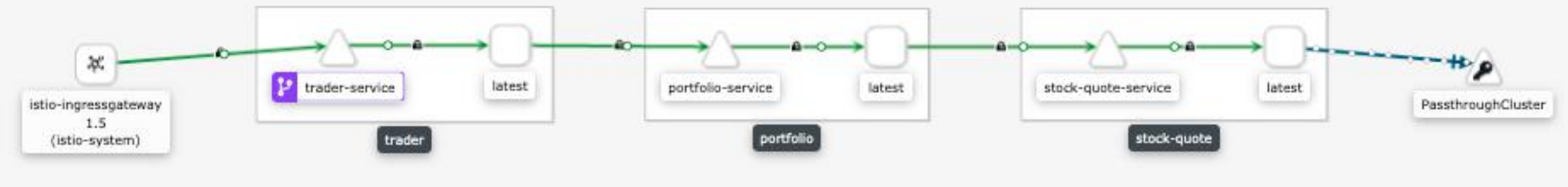


linistio10-85f044fc29ce613c264409c04a7

○ Create a new portfolio
● Retrieve selected portfolio
○ Update selected portfolio (add stock)
○ Delete selected portfolio

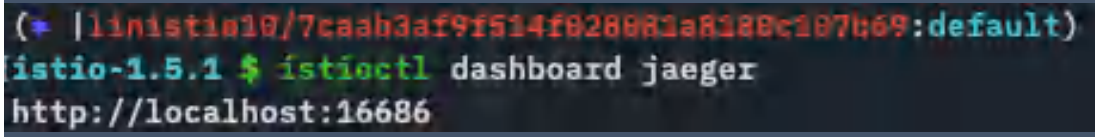
	Owner	Total
●	linsun	$13,511.00

- Visualize traffic animation with Kiali



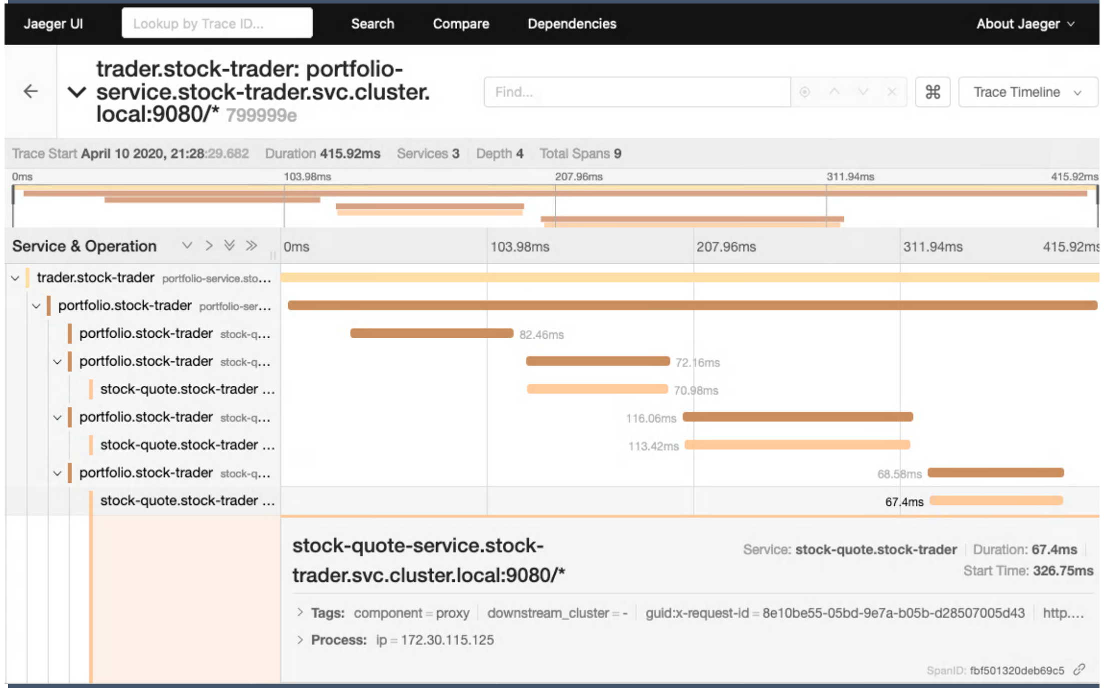
Observe trace spans for each request

- Bring up Jaeger dashboard



```
(⎈ |linistio10/7caab3af9f514f028881a8188c107b69:default)
istio-1.5.1 $ istioctl dashboard jaeger
http://localhost:16686
```

- Click on a request to view trace spans among services



trader.stock-trader: portfolio-service.stock-trader.svc.cluster.local:9080/* 799999e

Trace Start **April 10 2020, 21:28**:29.682 Duration **415.92ms** Services **3** Depth **4** Total Spans **9**

```
0ms        103.98ms        207.96ms        311.94ms        415.92ms

Service & Operation              0ms   103.98ms   207.96ms   311.94ms   415.92ms
trader.stock-trader portfolio-service.sto...
  portfolio.stock-trader portfolio-ser...
    portfolio.stock-trader stock-q...          82.46ms
    portfolio.stock-trader stock-q...          72.16ms
      stock-quote.stock-trader ...             70.98ms
    portfolio.stock-trader stock-q...          116.06ms
      stock-quote.stock-trader ...             113.42ms
    portfolio.stock-trader stock-q...          68.58ms
      stock-quote.stock-trader ...             67.4ms
```

stock-quote-service.stock-trader.svc.cluster.local:9080/*

Service: **stock-quote.stock-trader** | Duration: **67.4ms** | Start Time: **326.75ms**

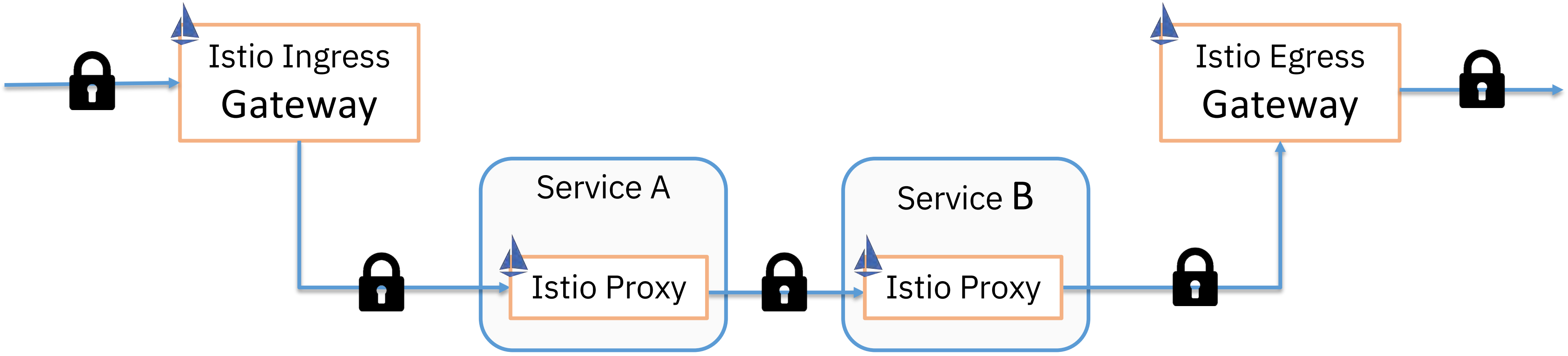
> **Tags:** component = proxy | downstream_cluster = - | guid:x-request-id = 8e10be55-05bd-9e7a-b05b-d28507005d43 | http....

> **Process:** ip = 172.30.115.125

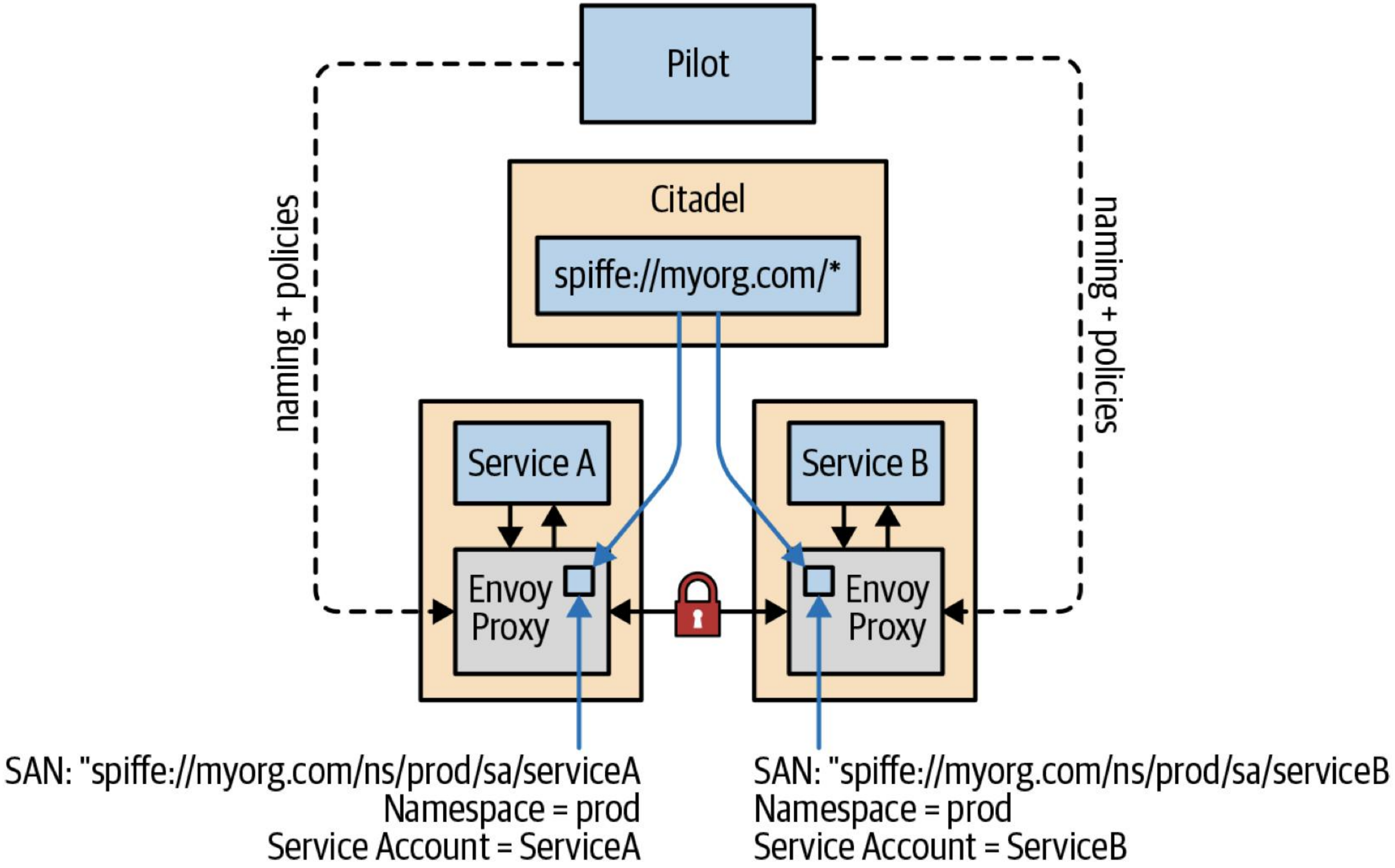
SpanID: fbf501320deb69c5

Adoption Approaches: Secure Mesh

- Incrementally add services to the mesh
- Incrementally set secure communication (mTLS)
- Auto mTLS to set client security settings based on target service

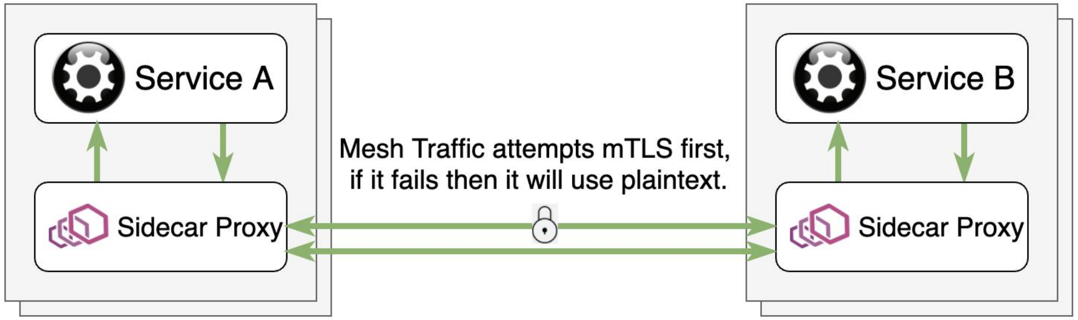


Istio Security Architecture

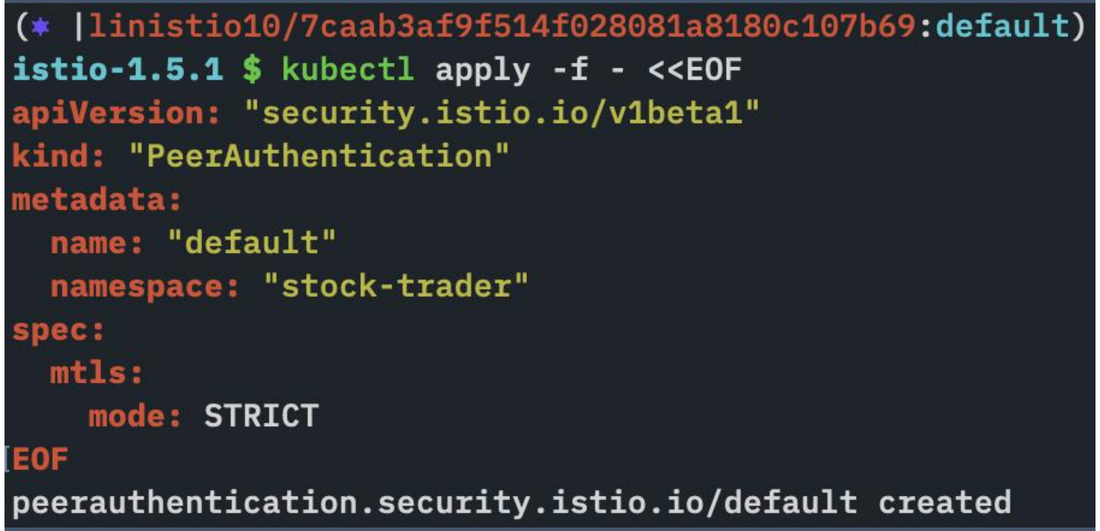

Pilot
Citadel
spiffe://myorg.com/*
naming + policies
naming + policies
Service A
Service B
Envoy Proxy
Envoy Proxy
SAN: "spiffe://myorg.com/ns/prod/sa/serviceA
Namespace = prod
Service Account = ServiceA
SAN: "spiffe://myorg.com/ns/prod/sa/serviceB
Namespace = prod
Service Account = ServiceB

Securing communication Within Istio

- Permissive mTLS improves mTLS onboarding experience.


Service A
Sidecar Proxy
Mesh Traffic attempts mTLS first, if it fails then it will use plaintext.
Service B
Sidecar Proxy

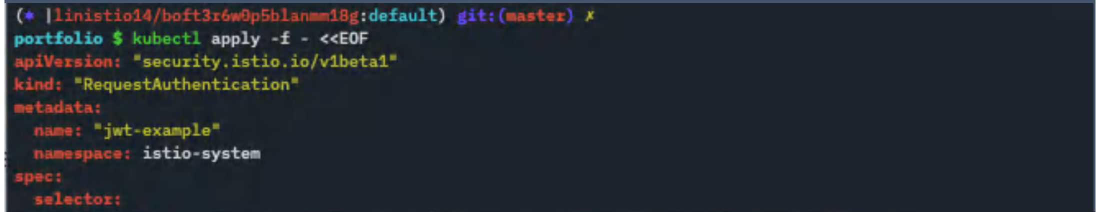
- Enforce mTLS traffic in the mesh with authentication policy



```
(⎈ |linistio10/7caab3af9f514f028081a8180c107b69:default)
istio-1.5.1 $ kubectl apply -f - <<EOF
apiVersion: "security.istio.io/v1beta1"
kind: "PeerAuthentication"
metadata:
  name: "default"
  namespace: "stock-trader"
spec:
  mtls:
    mode: STRICT
EOF
peerauthentication.security.istio.io/default created
```

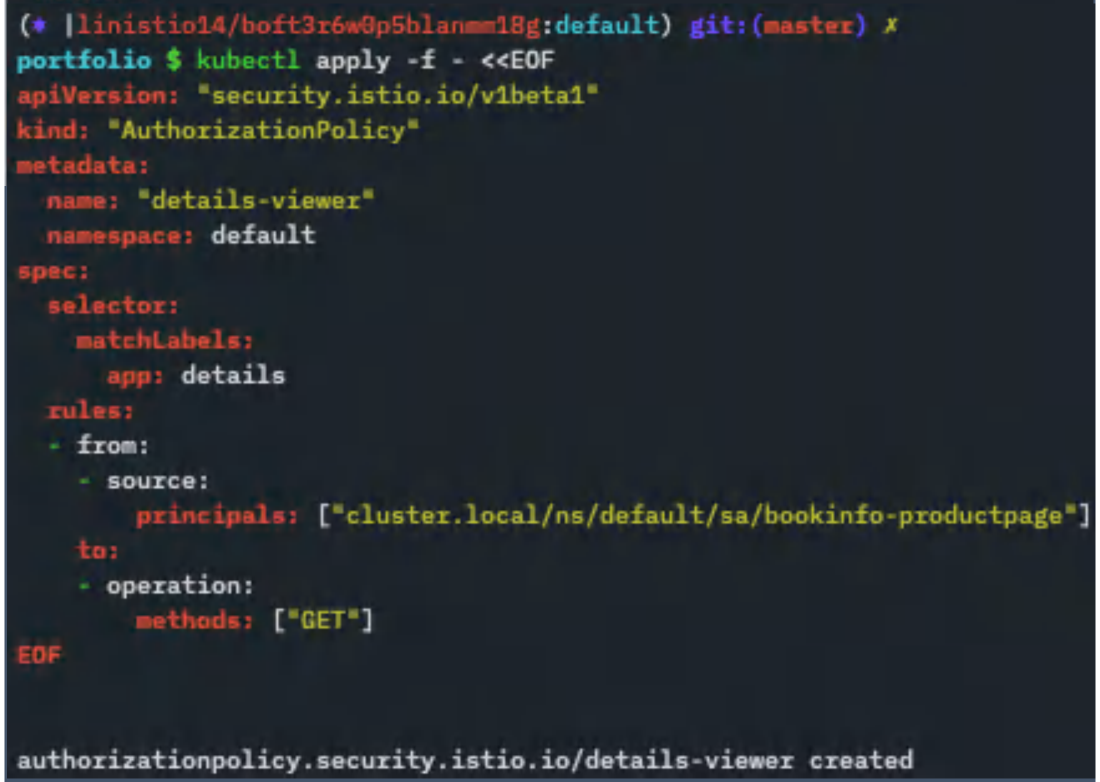
Securing communication Within Istio

- End user authentication



```
(⎈ |linistio14/boft3r6w0p5blanmm18g:default) git:(master) ✗
portfolio $ kubectl apply -f - <<EOF
apiVersion: "security.istio.io/v1beta1"
kind: "RequestAuthentication"
metadata:
  name: "jwt-example"
  namespace: istio-system
spec:
  selector:
```

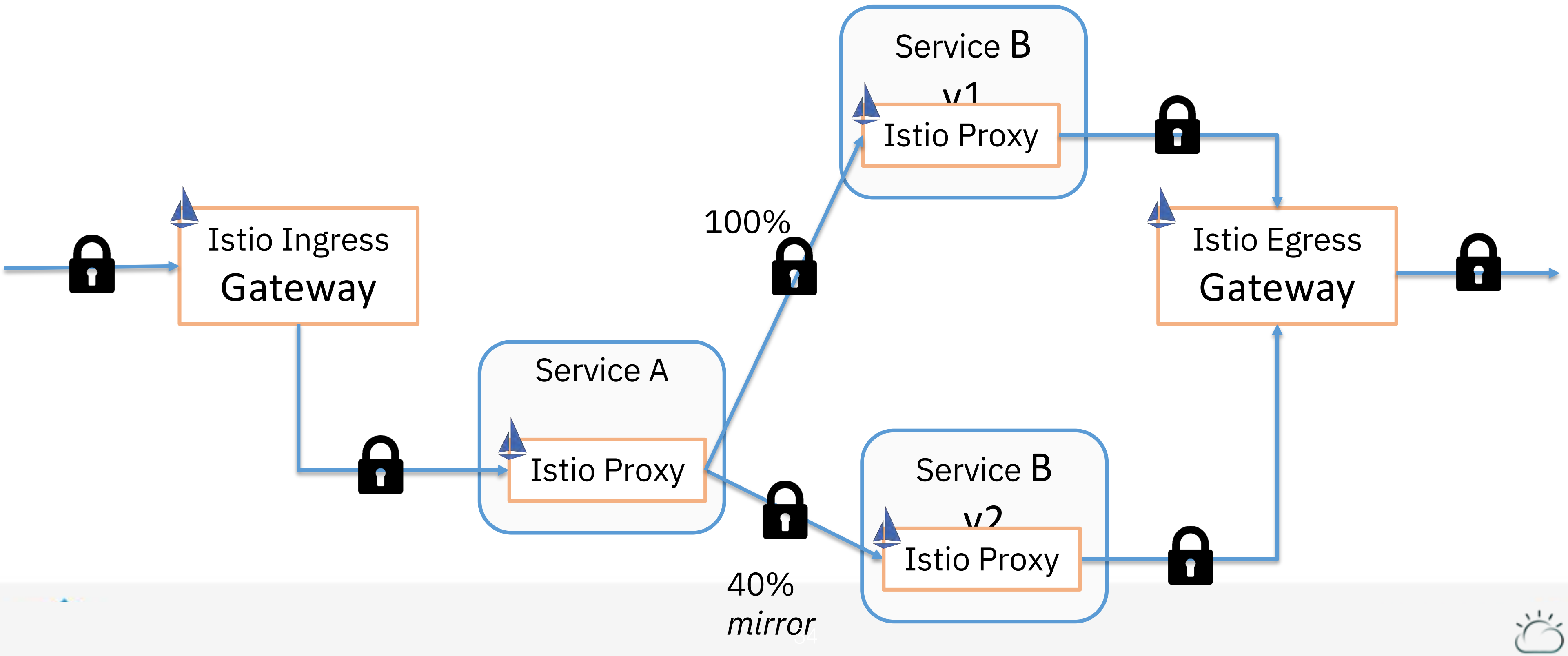
- Authorization



```
(⎈ |linistio14/boft3r6w0p5blanmm18g:default) git:(master) ✗
portfolio $ kubectl apply -f - <<EOF
apiVersion: "security.istio.io/v1beta1"
kind: "AuthorizationPolicy"
metadata:
  name: "details-viewer"
  namespace: default
spec:
  selector:
    matchLabels:
      app: details
  rules:
  - from:
    - source:
        principals: ["cluster.local/ns/default/sa/bookinfo-productpage"]
    to:
    - operation:
        methods: ["GET"]
EOF


authorizationpolicy.security.istio.io/details-viewer created
```

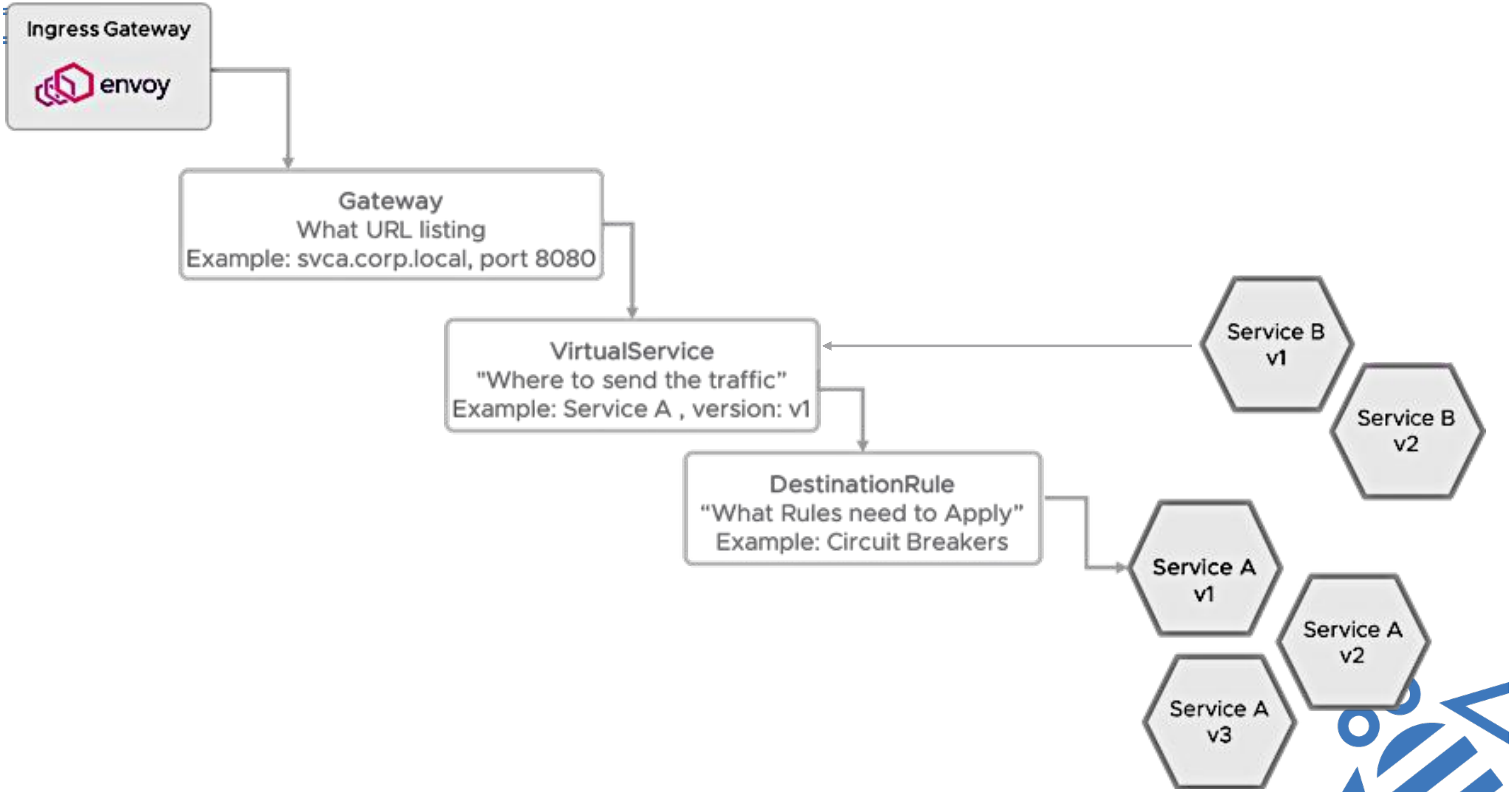
Adoption Approaches: Traffic Management

- Route traffic between versions
- Add resiliency via circuit breakers, time outs, and retries



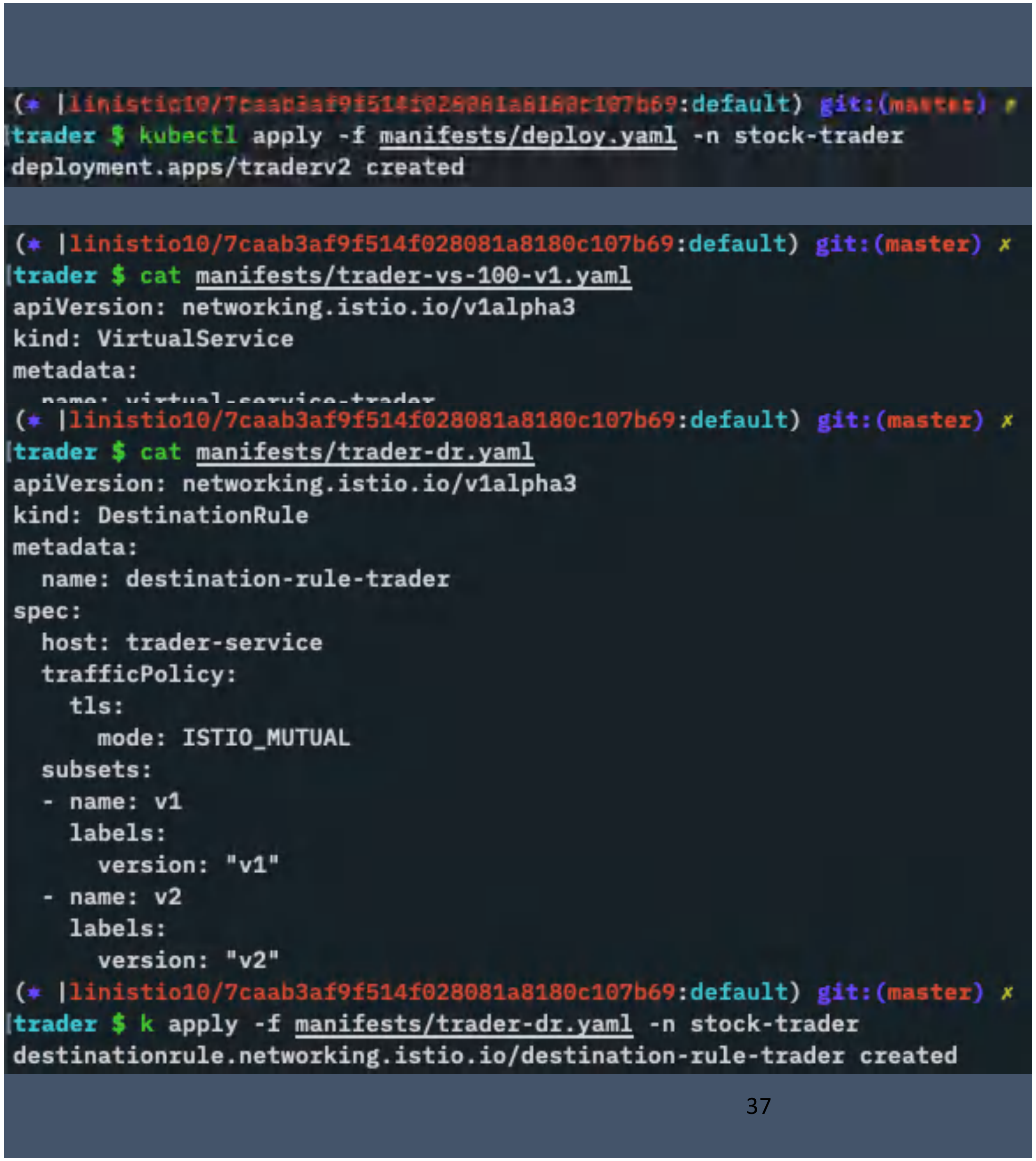
Istio Network Resources

Gateway	Edge load balancer configuration
Virtual Service	List of routing rules
Destination Rule	Policies applied to a destination
Service entry	Access external services by services
Sidecar	Sidecar proxy configuration scope


Ingress Gateway
envoy
Gateway
What URL listing
Example: svca.corp.local, port 8080
VirtualService
"Where to send the traffic"
Example: Service A , version: v1
DestinationRule
"What Rules need to Apply"
Example: Circuit Breakers
Service B v1
Service B v2
Service A v1
Service A v2
Service A v3

http://www.routetocloud.com/2019/04/introduction-to-istio-service-mesh/

- Dark Launch

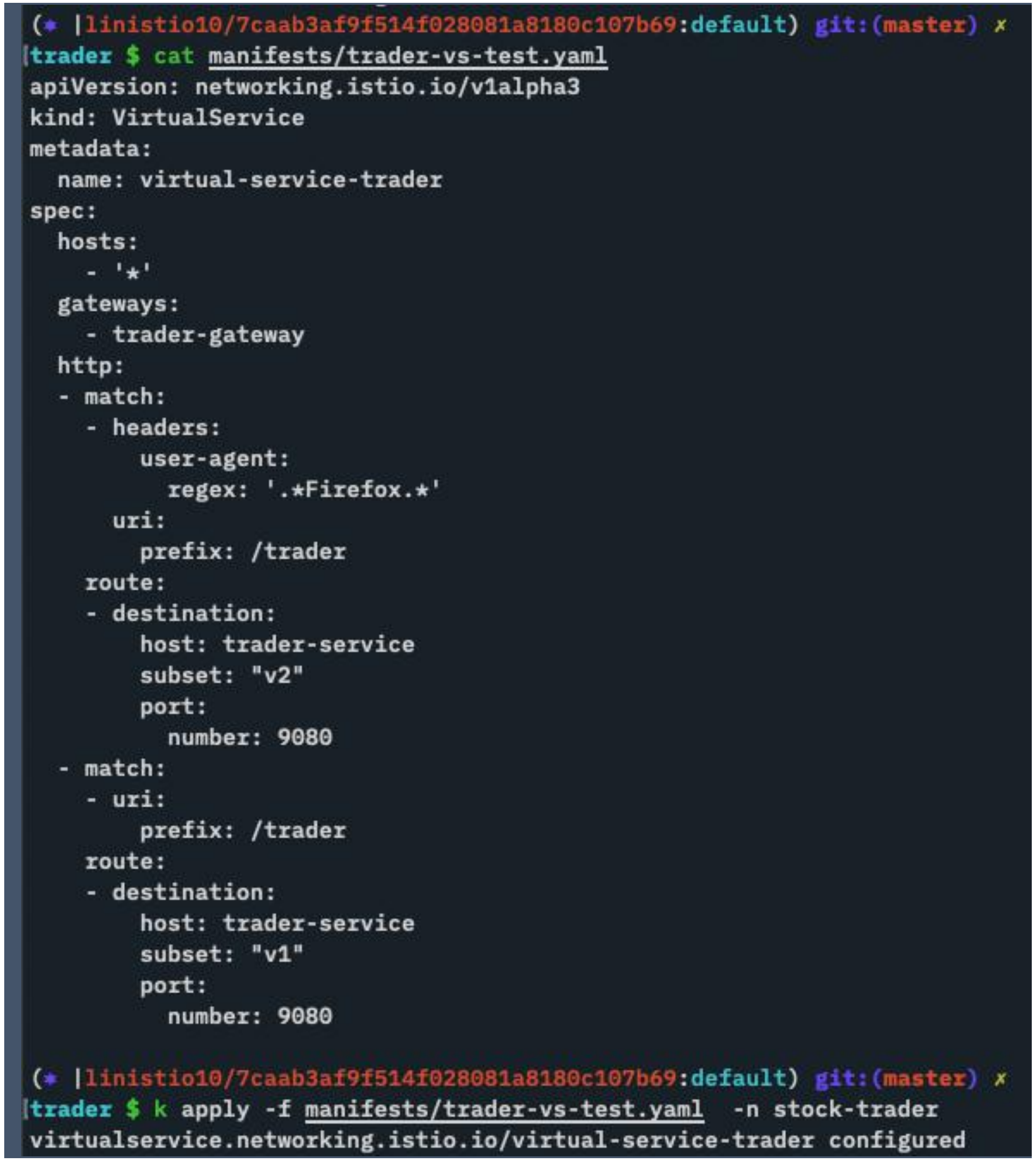


```
(⎈ |linistio10/7caab3af9f514f028081a8180c107b69:default) git:(master) ✗
trader $ kubectl apply -f manifests/deploy.yaml -n stock-trader
deployment.apps/traderv2 created
```

```
(⎈ |linistio10/7caab3af9f514f028081a8180c107b69:default) git:(master) ✗
trader $ cat manifests/trader-vs-100-v1.yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: virtual-service-trader
```

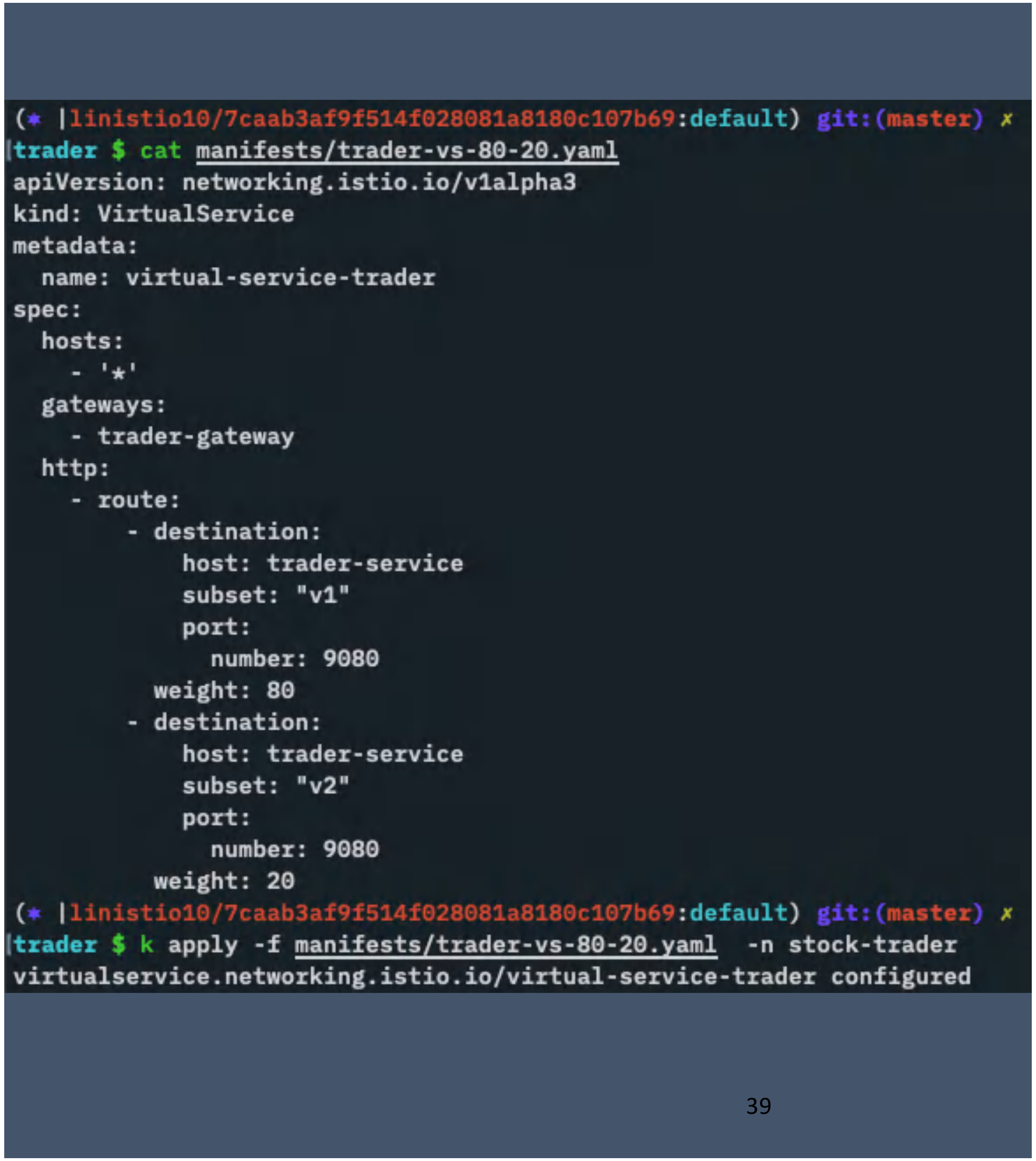
```
(⎈ |linistio10/7caab3af9f514f028081a8180c107b69:default) git:(master) ✗
trader $ cat manifests/trader-dr.yaml
apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: destination-rule-trader
spec:
  host: trader-service
  trafficPolicy:
    tls:
      mode: ISTIO_MUTUAL
  subsets:
  - name: v1
    labels:
      version: "v1"
  - name: v2
    labels:
      version: "v2"
(⎈ |linistio10/7caab3af9f514f028081a8180c107b69:default) git:(master) ✗
trader $ k apply -f manifests/trader-dr.yaml -n stock-trader
destinationrule.networking.istio.io/destination-rule-trader created
```

- Selectively Route Requests



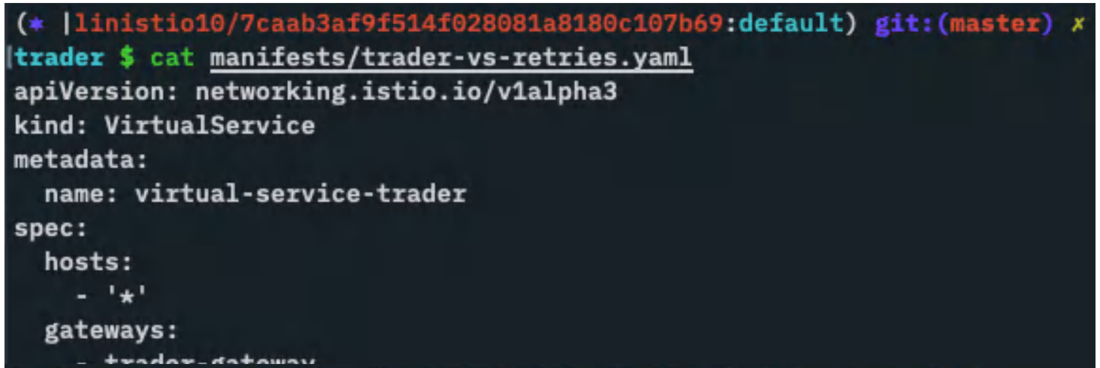
```
(⎈ |linistio10/7caab3af9f514f028081a8180c107b69:default) git:(master) ✗
trader $ cat manifests/trader-vs-test.yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: virtual-service-trader
spec:
  hosts:
    - '*'
  gateways:
    - trader-gateway
  http:
  - match:
    - headers:
        user-agent:
          regex: '.*Firefox.*'
      uri:
        prefix: /trader
    route:
    - destination:
        host: trader-service
        subset: "v2"
        port:
          number: 9080
  - match:
    - uri:
        prefix: /trader
    route:
    - destination:
        host: trader-service
        subset: "v1"
        port:
          number: 9080

(⎈ |linistio10/7caab3af9f514f028081a8180c107b69:default) git:(master) ✗
trader $ k apply -f manifests/trader-vs-test.yaml  -n stock-trader
virtualservice.networking.istio.io/virtual-service-trader configured
```

- Canary Testing

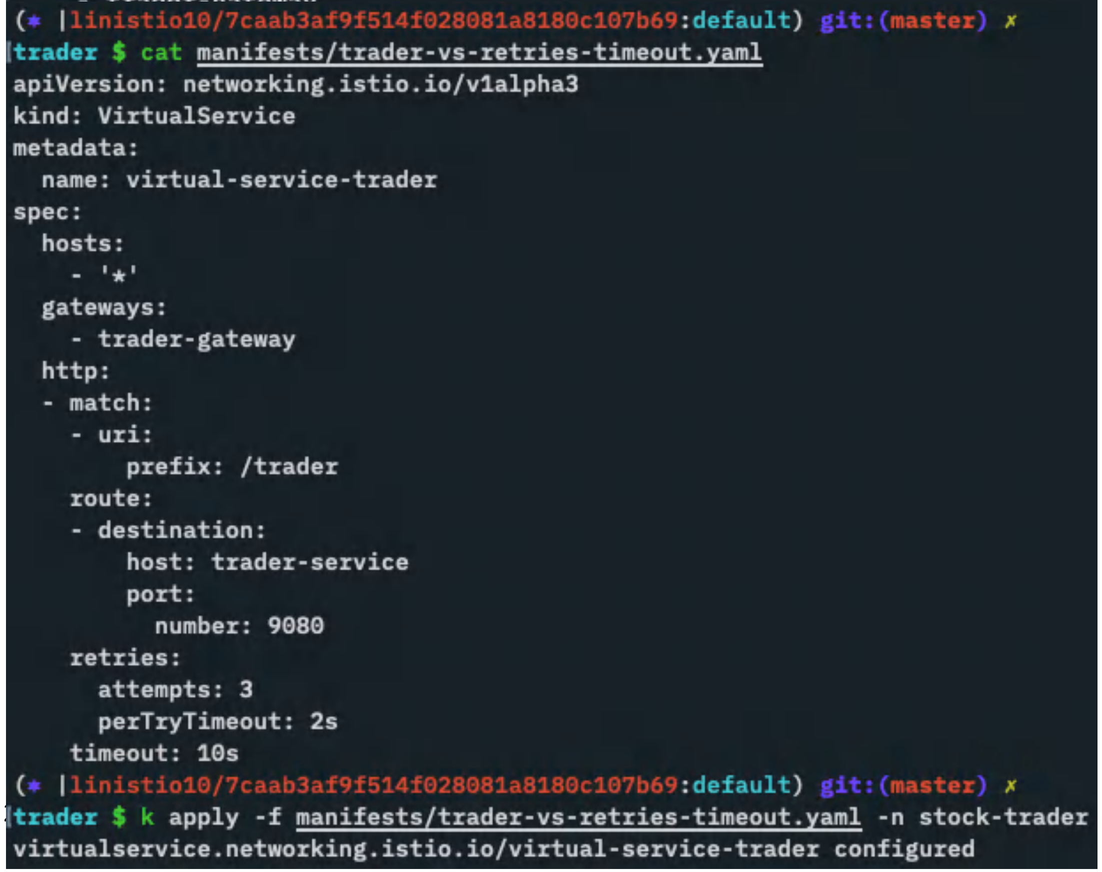


```
(* |linistio10/7caab3af9f514f028081a8180c107b69:default) git:(master) ✗
trader $ cat manifests/trader-vs-80-20.yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: virtual-service-trader
spec:
  hosts:
    - '*'
  gateways:
    - trader-gateway
  http:
    - route:
        - destination:
            host: trader-service
            subset: "v1"
            port:
              number: 9080
          weight: 80
        - destination:
            host: trader-service
            subset: "v2"
            port:
              number: 9080
          weight: 20
(* |linistio10/7caab3af9f514f028081a8180c107b69:default) git:(master) ✗
trader $ k apply -f manifests/trader-vs-80-20.yaml  -n stock-trader
virtualservice.networking.istio.io/virtual-service-trader configured
```

- Resiliency

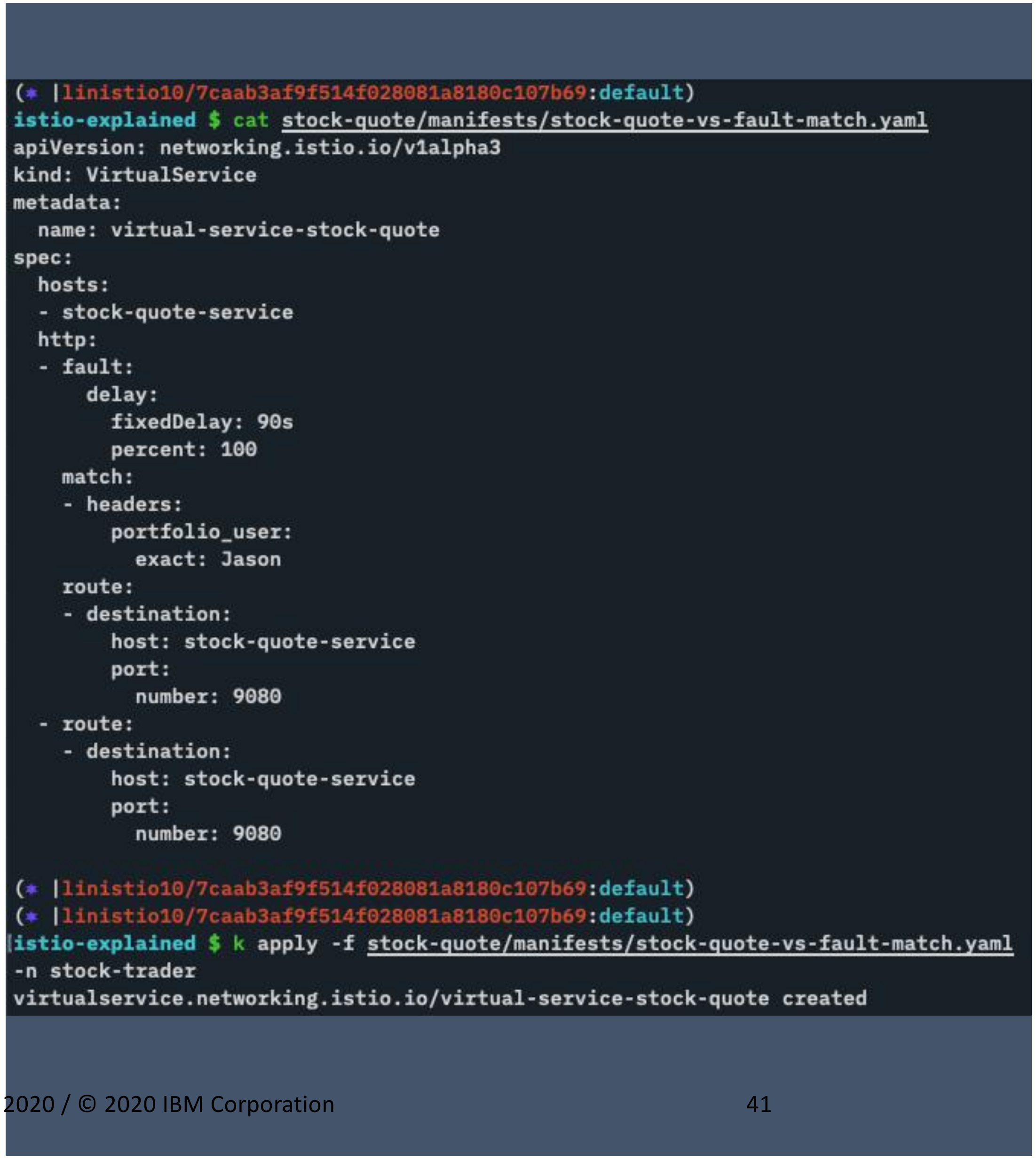


```
(* |linistio10/7caab3af9f514f028081a8180c107b69:default) git:(master) ✗
trader $ cat manifests/trader-vs-retries.yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: virtual-service-trader
spec:
  hosts:
    - '*'
  gateways:
    - trader-gateway
```



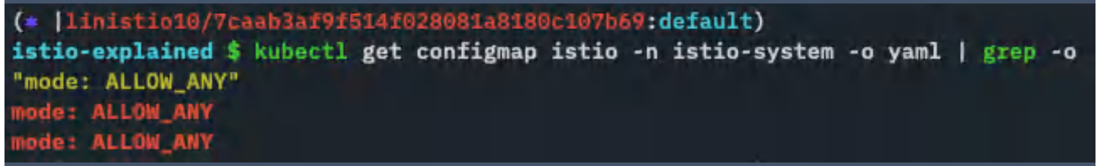
```
(* |linistio10/7caab3af9f514f028081a8180c107b69:default) git:(master) ✗
trader $ cat manifests/trader-vs-retries-timeout.yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: virtual-service-trader
spec:
  hosts:
    - '*'
  gateways:
    - trader-gateway
  http:
  - match:
    - uri:
        prefix: /trader
    route:
    - destination:
        host: trader-service
        port:
          number: 9080
    retries:
      attempts: 3
      perTryTimeout: 2s
    timeout: 10s
(* |linistio10/7caab3af9f514f028081a8180c107b69:default) git:(master) ✗
trader $ k apply -f manifests/trader-vs-retries-timeout.yaml -n stock-trader
virtualservice.networking.istio.io/virtual-service-trader configured
```

- Chaos Testing

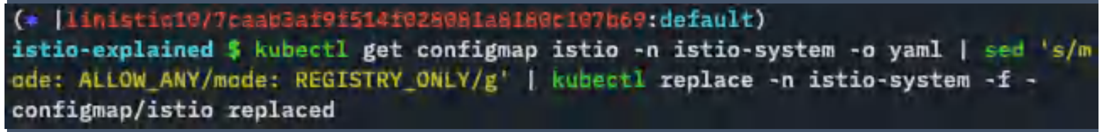


```
(* |linistio10/7caab3af9f514f028081a8180c107b69:default)
istio-explained $ cat stock-quote/manifests/stock-quote-vs-fault-match.yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: virtual-service-stock-quote
spec:
  hosts:
  - stock-quote-service
  http:
  - fault:
      delay:
        fixedDelay: 90s
        percent: 100
    match:
    - headers:
        portfolio_user:
          exact: Jason
    route:
    - destination:
        host: stock-quote-service
        port:
          number: 9080
  - route:
    - destination:
        host: stock-quote-service
        port:
          number: 9080

(* |linistio10/7caab3af9f514f028081a8180c107b69:default)
(* |linistio10/7caab3af9f514f028081a8180c107b69:default)
istio-explained $ k apply -f stock-quote/manifests/stock-quote-vs-fault-match.yaml
-n stock-trader
virtualservice.networking.istio.io/virtual-service-stock-quote created
```

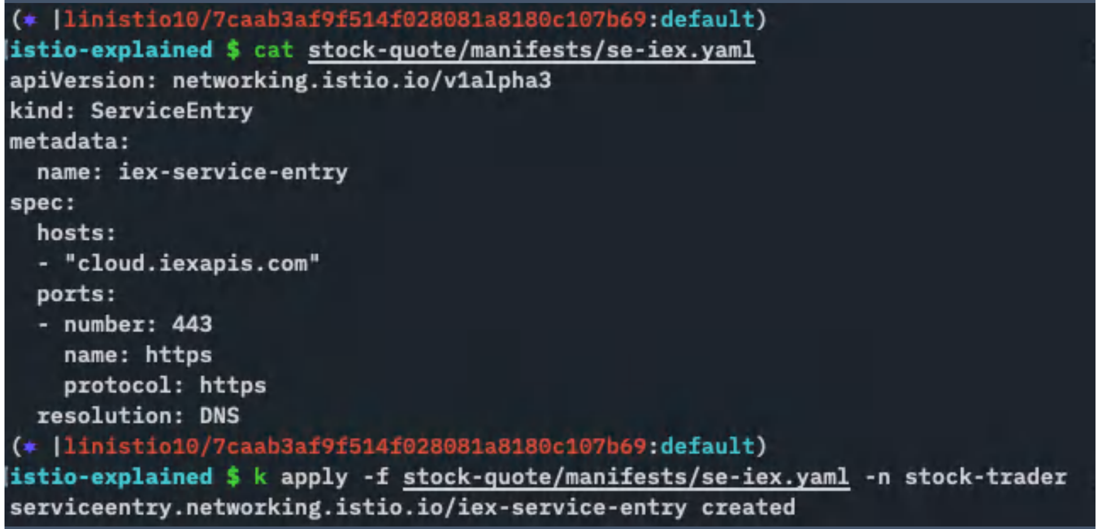
- Control Outbound Traffic



```
(* |linistio10/7caab3af9f514f028081a8180c107b69:default)
istio-explained $ kubectl get configmap istio -n istio-system -o yaml | grep -o
"mode: ALLOW_ANY"
mode: ALLOW_ANY
mode: ALLOW_ANY
```



```
(* |linistio10/7caab3af9f514f028081a8180c107b69:default)
istio-explained $ kubectl get configmap istio -n istio-system -o yaml | sed 's/m
ode: ALLOW_ANY/mode: REGISTRY_ONLY/g' | kubectl replace -n istio-system -f -
configmap/istio replaced
```



```
(* |linistio10/7caab3af9f514f028081a8180c107b69:default)
istio-explained $ cat stock-quote/manifests/se-iex.yaml
apiVersion: networking.istio.io/v1alpha3
kind: ServiceEntry
metadata:
  name: iex-service-entry
spec:
  hosts:
  - "cloud.iexapis.com"
  ports:
  - number: 443
    name: https
    protocol: https
  resolution: DNS
(* |linistio10/7caab3af9f514f028081a8180c107b69:default)
istio-explained $ k apply -f stock-quote/manifests/se-iex.yaml -n stock-trader
serviceentry.networking.istio.io/iex-service-entry created
```

IBM开源技术微讲堂

Istio系列

第2讲完

http://ibm.biz/opentech-ma