Istio in Practice - Day 2

Introductions

Each person say:

- Hello & where they are from
- How much k8s/Istio experience they have
- The main thing they want to learn
- Favourite something computer related
- Favourite something not computer related

Training

Learn and engage directly alongside our team, with courses for all stages of your Kubernetes journey

Workshops

Enhancing Kubernetes knowledge with our workshops. Whichever stage of your cloud native journey you are on we can help you do it better

Consulting

Consulting and engineering to make the most of Kubernetes and move you to production quickly

Subscription

Reference architecture, online training and SLA support 24x7 for your production Kubernetes deployment

Agenda

- Envoy Deep Dive
- Mixer Deep Dive
- Extensibility
- Fault Injection
- Circuit Breaking
- Traffic Mirroring
- Deployment Models

What is a Service Mesh?

An architectural pattern that provides common network services as a feature of the infrastructure

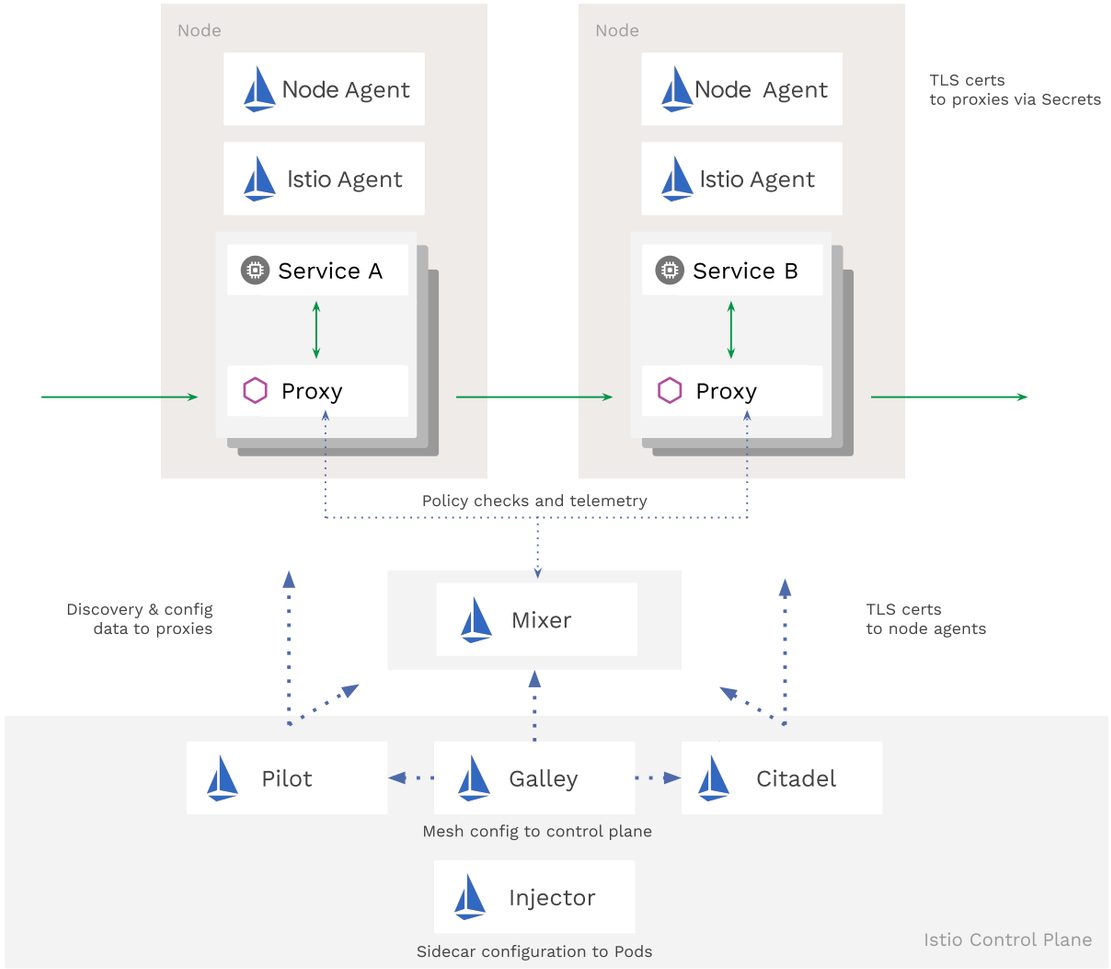
Why Do We Need a Service Mesh?

The scale and complexity of microservice architectures and distributed systems benefits greatly from a dedicated layer to orchestrate cross cutting concerns

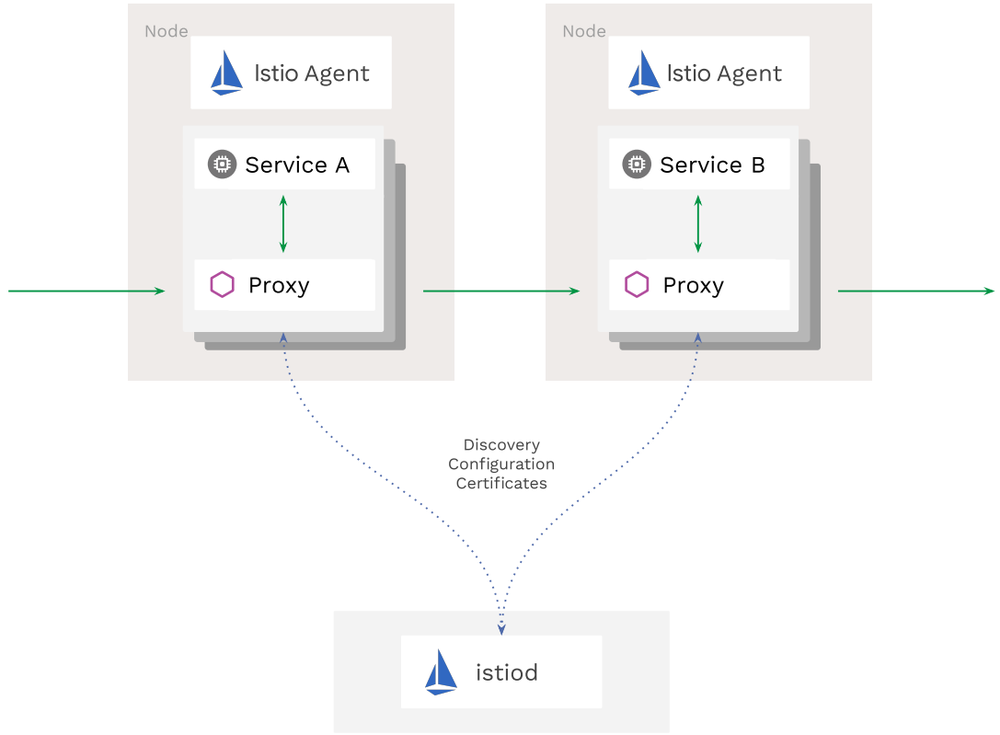
- Facilitates service-to-service communication between microservices
- Abstracts the network and allows operators to think in terms of services rather than specific IPs and ports
- Functionality includes discovery, load balancing, failure recovery, policy enforcement, monitoring and telemetry

Service Mesh Implementations

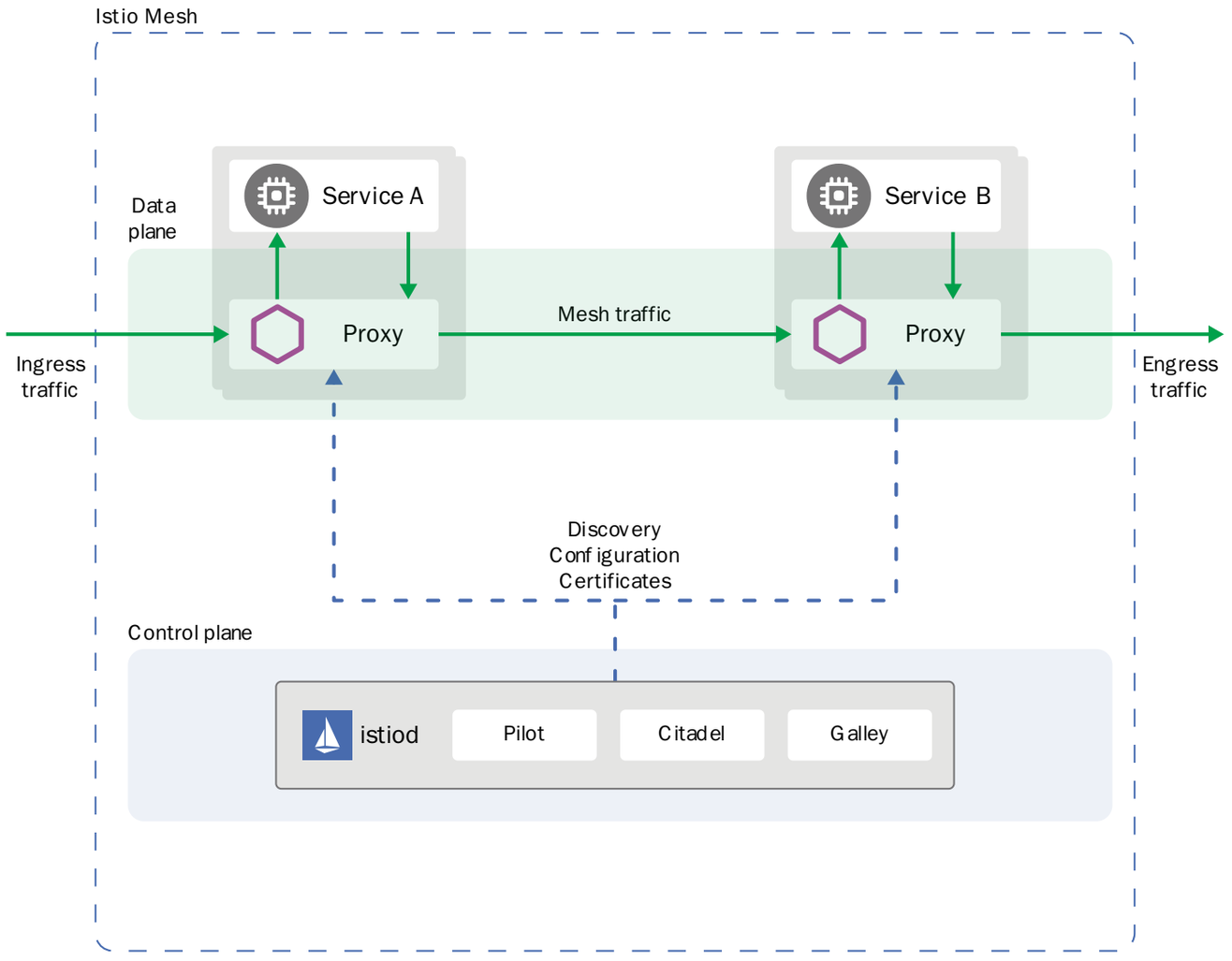
Istio Architecture v1.4.X



Istio Architecture v1.5.X+



Istio Architecture v1.5.X+



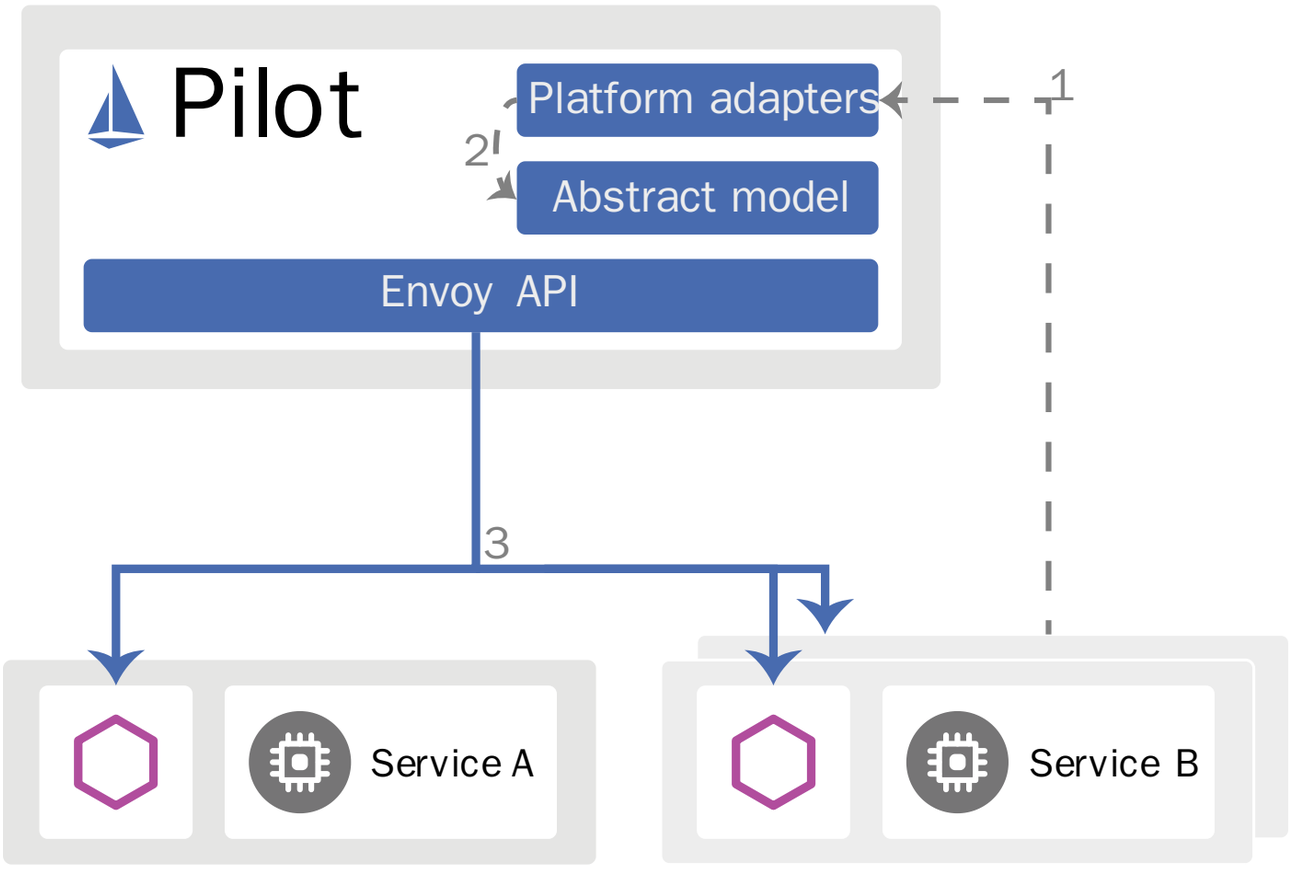
https://istio.io/docs/ops/deployment/architecture/arch.svg

Control Plane / `istiod`

- Pilot
- Citadel
- Galley
- Sidecar Injection

Pilot

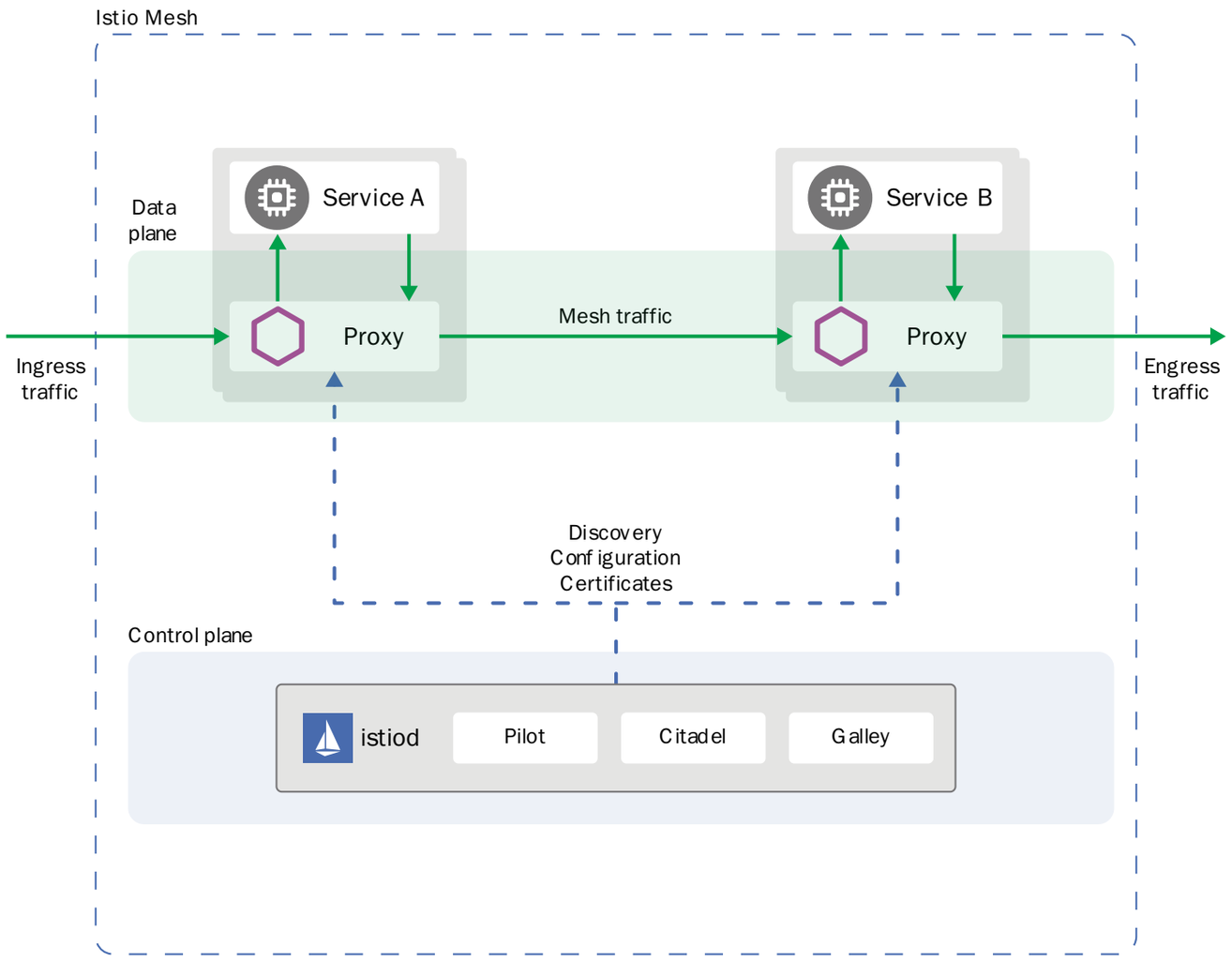
Serves xDS APIs to configure service discovery, traffic management capabilities and security for the Envoy sidecars



https://istio.io/latest/docs/ops/deployment/architecture/discovery.svg

Citadel

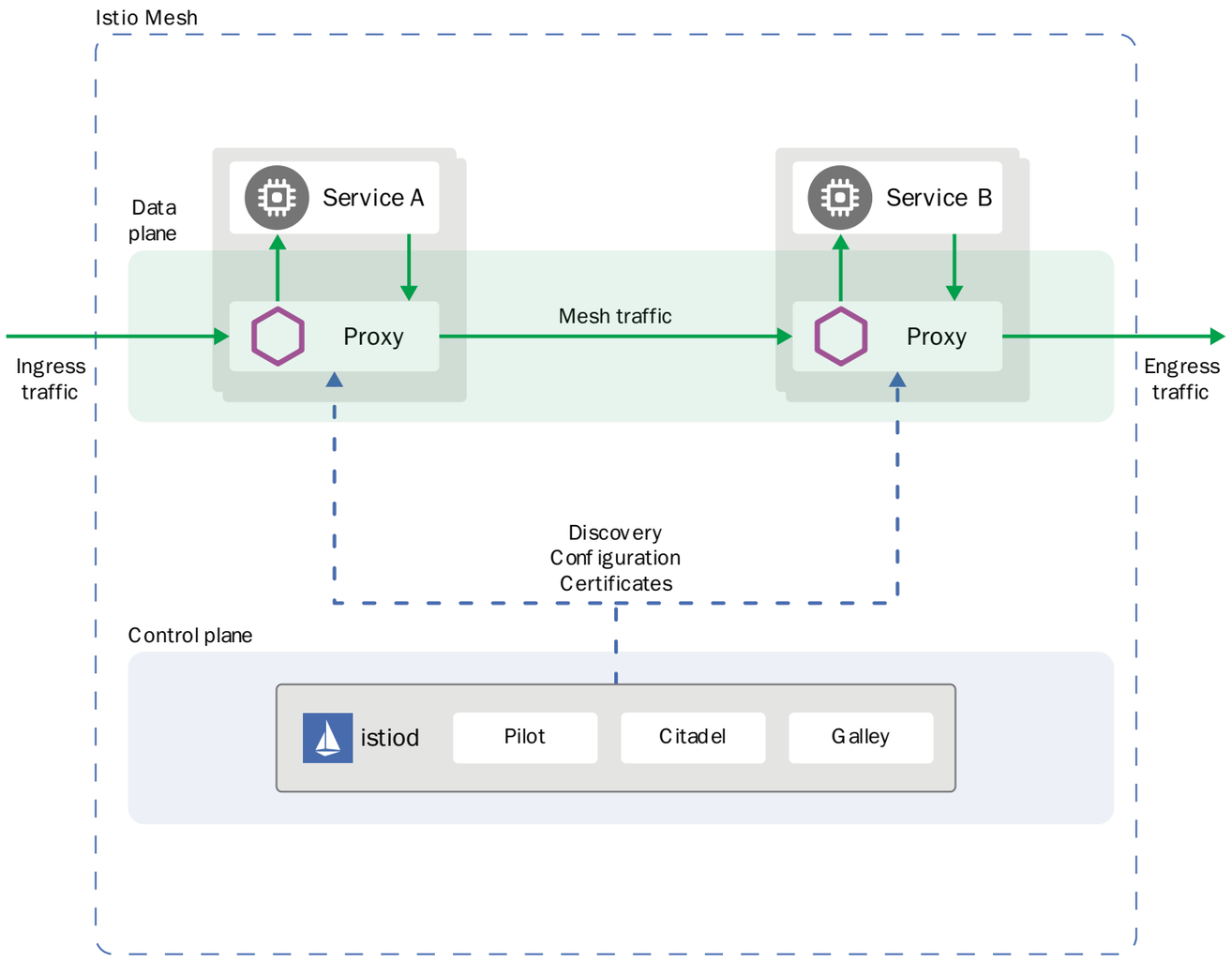
Enables strong service-to-service and end-user authentication with built-in identity and credential management



https://istio.io/latest/docs/ops/deployment/architecture/arch.svg

Galley

Converts platform specific CRDs into component configuration which can be consumed (e.g. by Pilot) over the Mesh Configuration Protocol



https://istio.io/latest/docs/ops/deployment/architecture/arch.svg

Sidecar Injection

Manual

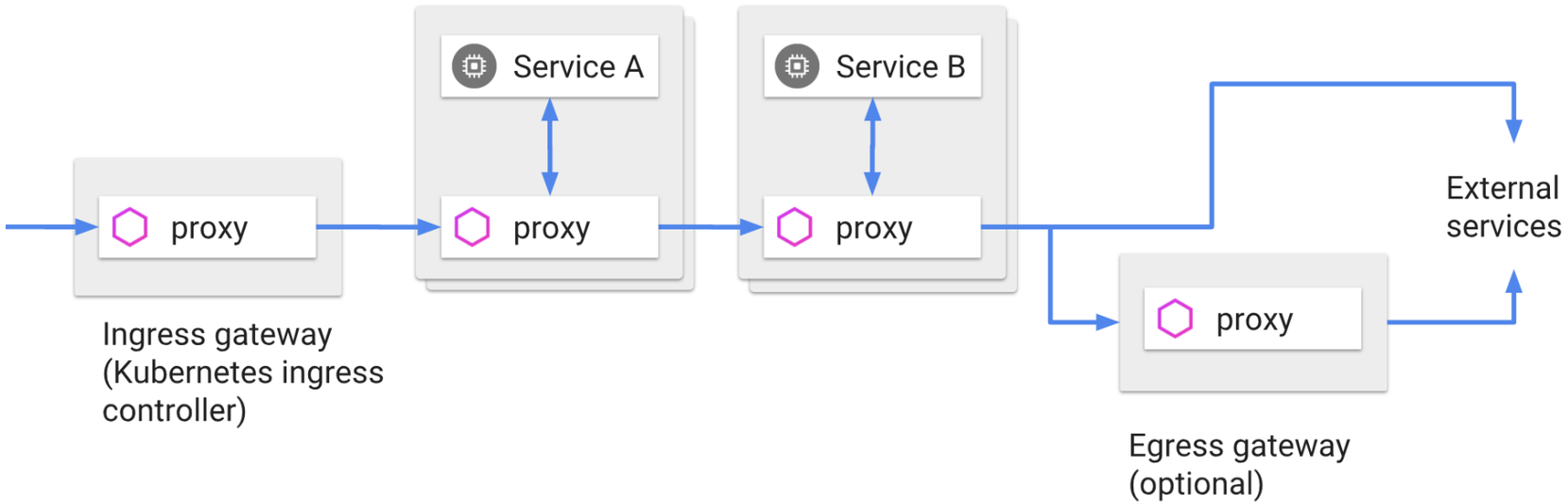
```
istioctl kube-inject -f deployment.yaml | \
    kubectl apply -f -
```

Automatic

```
apiVersion: admissionregistration.k8s.io/v1
kind: MutatingWebhookConfiguration
metadata:
  name: istio-sidecar-injector
webhooks:
- rules:
  - apiGroups:
    - ""
    apiVersions:
    - v1
    operations:
    - CREATE
    resources:
    - pods
    scope: '*'
...
```

Data Plane

Envoy



Workshop: Deploy GKE Cluster and Install Istio

IstioOperator API

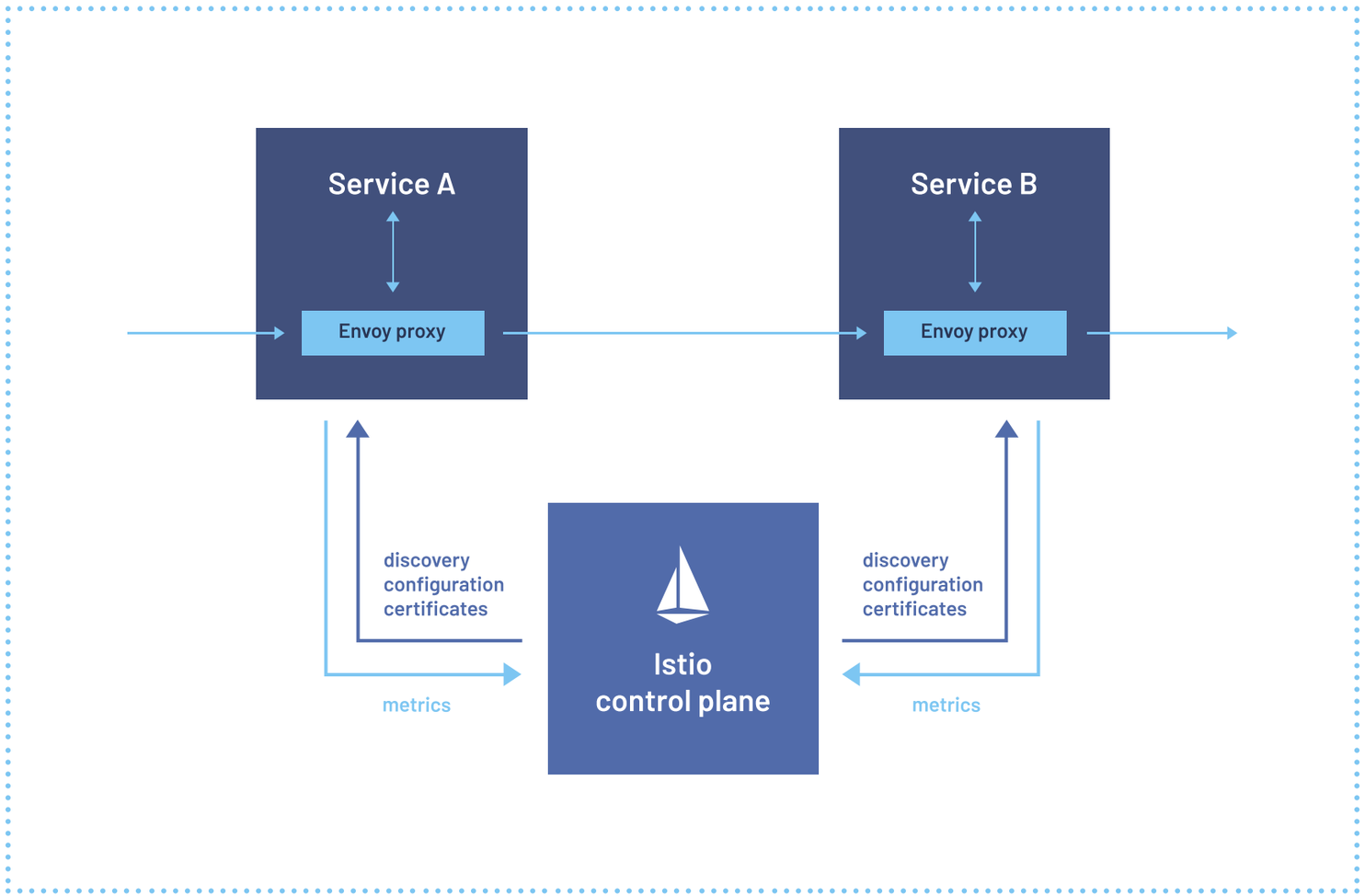
```
apiVersion: install.istio.io/v1alpha1
kind: IstioOperator
spec:
  profile: default
```

```
istioctl install -f istio-operator.yaml
```

Istio Sidecar Deep Dive

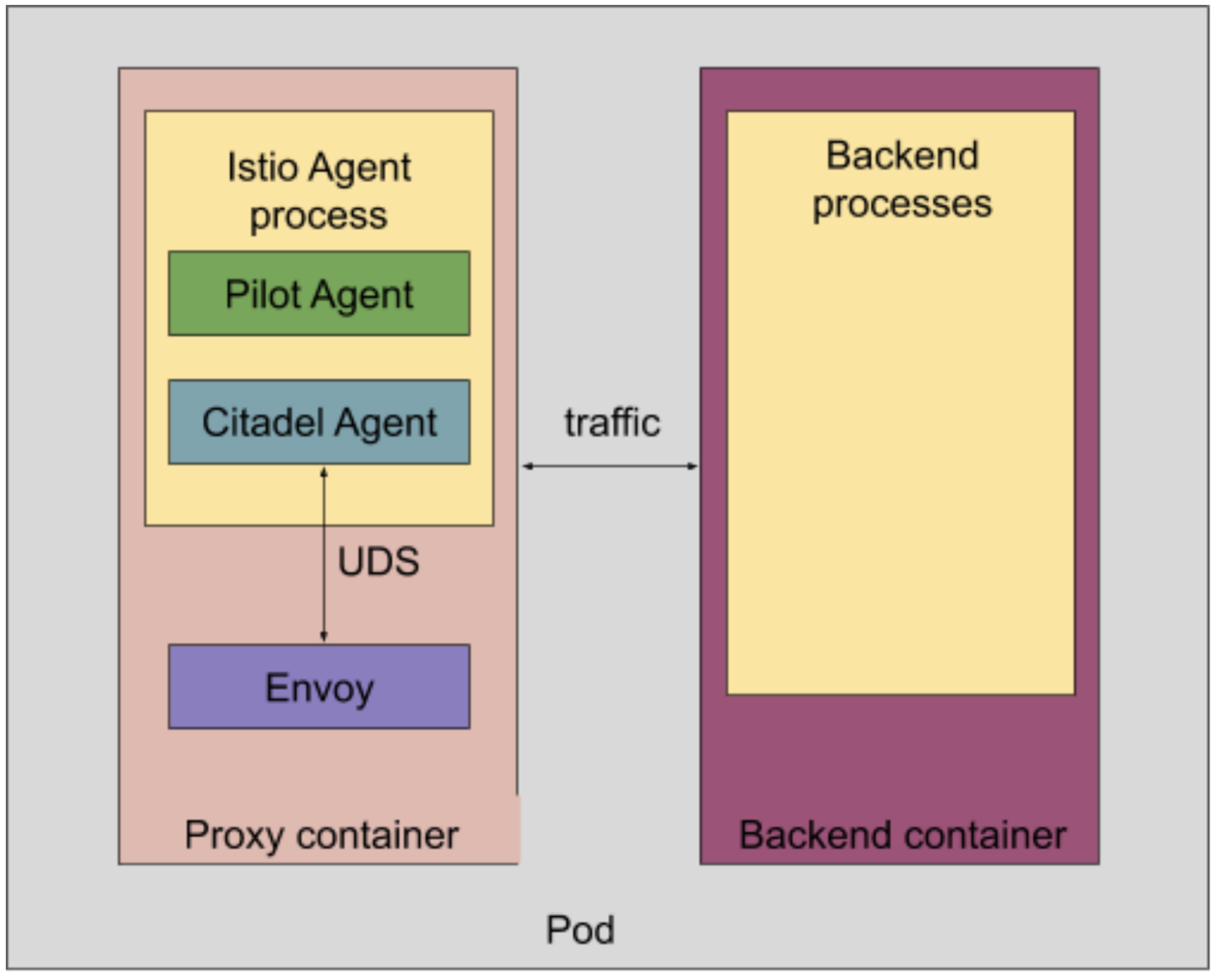
The proxy that powers the Istio data plane

Envoy Proxy

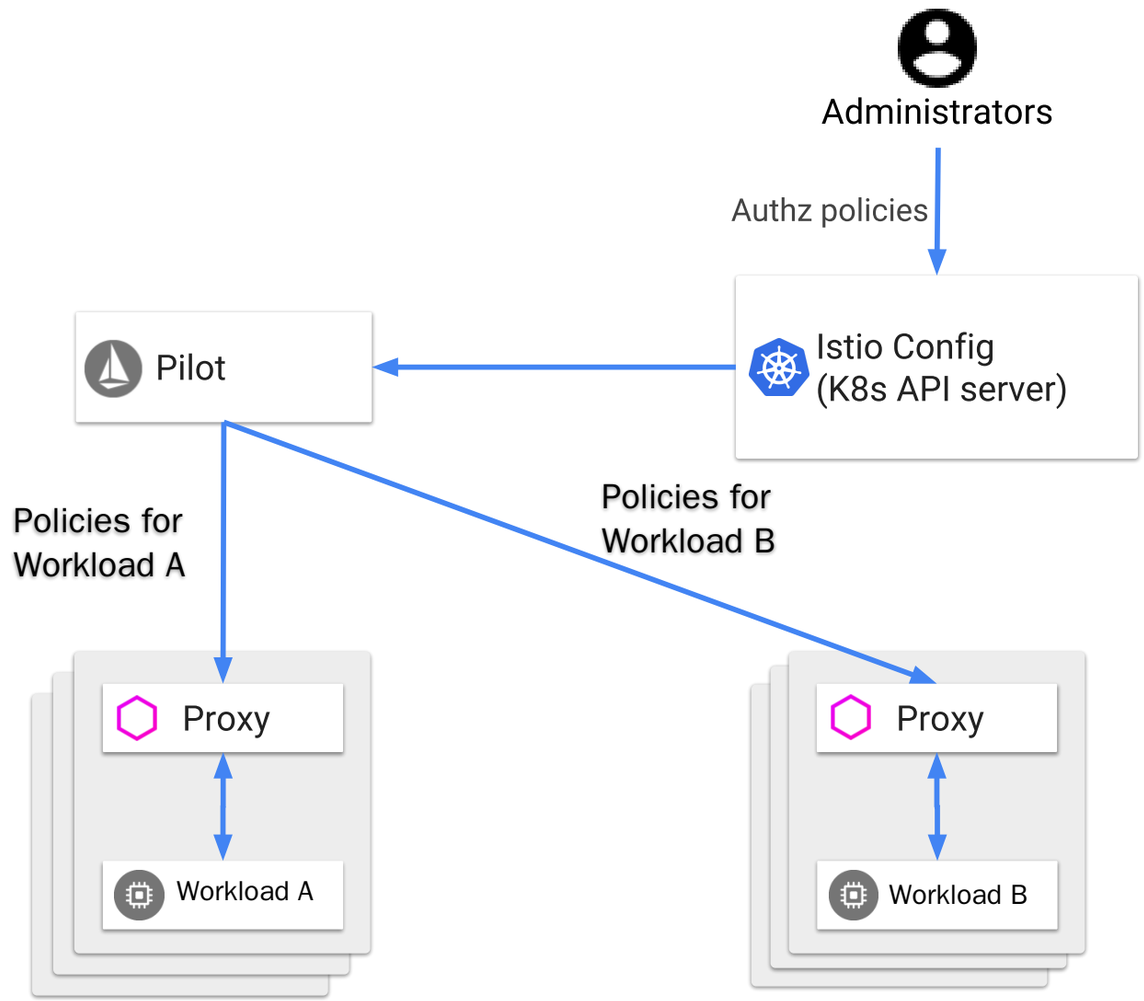


https://istio.io/latest/img/service-mesh.svg

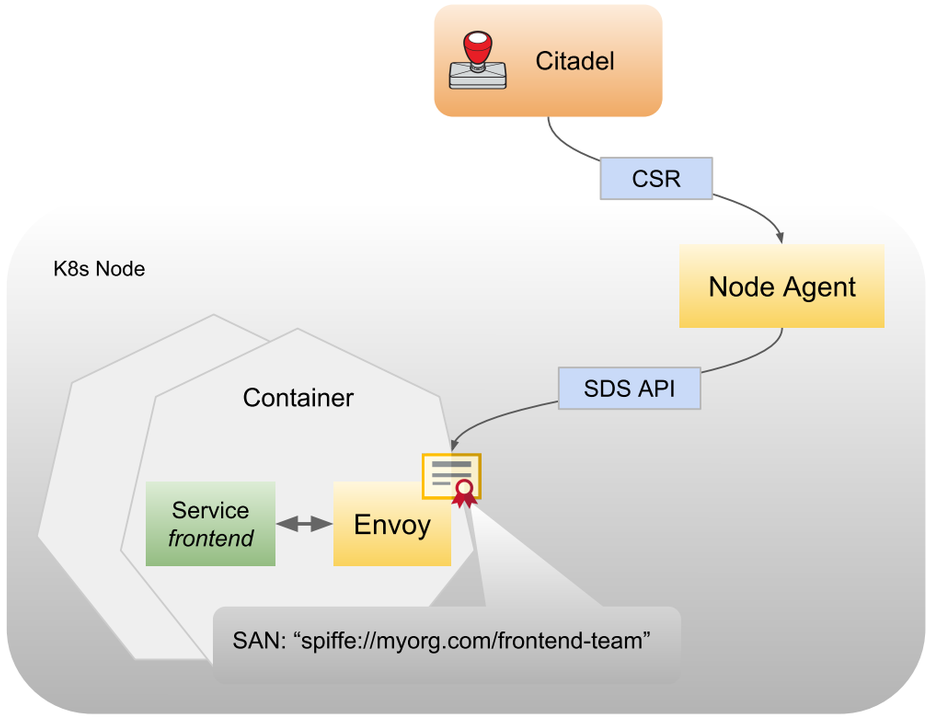
Istio Agent


Istio Agent process
Pilot Agent
Citadel Agent
UDS
Envoy
Proxy container
traffic
Backend processes
Backend container
Pod

Pilot


Administrators
Authz policies
Pilot
Istio Config
(K8s API server)
Policies for
Workload A
Policies for
Workload B
Proxy
Workload A
Proxy
Workload B

SDS (Secret Discovery Service) `1.4.x`



SDS (Secret Discovery Service) 1.5.x

```
$ ls -l /etc/istio/proxy/SDS
srw-rw-rw- 1 istio-proxy istio-proxy 0 Apr 13 19:58 /etc/istio/proxy/SDS
```

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
  ...
spec:
  containers:
  - name: istio-proxy
    image: docker.io/istio/proxyv2:1.5.1
    args:
    - proxy
    - sidecar
    - --configPath
    - /etc/istio/proxy
    ...
    volumeMounts:
    - name: istio-envoy
      mountPath: /etc/istio/proxy
    ...
  volumes:
  - name: istio-envoy
    emptyDir:
      medium: Memory
  ...
```

PodSecurityPolicies

- Required by `istio-init` so that it's permitted to modify IPTables rules (unless using Istio CNI)
- ~~Required when using SDS to read the UNIX domain socket created by the node agent~~ Unnecessary in `1.5.x`

Envoy Sidecar Injection

```
kubectl get mutatingwebhookconfiguration istio-sidecar-injector -o yaml
```

```
...
name: sidecar-injector.istio.io
namespaceSelector:
  matchLabels:
    istio-injection: enabled
rules:
- apiGroups:
  - ""
  apiVersions:
  - v1
  operations:
  - CREATE
  resources:
  - pods
  scope: '*'
...
```

Envoy Sidecar Injection

```
kubectl label namespace default istio-injection=enabled --overwrite
```

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
  ...
spec:
  initContainers:
  - name: istio-init
    image: docker.io/istio/proxyv2:1.5.1
    command:
    - istio-iptables
    ...
containers:
- name: nginx
  image: nginx
  ...
- name: istio-proxy
  image: docker.io/istio/proxyv2:1.5.1
  args:
  - proxy
  ...
...
```

Workshop - Envoy

Solution

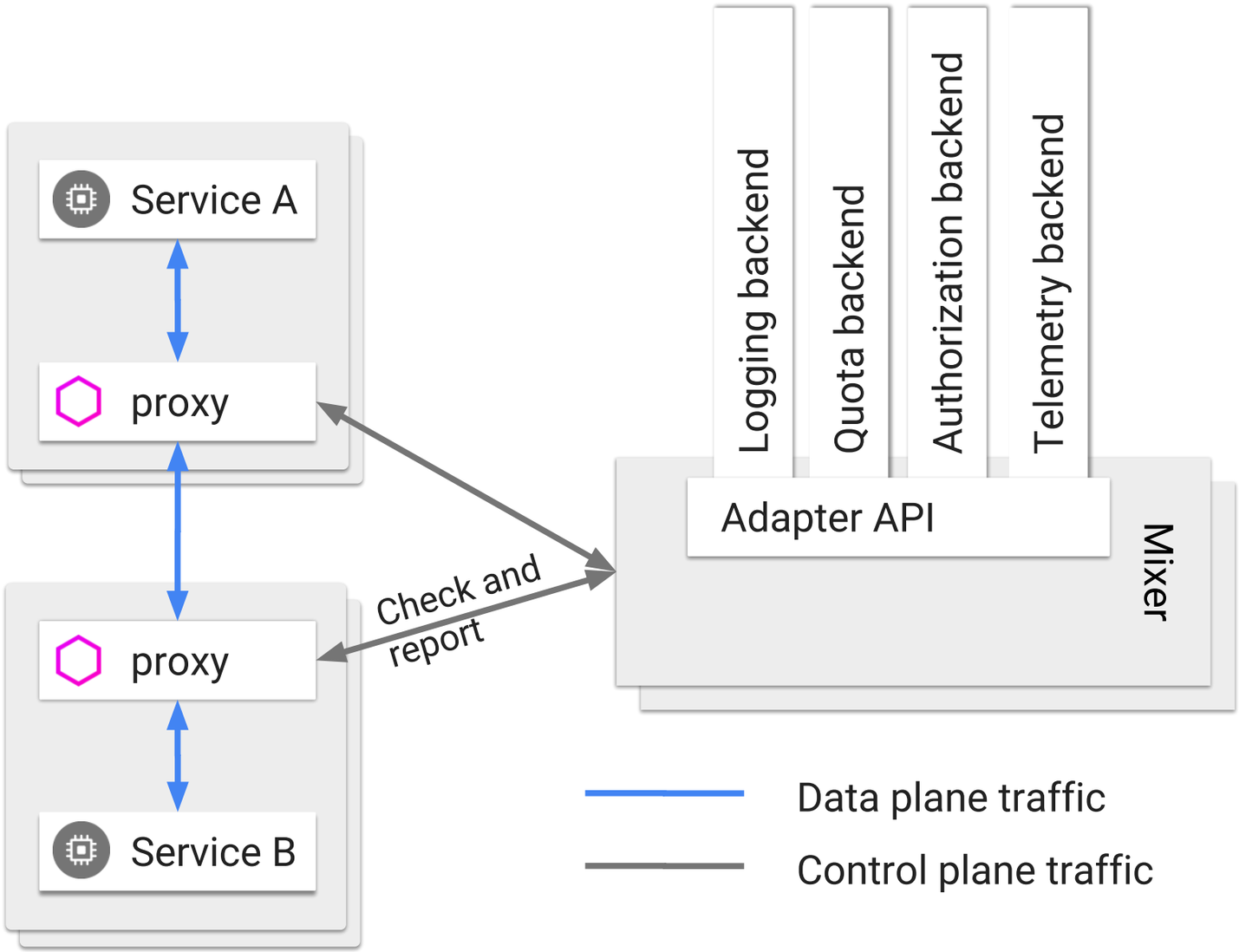
```
kubectl label namespace kube-system name=kube-system
kubectl label namespace istio-system name=istio-system
```

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: productpage
  namespace: netpol
spec:
  podSelector:
    matchLabels:
      app: productpage
policyTypes:
- Ingress
- Egress
ingress:
- from:
  - namespaceSelector:
      matchLabels:
        name: istio-system
    podSelector:
      matchLabels:
        app: istio-ingressgateway
  ports:
  - protocol: TCP
    port: 9080
- from:
  - namespaceSelector:
      matchLabels:
```

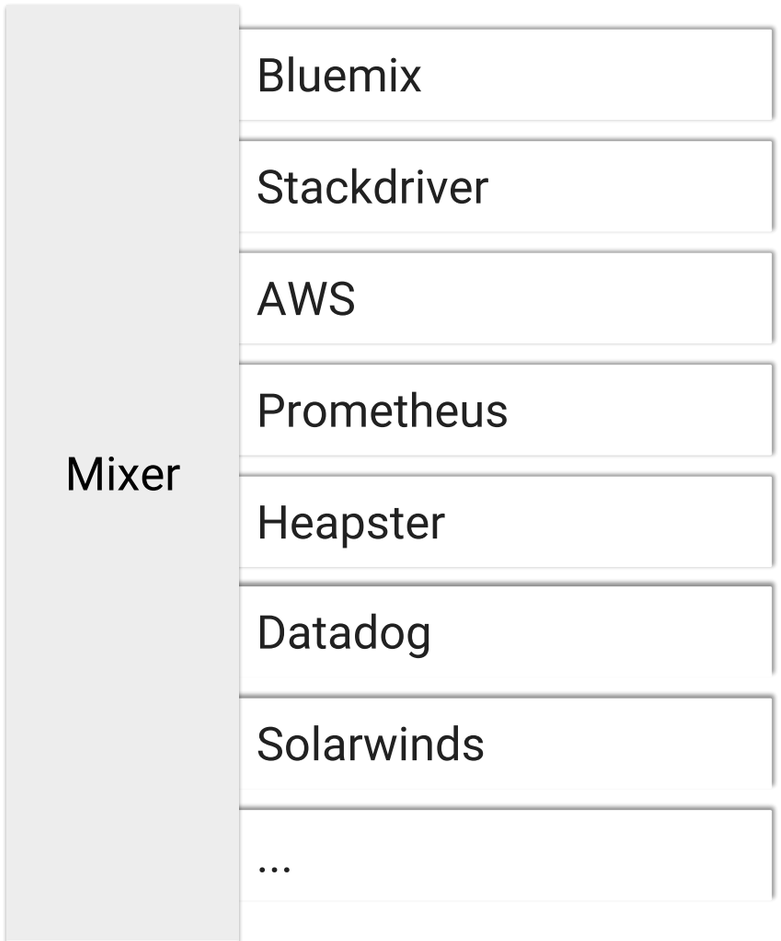
Mixer Deep Dive (deprecated)

Policy and telemetry

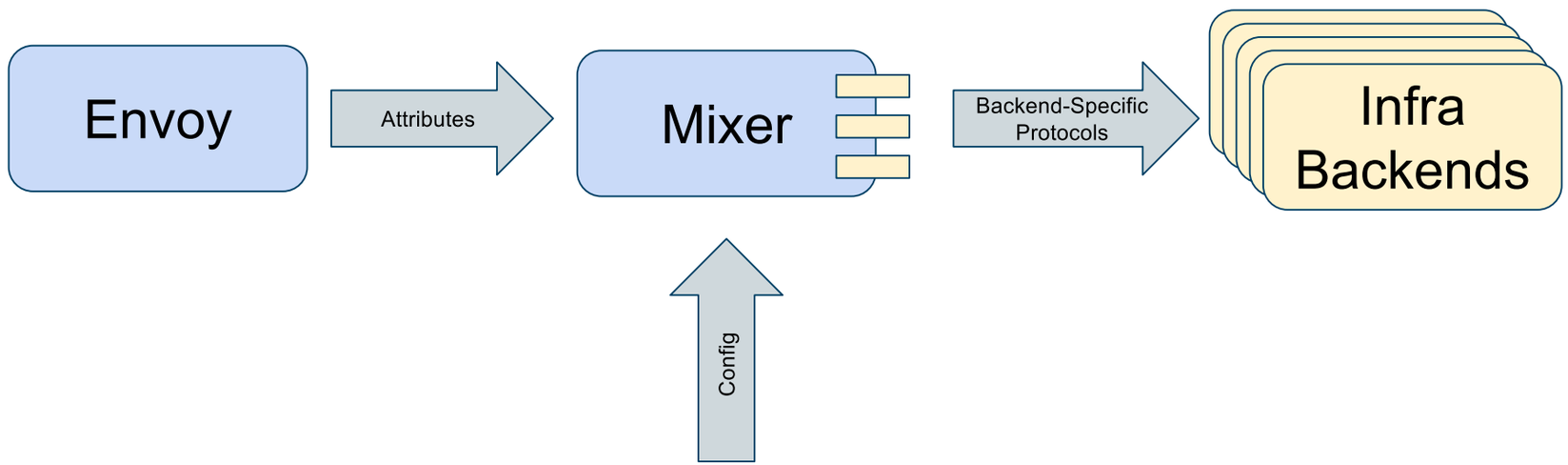
Mixer topology



Mixer and its Adapters



Attribute Machine



Operator

Workshop - Mixer

Solution

```
# Configuration for metric instance
apiVersion: config.istio.io/v1alpha2
kind: instance
metadata:
  name: request-count
  namespace: istio-system
spec:
  compiledTemplate: metric
  params:
    value: "1"
    dimensions:
      source_version: source.labels["version"] | "unknown"
      destination_service: destination.service.host | "unknown"
      destination_version: destination.labels["version"] | "unknown"
      response_code: response.code | 200
    monitored_resource_type: '"UNSPECIFIED"'
```

```
# Configuration for a Prometheus handler
apiVersion: config.istio.io/v1alpha2
kind: handler
metadata:
  name: prometheus-request-count
  namespace: istio-system
spec:
  compiledAdapter: prometheus
  params:
    metrics:
    - name: request_count # Prometheus metric name
      instance_name: request-count.instance.istio-system # Mixer instanc
      kind: COUNTER
      label_names:
      - source_version
      - destination_service
      - destination_version
      - response_code
```

```
# Rule to send metric instances to a Prometheus handler
apiVersion: config.istio.io/v1alpha2
kind: rule
metadata:
  name: request-count
  namespace: istio-system
spec:
  actions:
  - handler: prometheus-request-count
    instances: [ request-count ]
```

Extensibility

WebAssembly

WebAssembly is a sandboxing technology that defines a binary instruction format for a stack-based virtual machine.

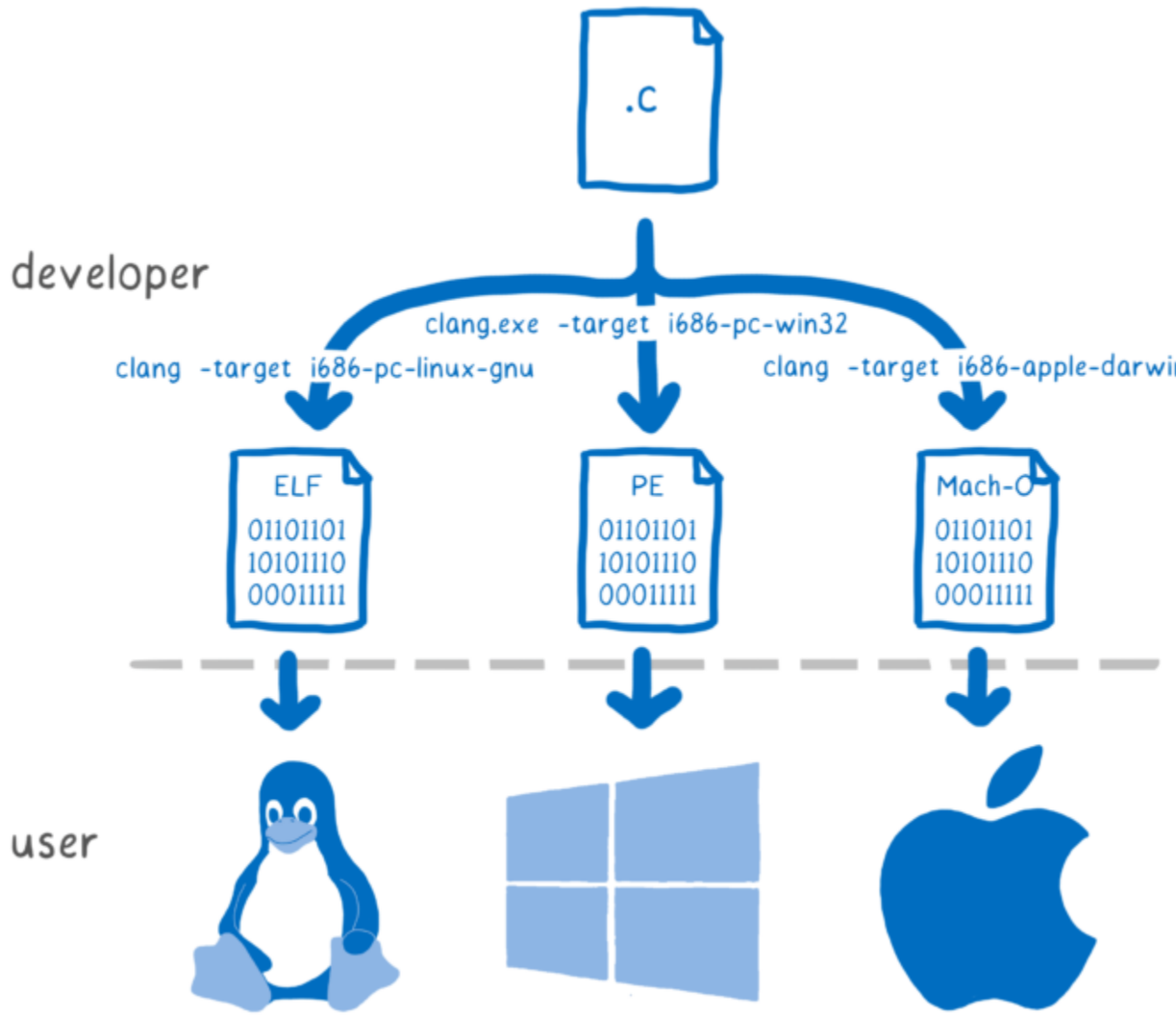
Host architecture specific runtimes can be used to run WebAssembly modules portably.

WebAssembly was originally designed to run high performance applications securely in the browser.

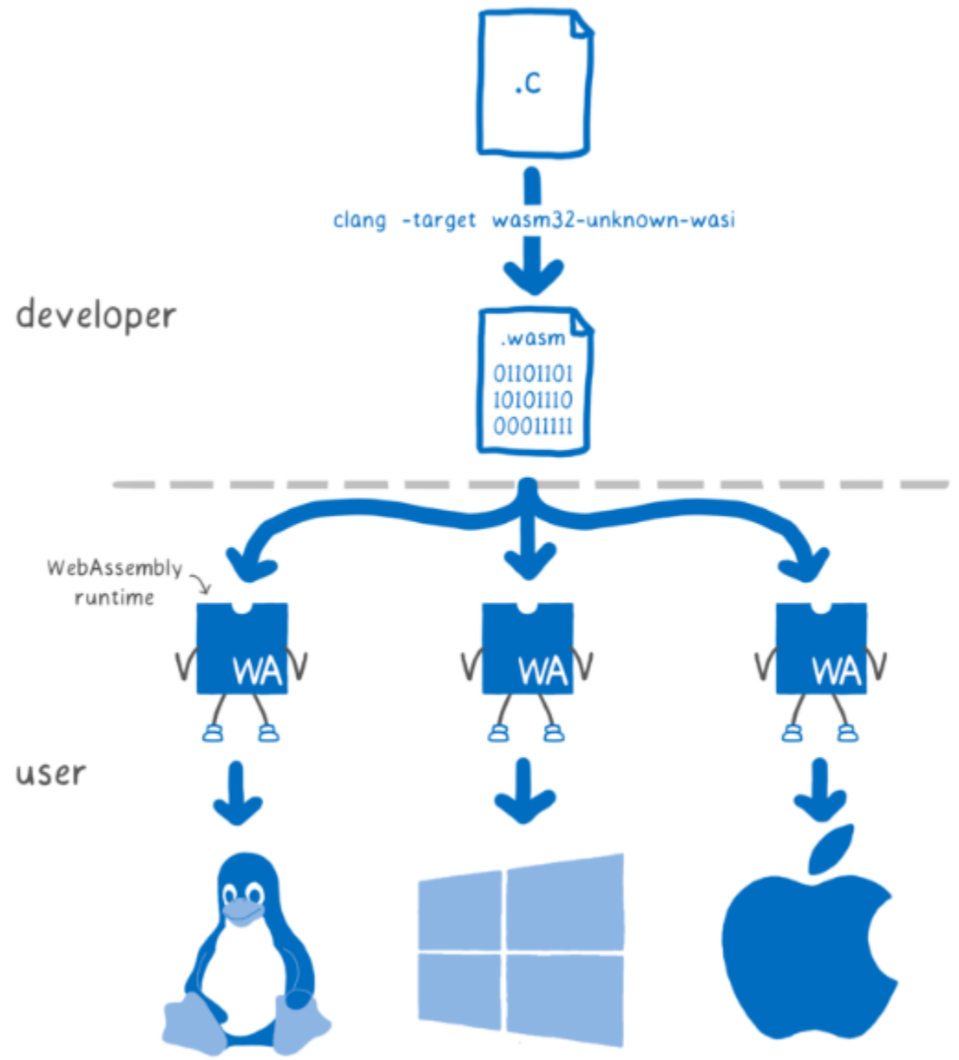
WebAssembly Goals:

- Fast, efficient, portable
- Readable and debuggable assembly language format
- Secure

WebAssembly



WebAssembly



WebAssembly System Interface (WASI)

Defines an ABI for WebAssembly modules to interact with their host environment:

- Files and filesystems
- Sockets
- Clocks
- Random numbers

Extensibility

Each Envoy listener (usually programmed through xDS or the Listener Discover Service (LDS) specifically) defines a set of filters that sit in the data path, collectively forming a chain.

EnvoyFilter is an Istio resource that provides a mechanism for modifying these filters. Use cases include:

- Interpreting custom protocols
- Generating telemetry
- Performing RBAC

Extensibility

```yaml
apiVersion: networking.istio.io/v1alpha3
kind: EnvoyFilter
metadata:
  name: reviews-lua
  namespace: bookinfo
spec:
  workloadSelector:
    labels:
      app: reviews
  configPatches:
# The first patch adds the lua filter to the listener/http connection
- applyTo: HTTP_FILTER
  match:
    context: SIDECAR_INBOUND
    listener:
      portNumber: 8080
      filterChain:
        filter:
          name: "envoy.http_connection_manager"
          subFilter:
            name: "envoy.router"
  patch:
    operation: INSERT_BEFORE
    value: # lua filter specification
      name: envoy.lua
      typed_config:
        "@type": "type.googleapis.com/envoy.config.filter.http.lua.v2.
        inlineCode: |
        ...
```

Extensibility

Envoy supports a number of in-built filters, including `envoy.filters.http.wasm` which supports running WASM modules. This filter can be used to dynamically extend Envoy.

Proxy-Wasm is a proxy specific ABI allowing these modules to interact with their host proxy. This replaces Mixer as the primary extension mechanism in Istio.

Istio 1.6 will provide a uniform configuration API for Proxy-Wasm plugins.

Proxy-Wasm will eventually become part of WASI.

Extensibility

WebAssembly sandbox goals:

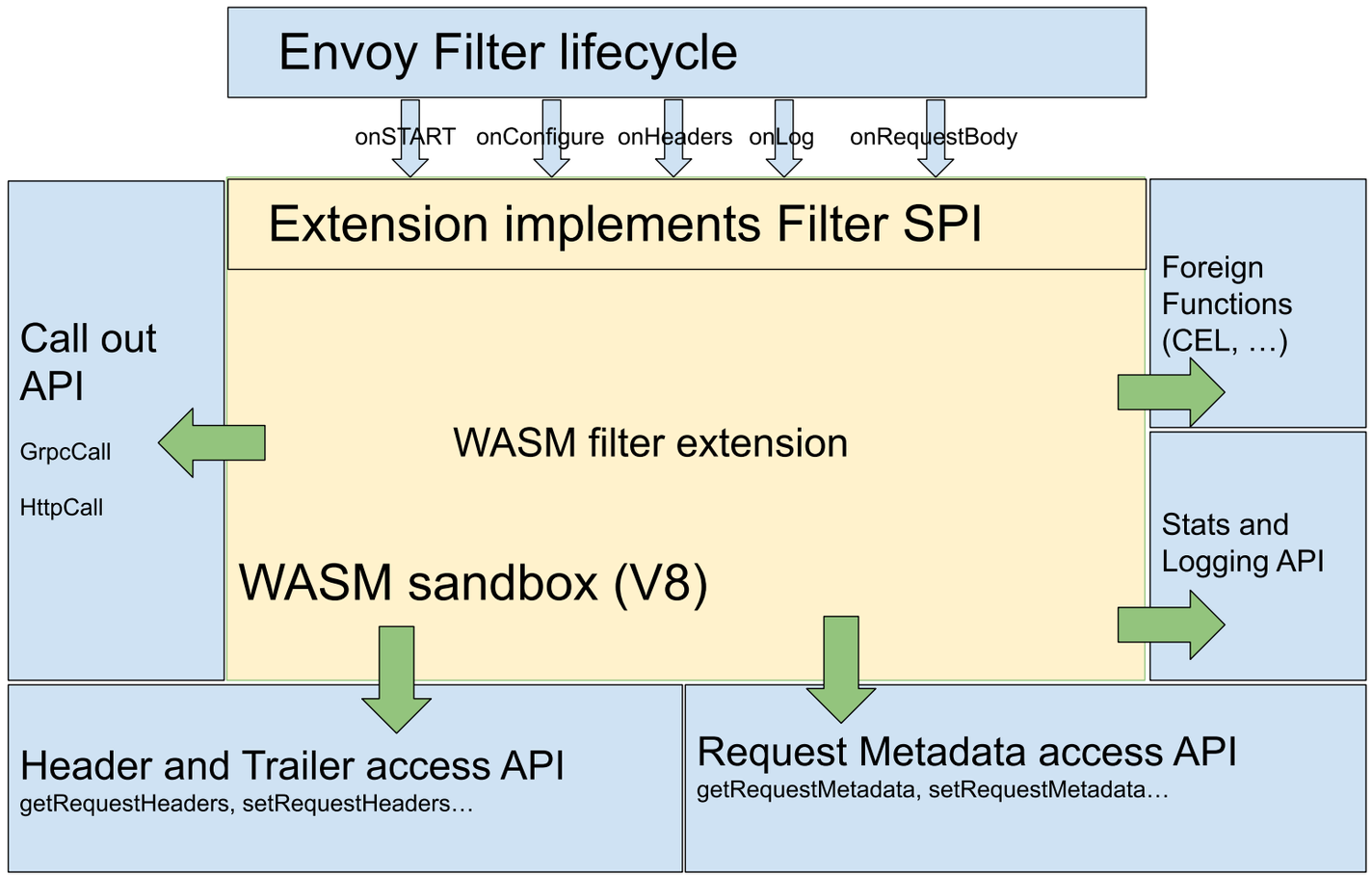
- **Efficiency** - An extension adds low latency, CPU, and memory overhead.
- **Function** - An extension can enforce policy, collect telemetry, and perform payload mutations.
- **Isolation** - A programming error or crash in one plugin does affect other plugins.
- **Configuration** - The plugins are configured using an API that is consistent with other Istio APIs. An extension can be configured dynamically.
- **Operator** - An extension can be canaried and deployed as log-only, fail-open or fail-close.
- **Extension developer** - The plugin can be written in several programming languages.

High-level Architecture

Istio extensions (Proxy-Wasm plugins) have several components:

- **Filter Service Provider Interface (SPI)** for building Proxy-Wasm plugins for filters.
- **Sandbox** V8 Wasm Runtime embedded in Envoy.
- **Host APIs** for headers, trailers and metadata.
- **Call out APIs** for gRPC and HTTP calls.
- **Stats and Logging APIs** for metrics and monitoring.

High-level Architecture



Fault Injection

Fault Injection

Fault injection is a form of chaos experimentation that introduces errors into a system to ensure that it can withstand and recover from error conditions.

This process can be particularly useful to ensure that your failure recovery policies aren’t incompatible or too restrictive, potentially resulting in critical service unavailability.

Fault Injection

You can inject two types of faults, both configured using a `VirtualService`

Delays

Delays are timing failures. They mimic increased network latency or an overloaded upstream service.

Aborts

Aborts are crash failures. They mimic failures in upstream services. Aborts usually manifest in the form of HTTP error codes or TCP connection failures.

Fault Injection

```yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: ratings
spec:
  hosts:
  - ratings
  http:
  - fault:
      delay:
        percentage:
          value: 0.1
        fixedDelay: 5s
    route:
    - destination:
        host: ratings
        subset: v1
```

Fault Injection

```yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: ratings
spec:
  hosts:
  - ratings
  http:
  - fault:
      abort:
        httpStatus: 500
        percentage:
          value: 10
    route:
    - destination:
        host: ratings
        subset: v1
```

Workshop - Fault Injection

Circuit Breaking

Circuit Breaking

Circuit breakers are another useful mechanism Istio provides for creating resilient microservice-based applications.

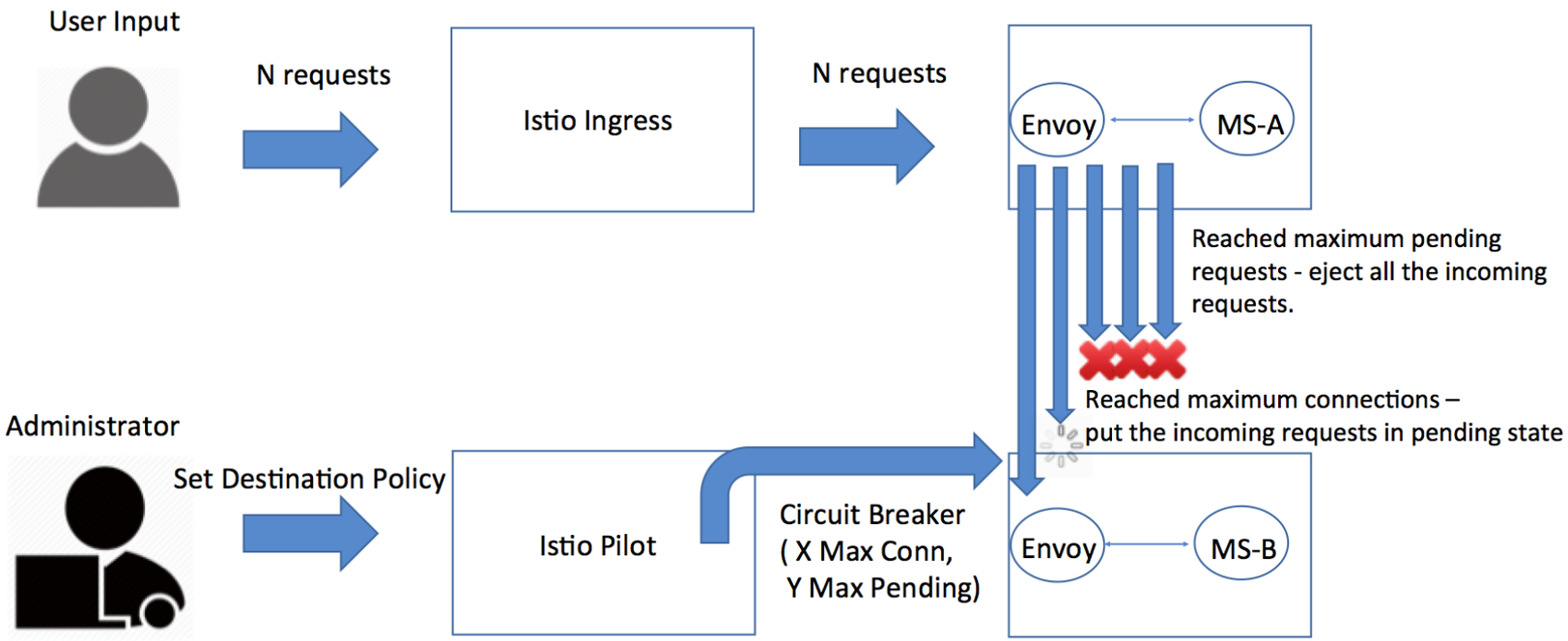
A circuit breaker pattern ensures fast failure rather than having clients try to connect to an overloaded or failing host.

Circuit Breaking

The following example limits the number of concurrent connections for the reviews service workloads of the `v1` subset to 100:

```
apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: reviews
spec:
  host: reviews
  subsets:
  - name: v1
    labels:
      version: v1
    trafficPolicy:
      connectionPool:
        tcp:
          maxConnections: 100
```

Circuit Breaking



https://developer.ibm.com/code/wp-content/uploads/sites/118/2017/08/Screen-Shot-2017-08-22-at-12.50.46-AM-768x342.png

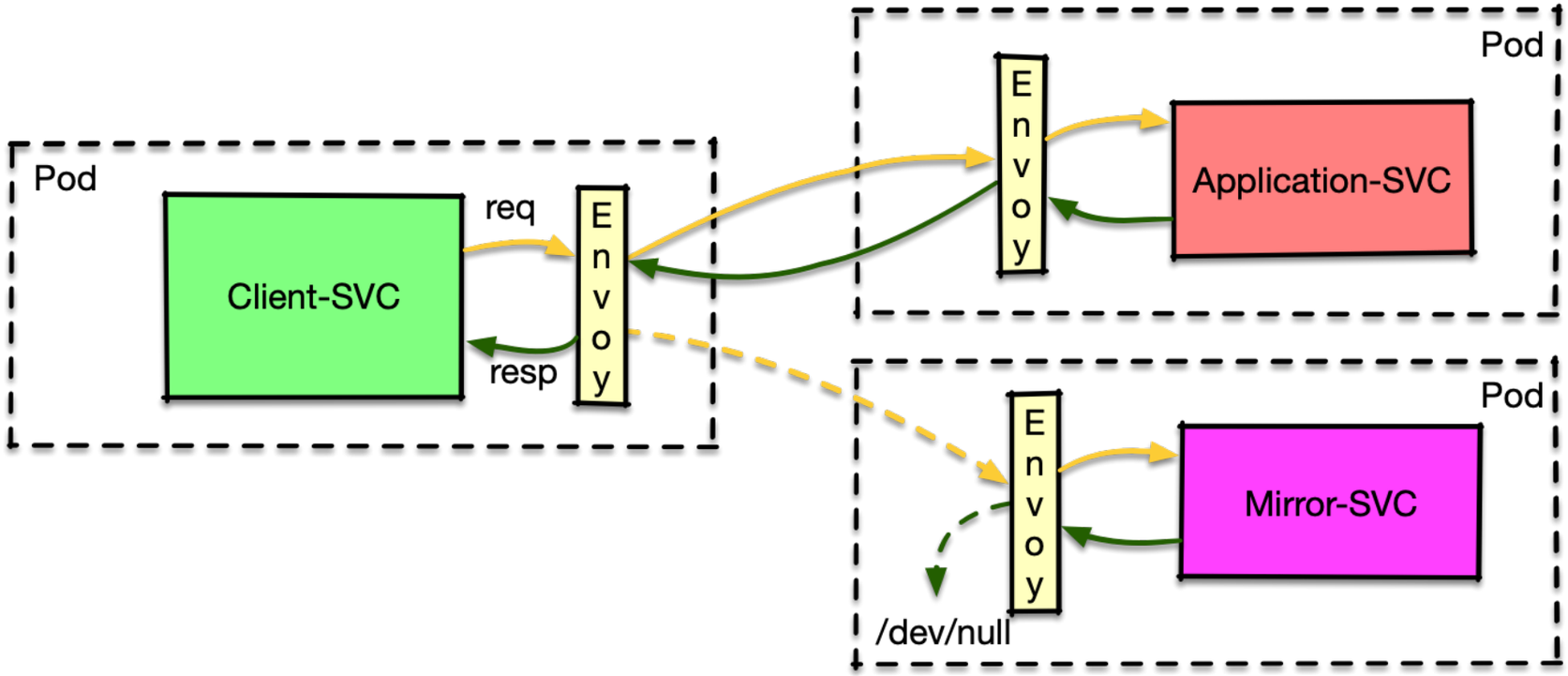
Workshop - Circuit Breaking

Traffic Mirroring

Traffic Mirroring

Mirroring sends a copy of live traffic to a mirrored service. The mirrored traffic happens out of band of the critical request path for the primary service.

Traffic Mirroring



https://miro.medium.com/max/1438/1*9_3WRLFmo3dcOvVMmdbkVw.png

Traffic Mirroring

Mirroring traffic can be achieved using a `VirtualService`:

```yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: httpbin
spec:
  hosts:
  - httpbin
  http:
  - route:
    - destination:
        host: httpbin
        subset: v1
      weight: 100
    mirror:
      host: httpbin
      subset: v2
    mirror_percentage:
      value: 100
```

Workshop - Traffic Mirroring

Deployment Models

Replicated control planes, federated meshes and more...

Deployment Tools

- Helm (deprecated)
- `istioctl`
- Operator (experimental)
- Managed add-ons

`istioctl` Install

```
istioctl install --set profile=demo
```

	default	demo	minimal	external	empty	preview
Core components						
`istio-egressgateway`		✔				
`istio-ingressgateway`	✔	✔				✔
`istiod`	✔	✔	✔			✔

IstioOperator Resource

An `IstioOperator` resource describes an Istio deployment.

```yaml
apiVersion: install.istio.io/v1alpha1
kind: IstioOperator
metadata:
  name: example-istiocontrolplane
  namespace: istio-system
spec:
  profile: demo
```

IstioOperator Resource

Can be used to customise Istio components.

```yaml
apiVersion: install.istio.io/v1alpha1
kind: IstioOperator
metadata:
  name: istio
spec:
  components:
    ingressGateways:
    - enabled: true
      name: istio-ingressgateway
      k8s:
        service:
          type: ClusterIP
        serviceAnnotations:
          # stand-alone NEG annotation
          cloud.google.com/neg: '{"exposed_ports":{"443":{},"80":{}}}'
```

`istioctl` Install With IstioOperator

```
istioctl install --filename istio_operator.yaml
```

`istioctl` Manifest

Show generated manifests.

```
istioctl manifest generate --set profile=demo > istio_manifests.yaml
```

```
istioctl manifest generate --filename istio_operator.yaml > istio_manife
```

Istio Operator

Manage Istio deployment with an operator.

```
istioctl operator init
```

```
kubectl apply -f istio_operator.yaml
```

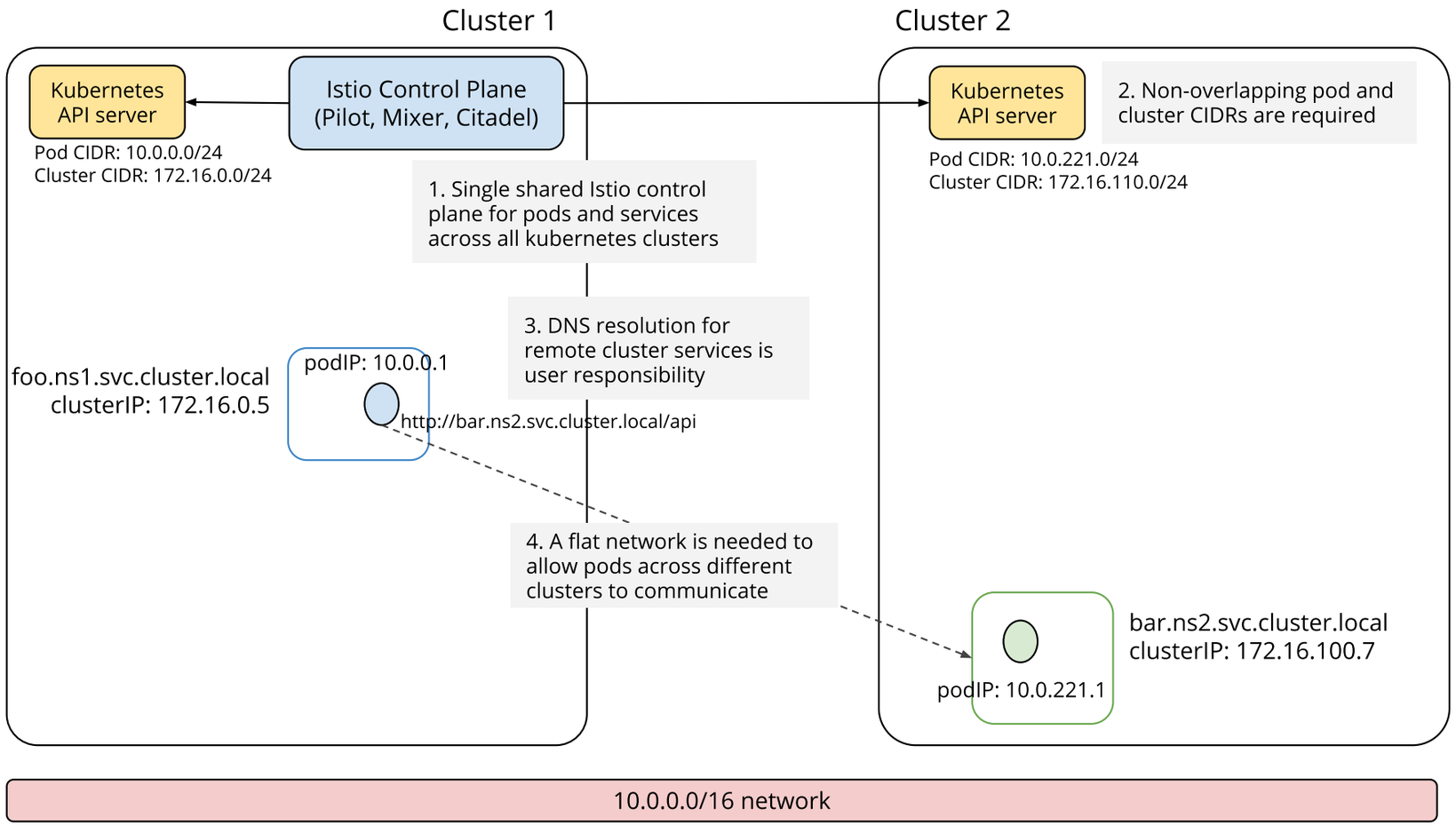
GKE

```
gcloud beta container clusters create ...
    --addons=Istio \
    --istio-config=auth=MTLS_PERMISSIVE
```

Deployment Models

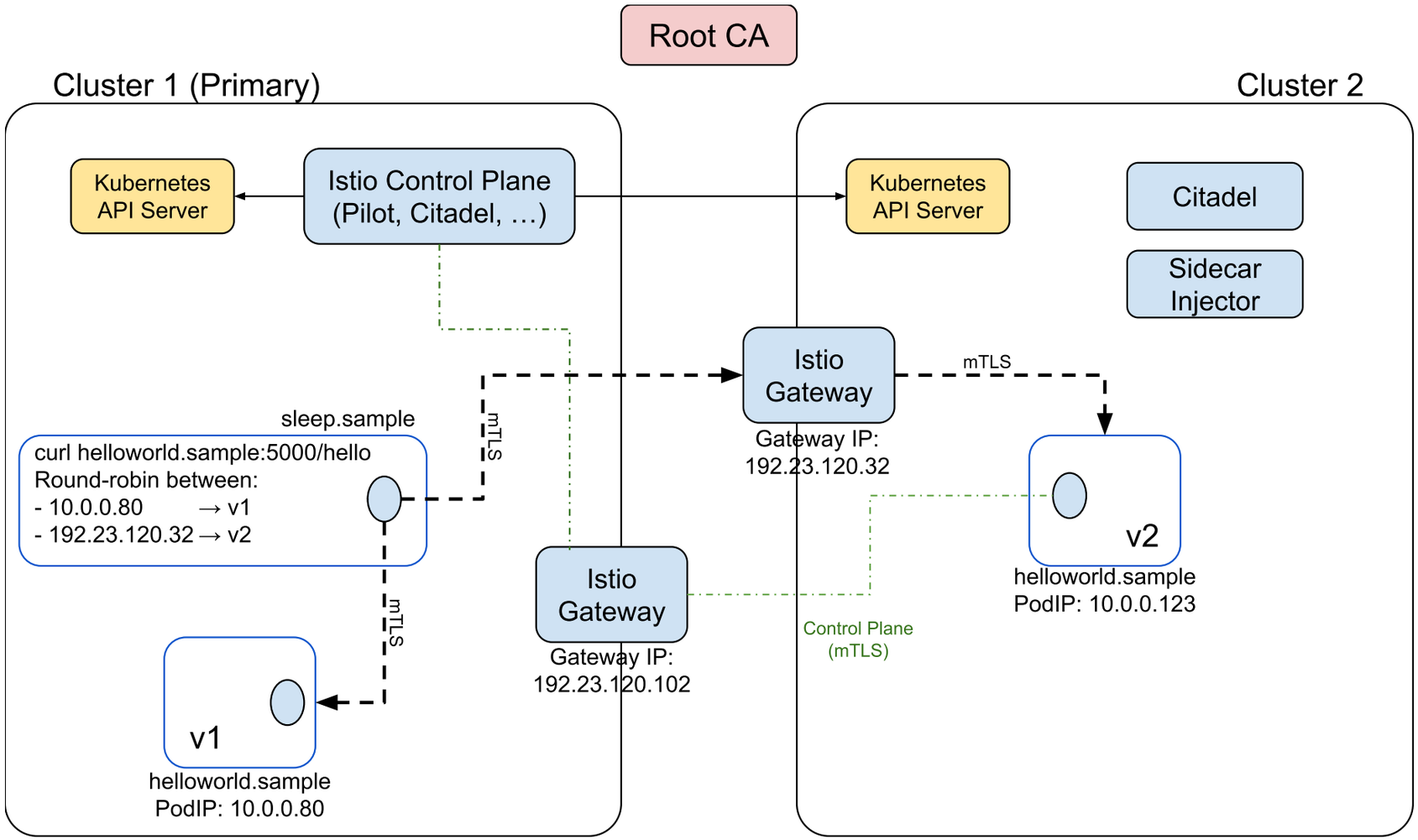
- Single or multiple cluster
- Single or multiple network
- Single or multiple control plane
- Single or multiple mesh

Multicluster, shared control plane, single network

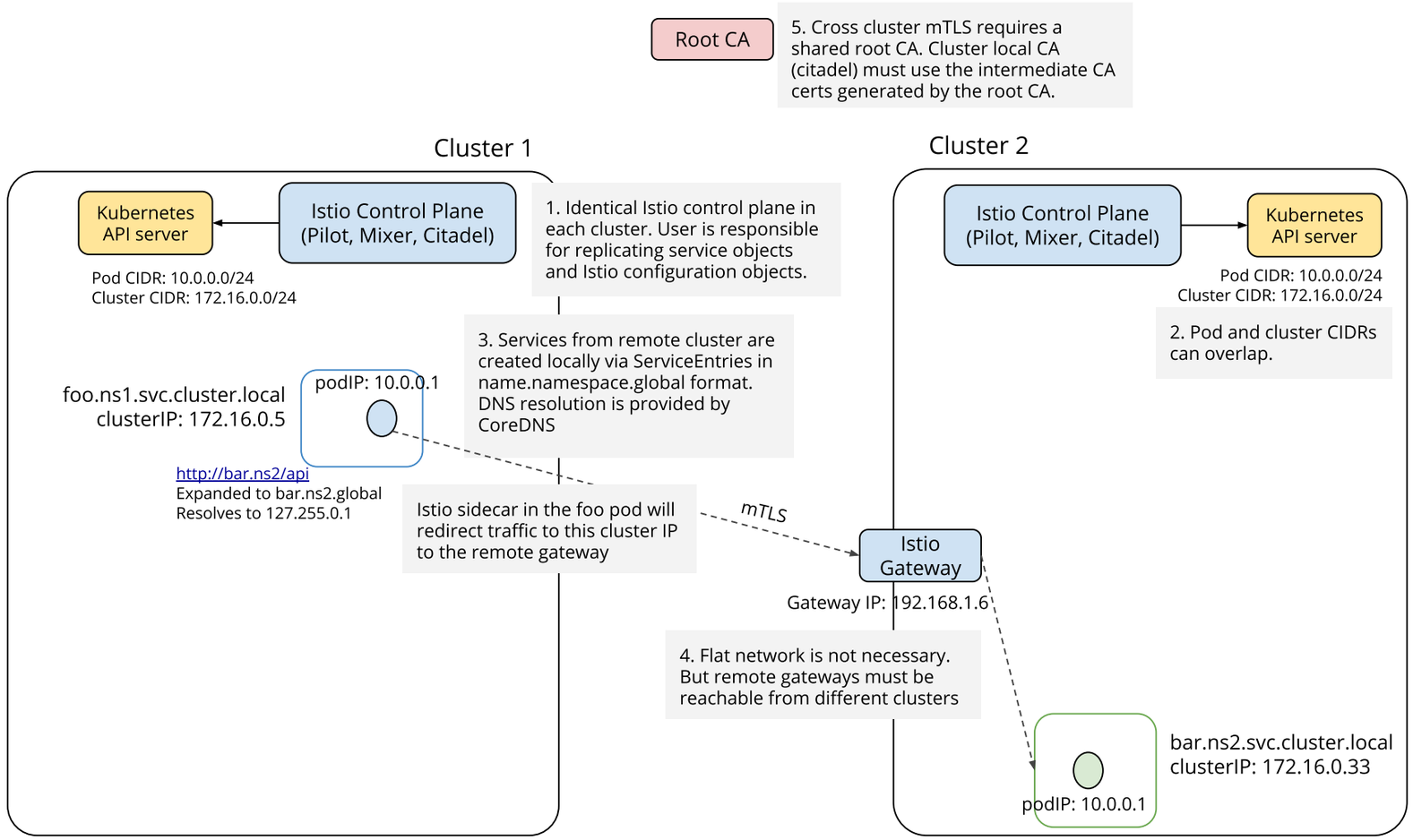


https://istio.io/docs/setup/install/multicluster/shared/

Multicluster, shared control plane, multiple networks



Multicluster, replicated control planes, multiple networks



Shared vs Multiple Control planes

	Shared Control plane	Multiple Control planes
	- Easy configuration - Centralized control	- Distributed / decentralized controlplane - No SPoF - Scalable
	- Single point of failure - Latency considerations - Scalability considerations	- Harder configuration - Larger footprint

Finally...

Training

- Kubernetes in Practice (Day One and Day Two)
- Kubernetes for Application Developers (Day One and Day Two)
- Istio in Practice (Day One and Day Two)
- Operational Wargaming
- Kubernetes from Scratch
- Extending Kubernetes with Operators
- Serverless on Kubernetes with Knative

Consulting

On-site / Remote / Project-based

Production Readiness Review

Cloud Native Discovery Workshop

Jetstack Subscription

Jetstack Subscription is designed to help you verify, support and continually optimise your Kubernetes production environment.

- Break Fix Support →
- Customer Reliability Engineering →
- Kubernetes Reference Architecture →
- Automated Cluster Compliance Checking →
- Online Training and Wargaming →

Register Interest

Jetstack is trusted by

monzo Google Cloud babylon Pearson

The End