



Istio: Weaving, Observing and Securing Microservices

LIN SUN

Senior Technical Staff Member, IBM



@linsun_unc



The Problem

modern distributed architecture



container based services
deployed into dynamic environments
composed via the network

The Problem

IT's shift to a modern distributed architecture has left enterprises unable to **connect, observe or secure or control** their services in a consistent way.

Service Mesh

A service mesh provides a **transparent and language-independent** network for connecting, observing, securing and controlling the connectivity between services.

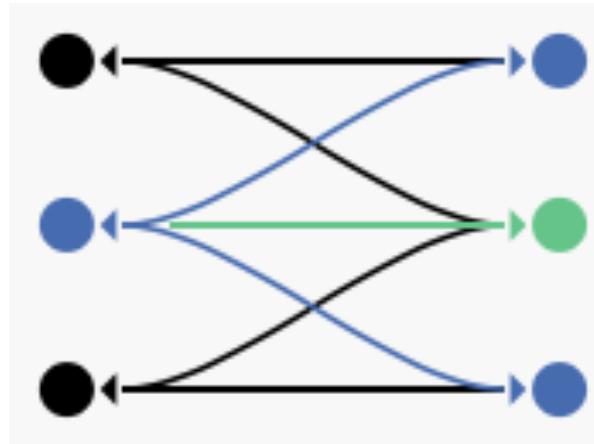


Istio

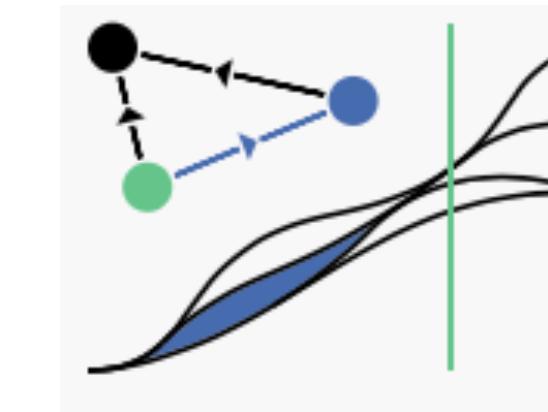
An **open service mesh platform to connect, observe, secure, and control microservices.**

Istio

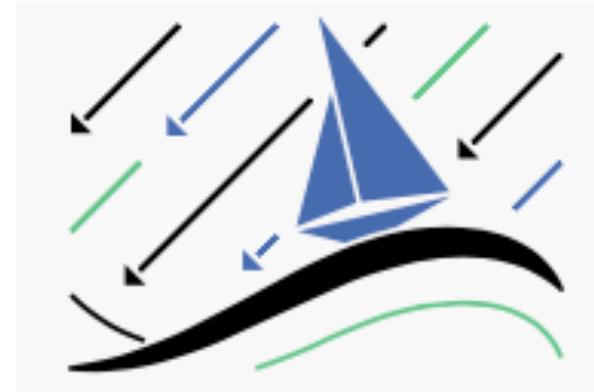
An open service mesh platform to connect, observe, secure, and control microservices.



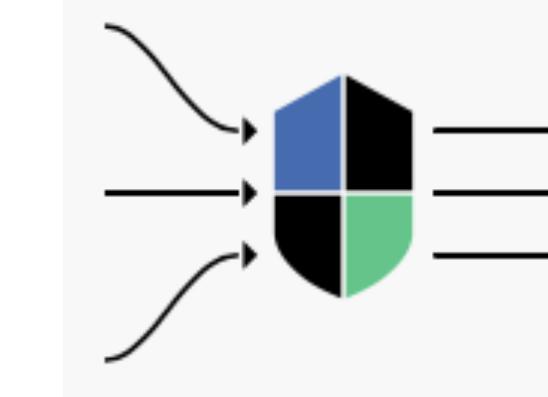
Connect: Traffic Control, Discovery,
Load Balancing, Resiliency



Observe: Metrics, Logging, Tracing

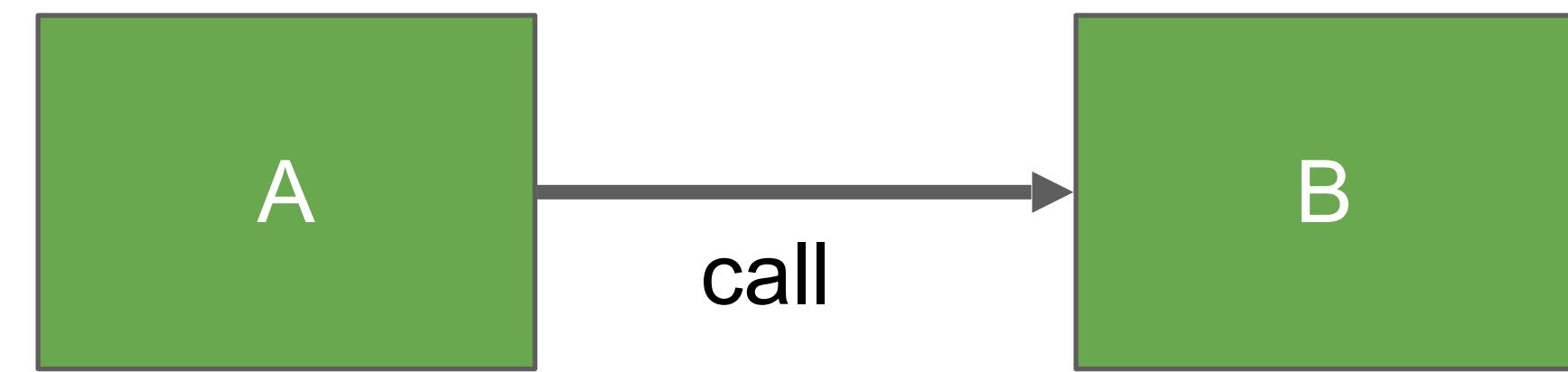


Secure: Encryption (TLS),
Authentication, and Authorization of
service-to-service communication



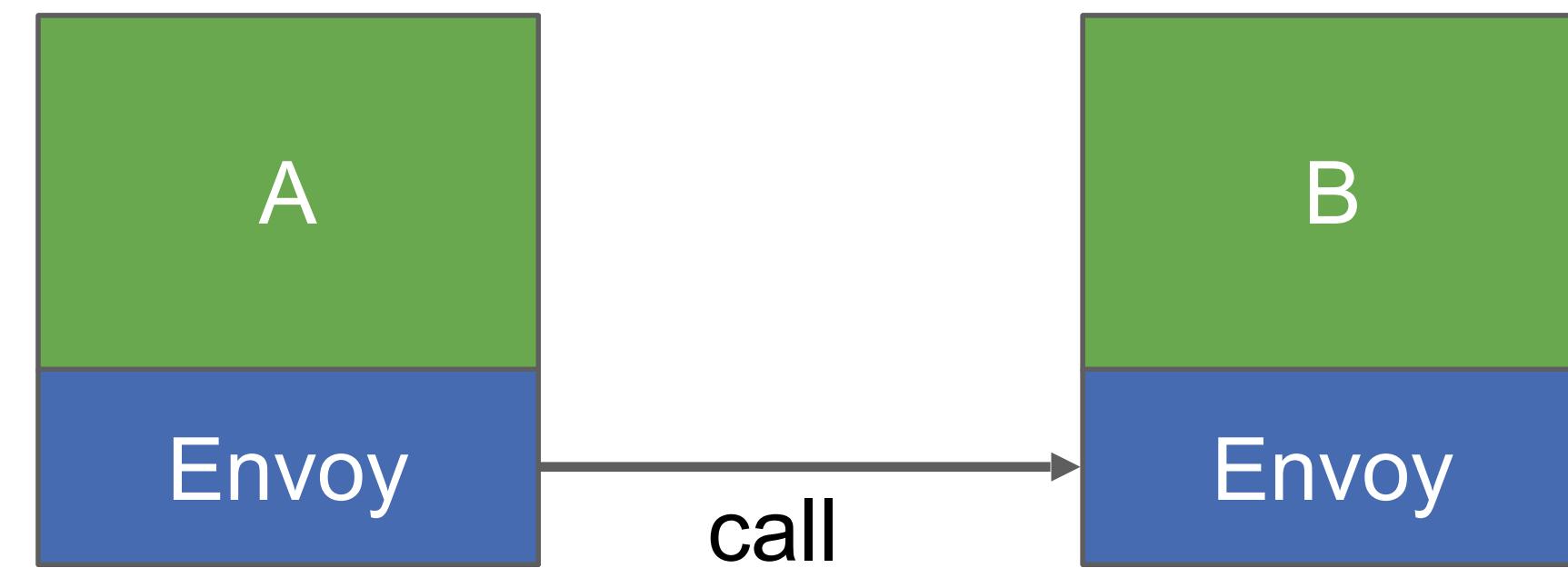
Control: Policy Enforcement

How does it work?



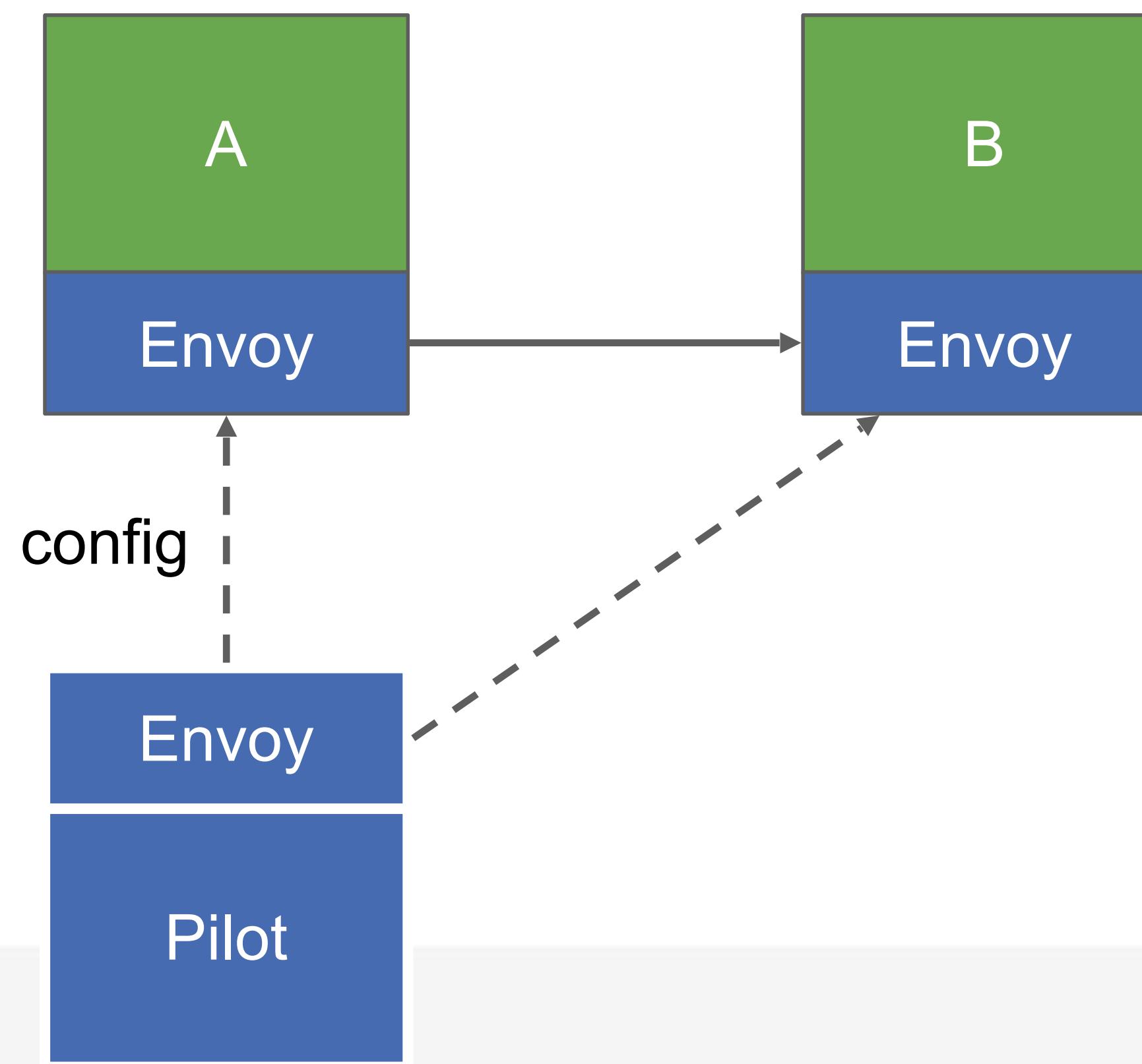
How does it work?

1. Deploy a proxy (Envoy) beside your application (“sidecar deployment”)



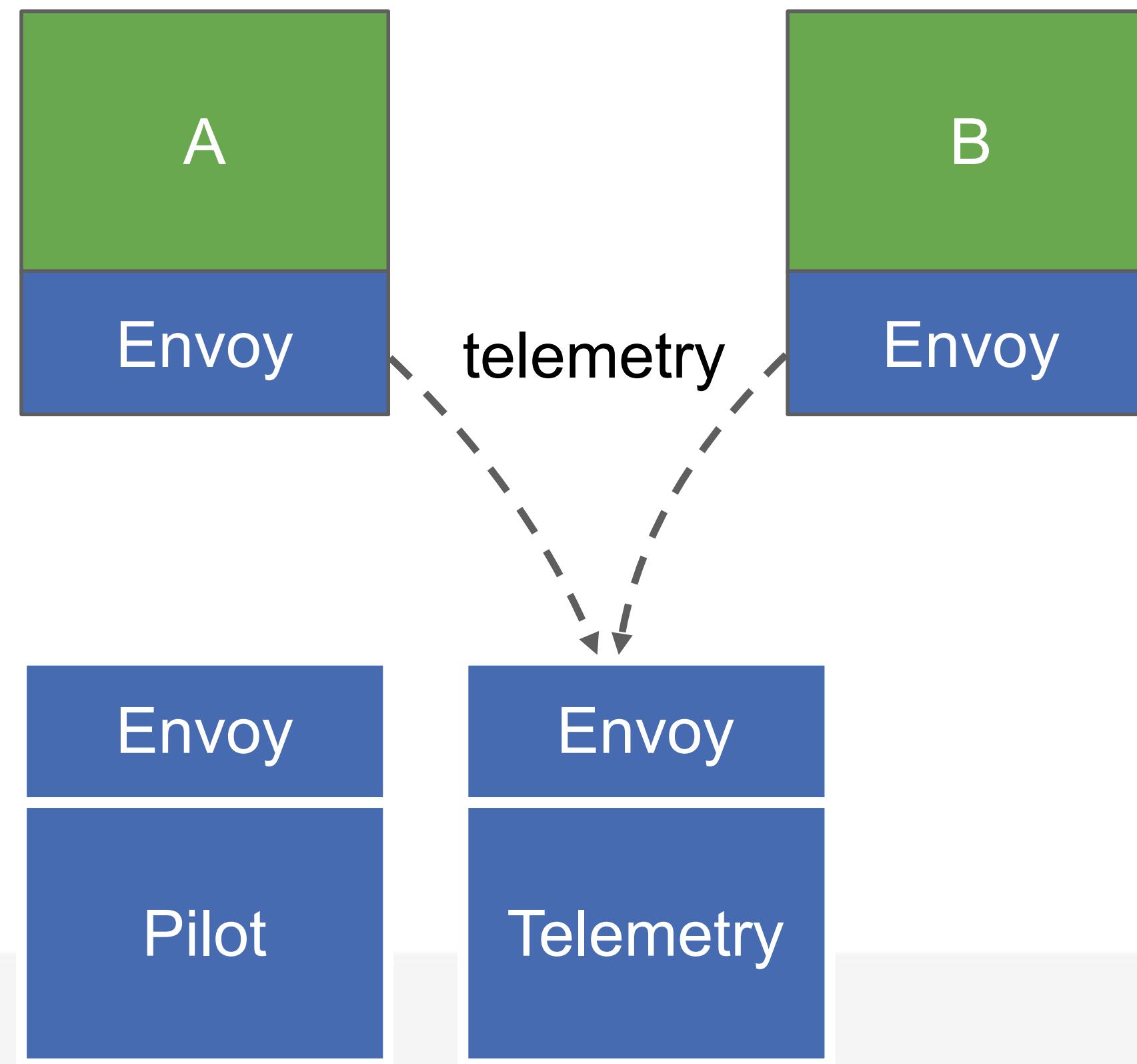
How does it work?

2. Deploy Pilot to configure the sidecars



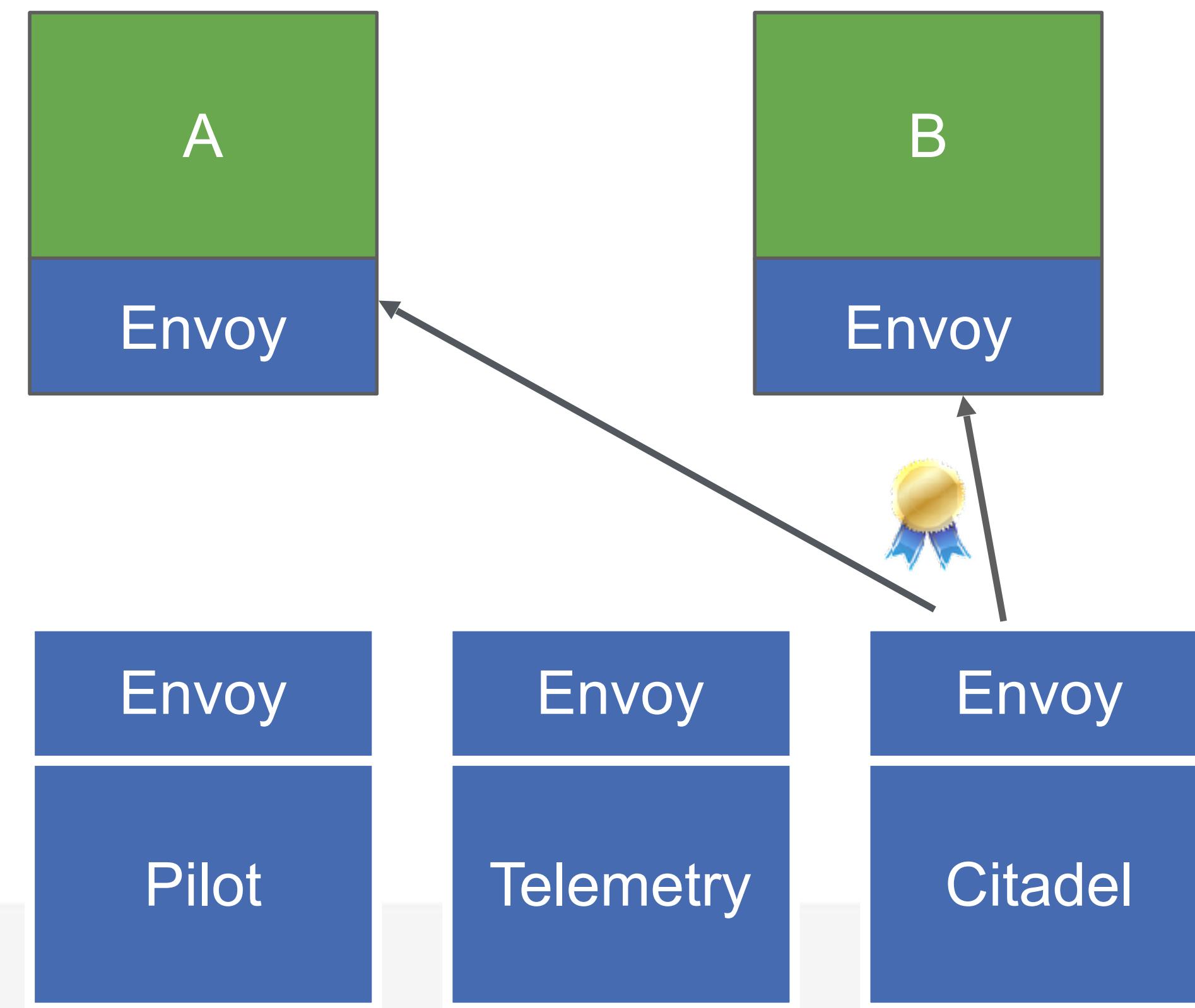
How does it work?

3. Deploy Telemetry to get telemetry



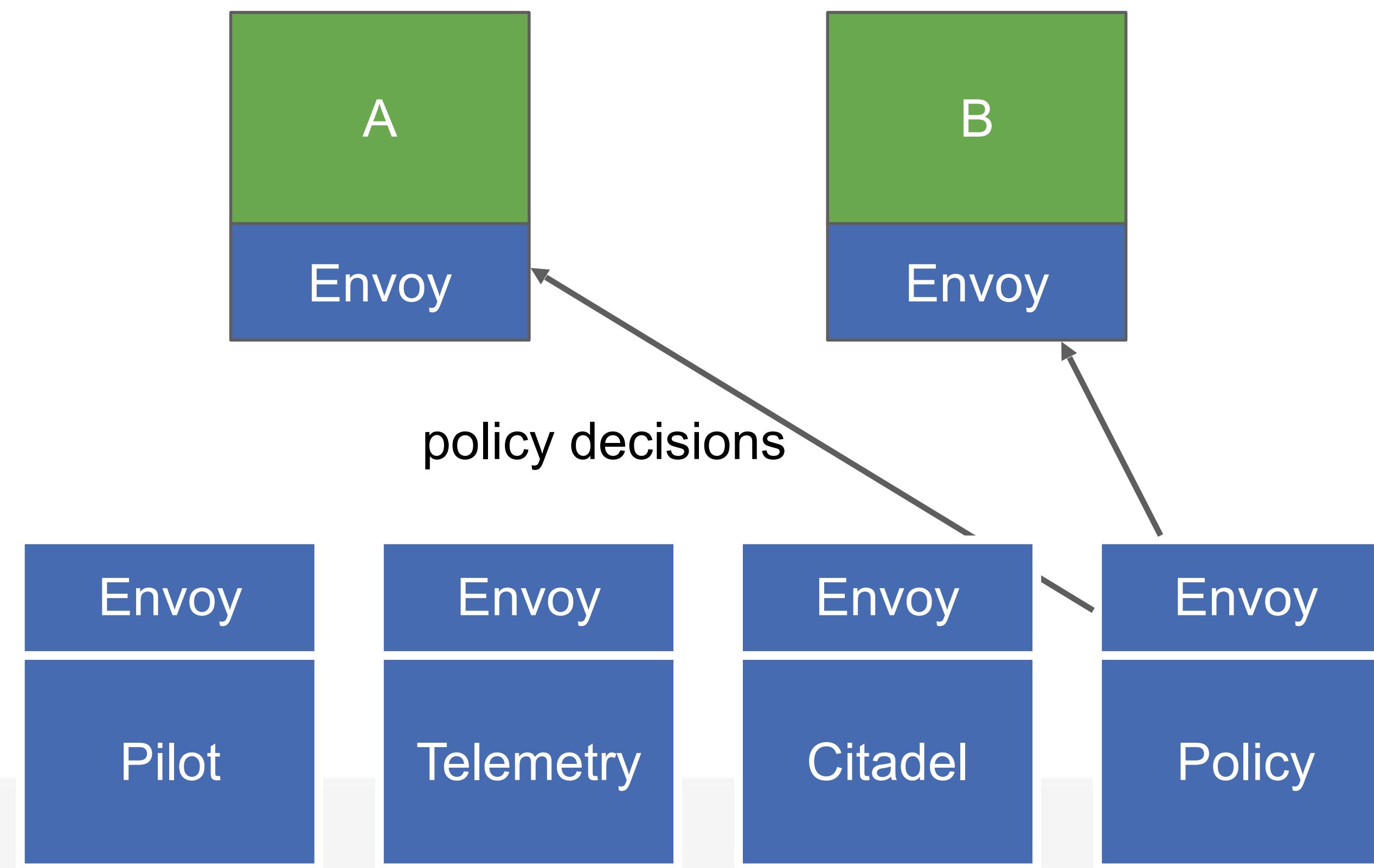
How does it work?

4. Deploy Citadel to assign identities and enable secure communication



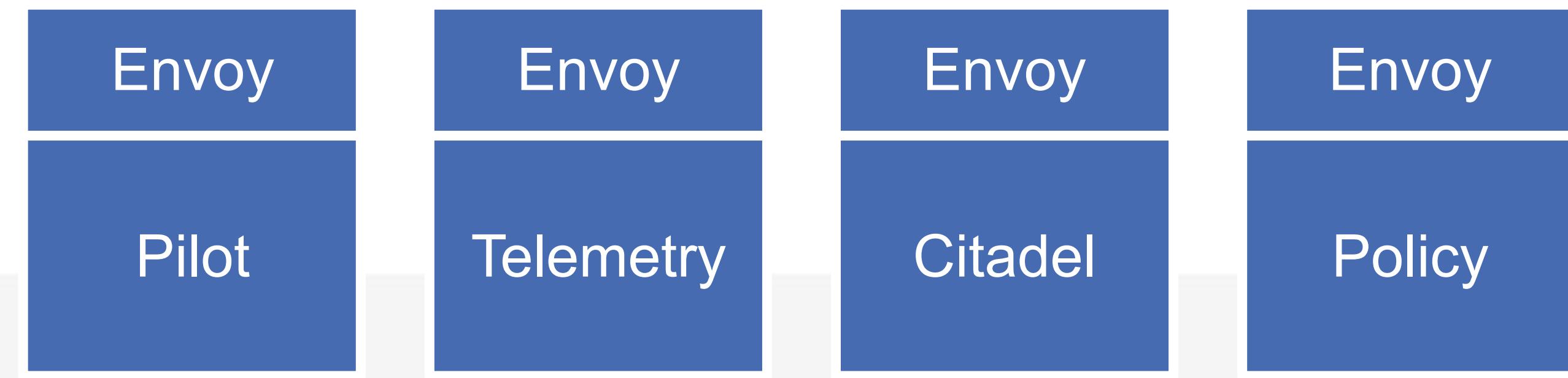
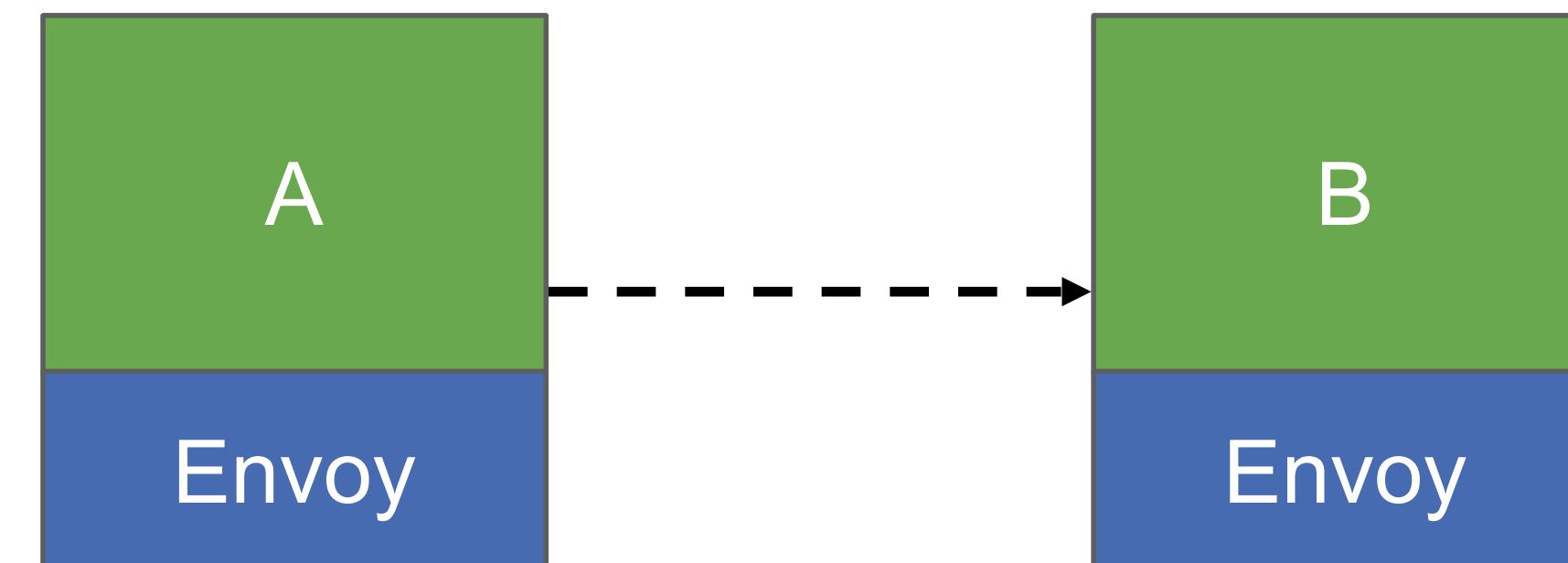
How does it work?

5. Deploy Policy to enforce policies



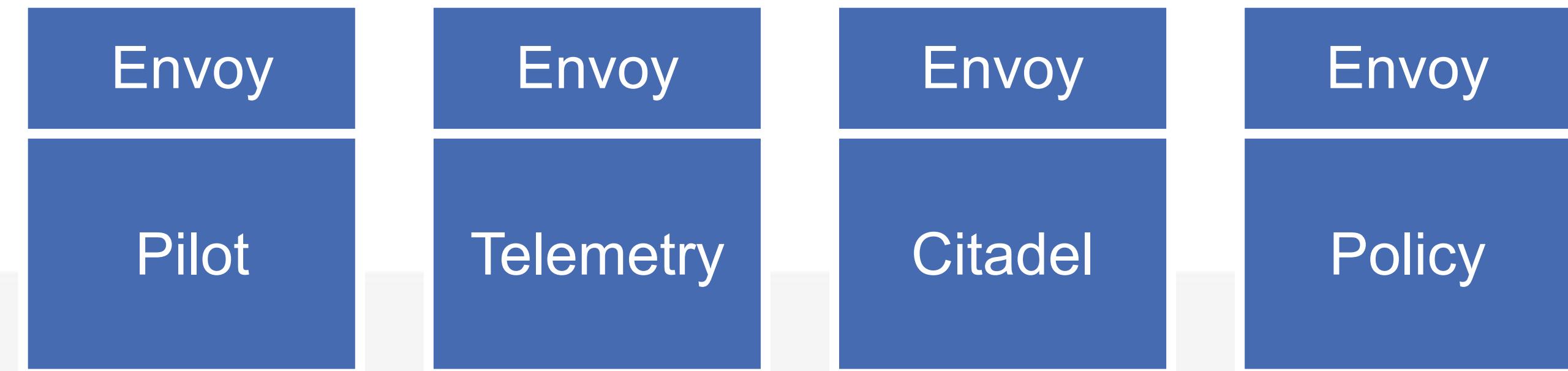
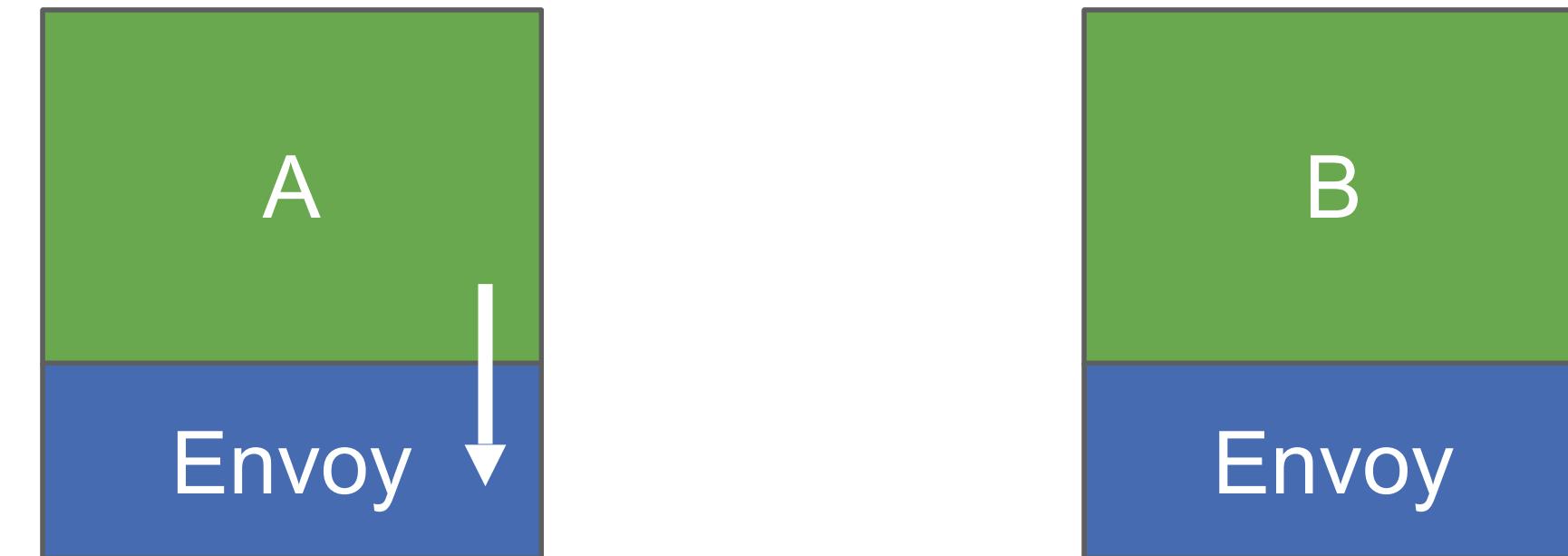
How does it work?

A calls B



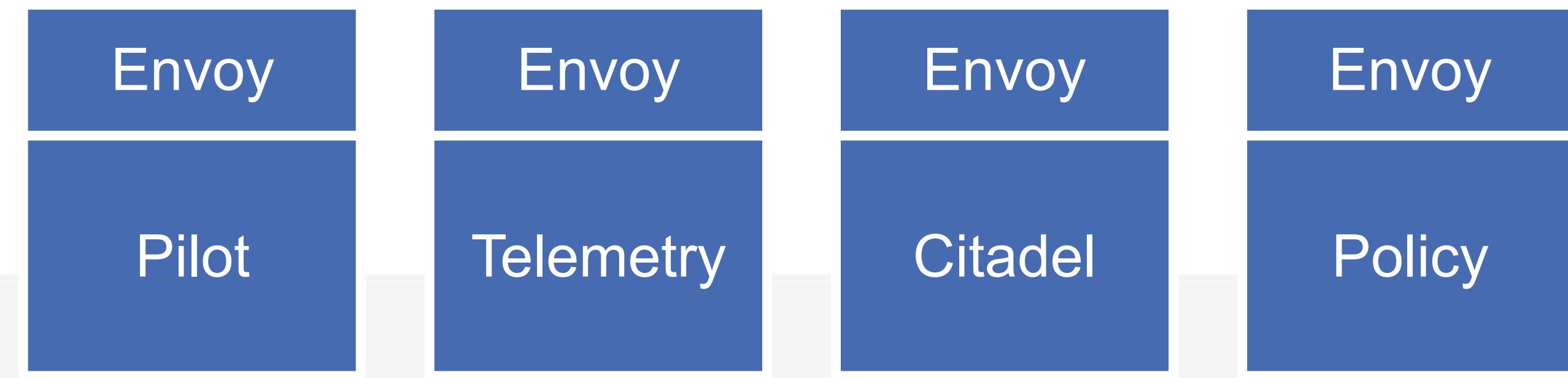
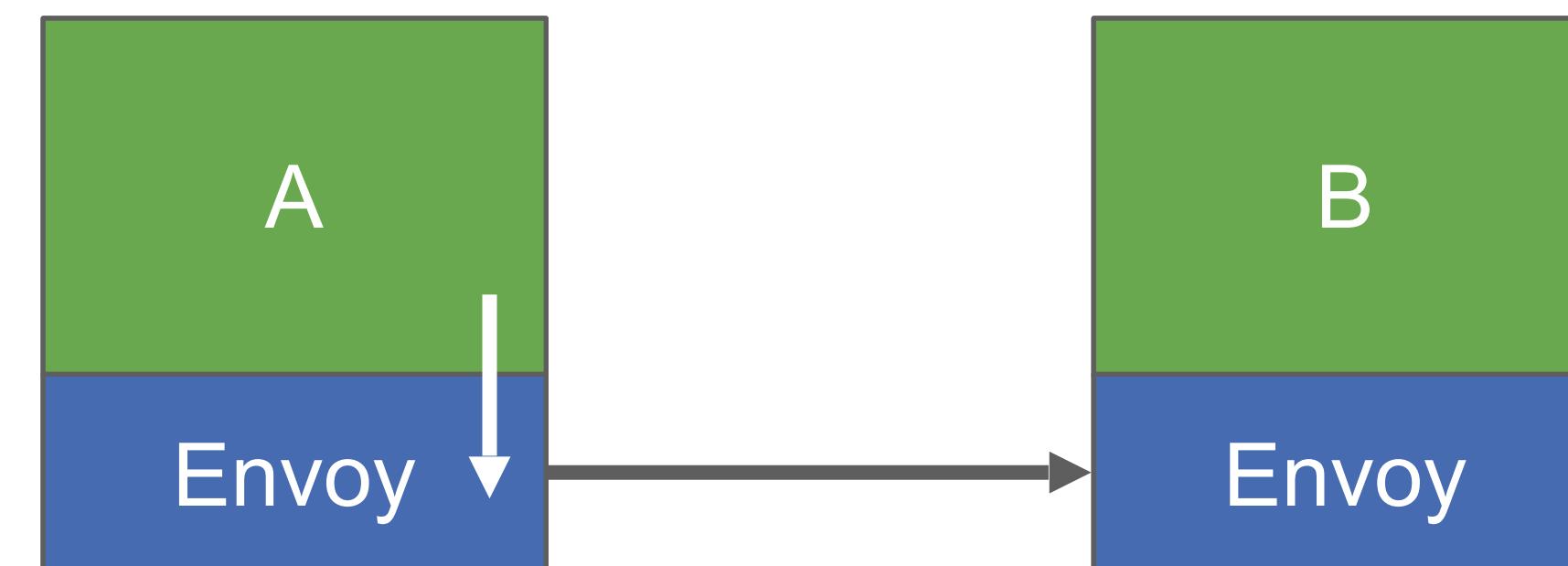
How does it work?

A's sidecar intercepts the call



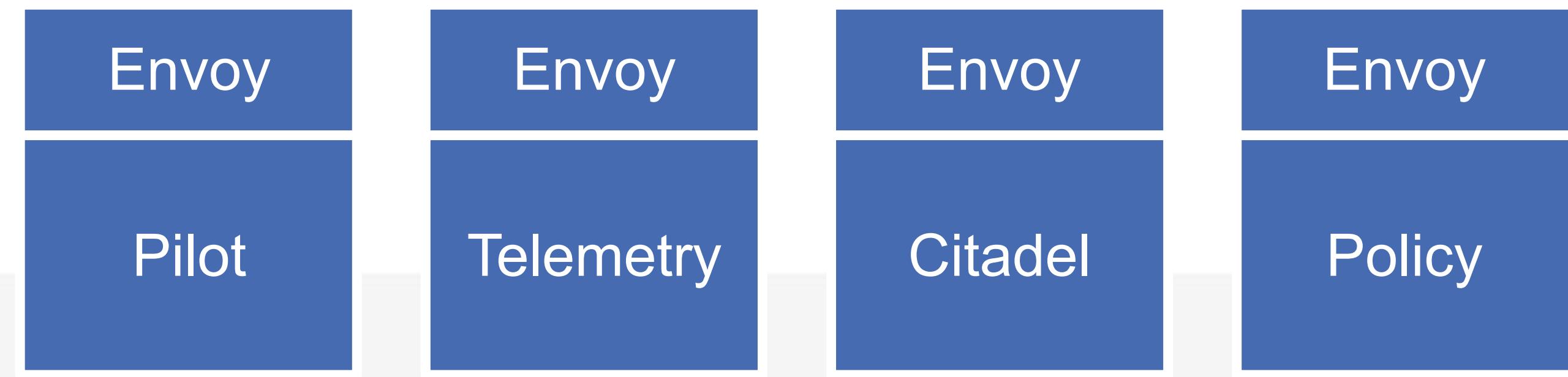
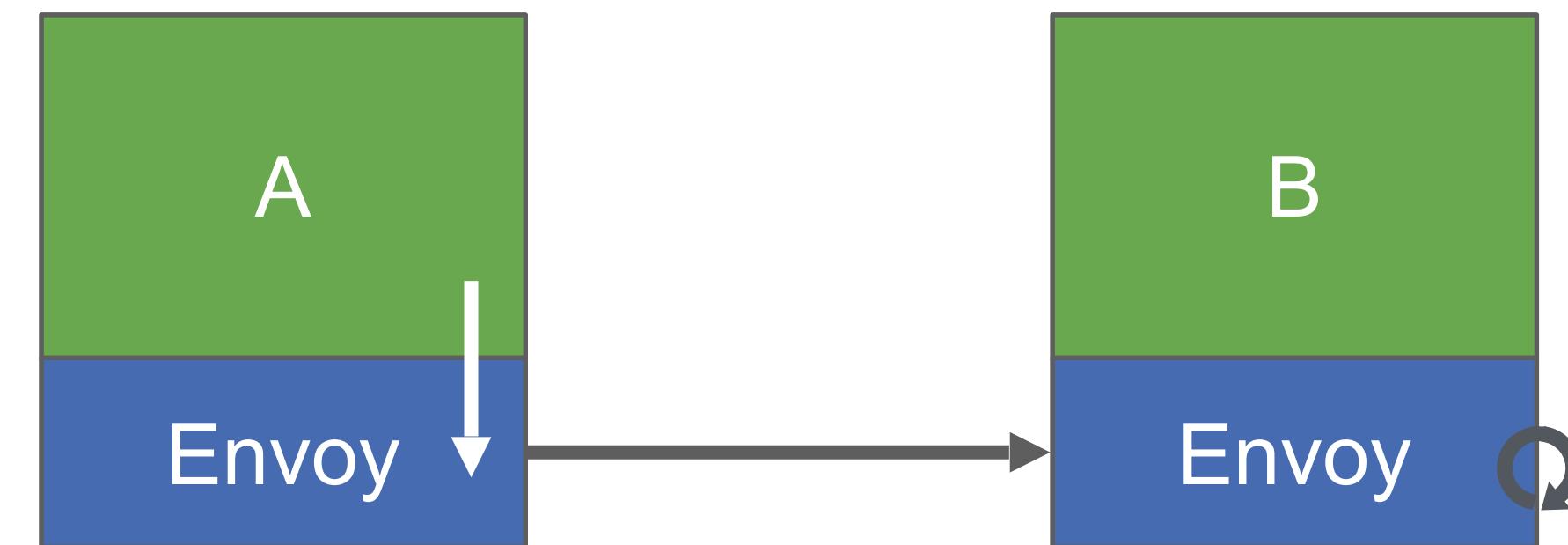
How does it work?

A's sidecar selects a destination



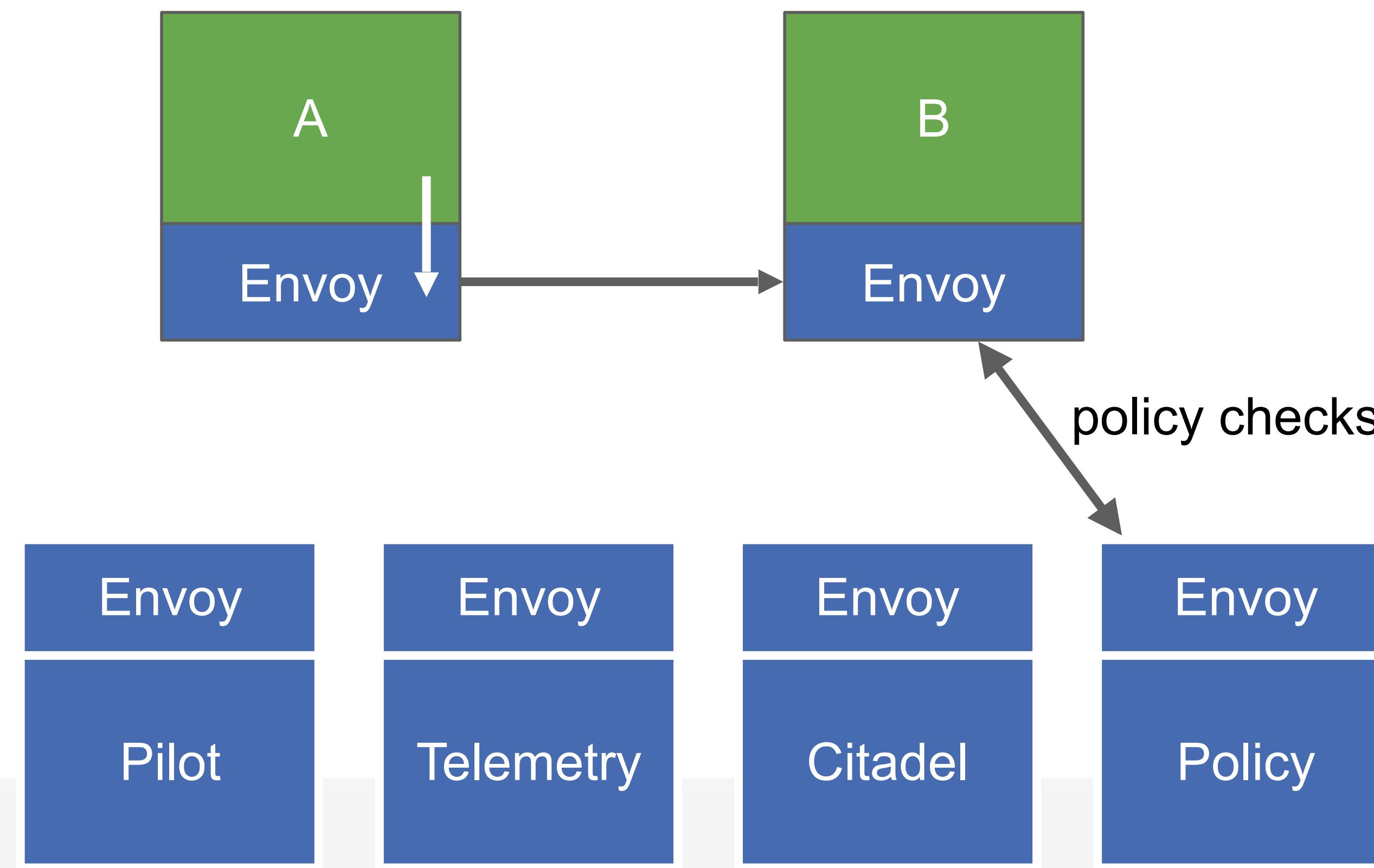
How does it work?

B's sidecar performs policy checks with local cache



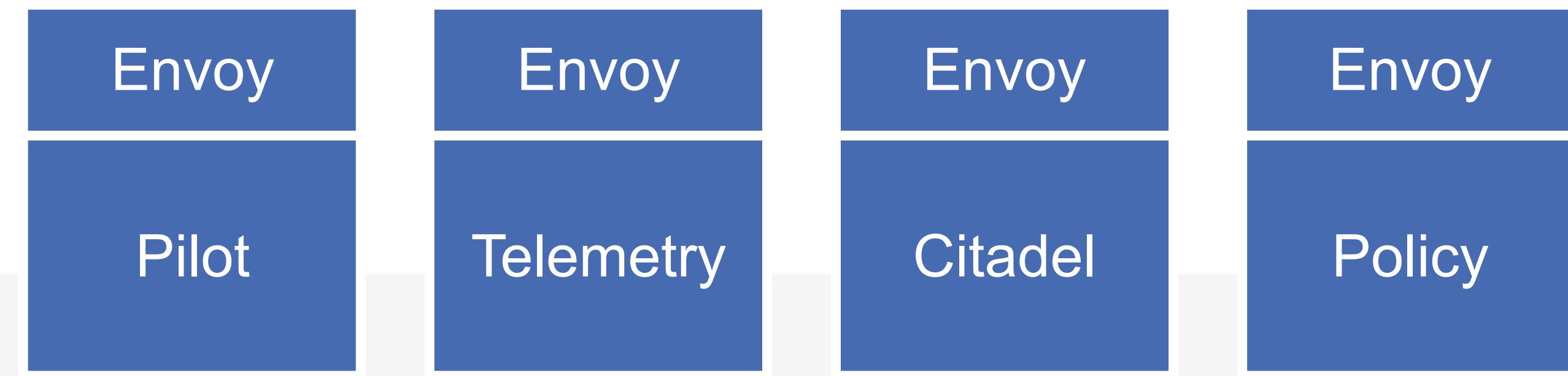
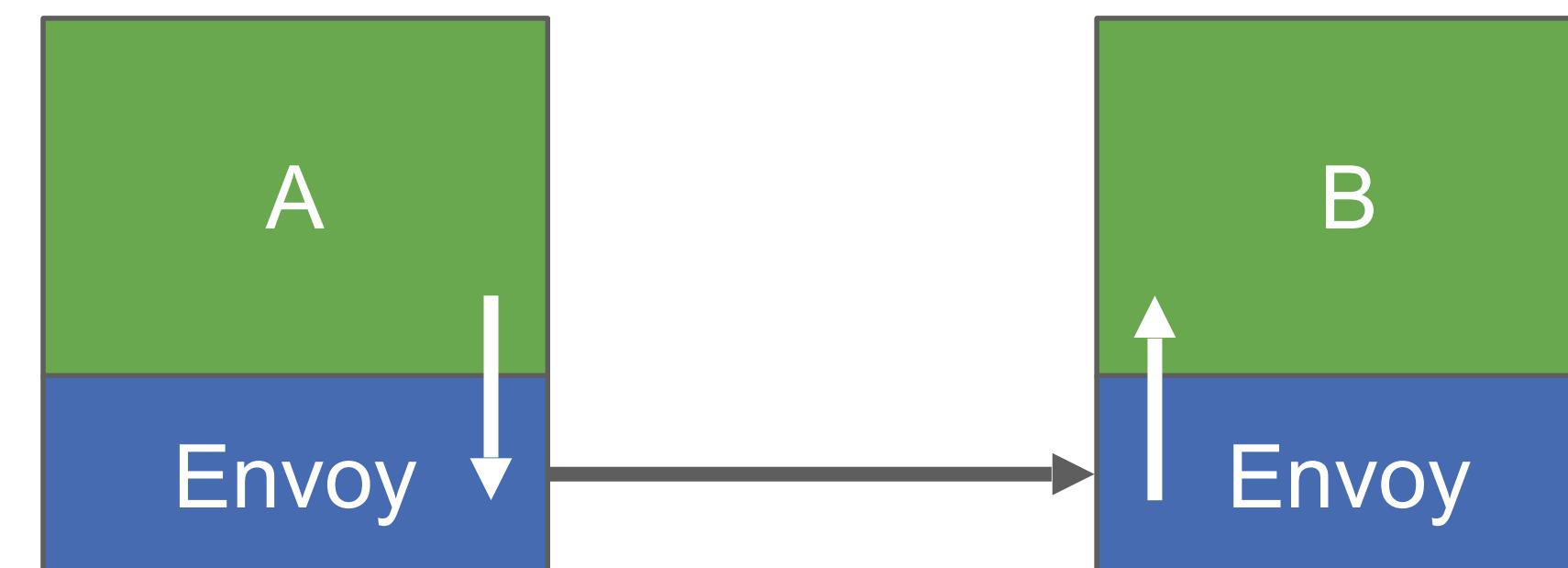
How does it work?

B's sidecar performs policy checks with Policy component



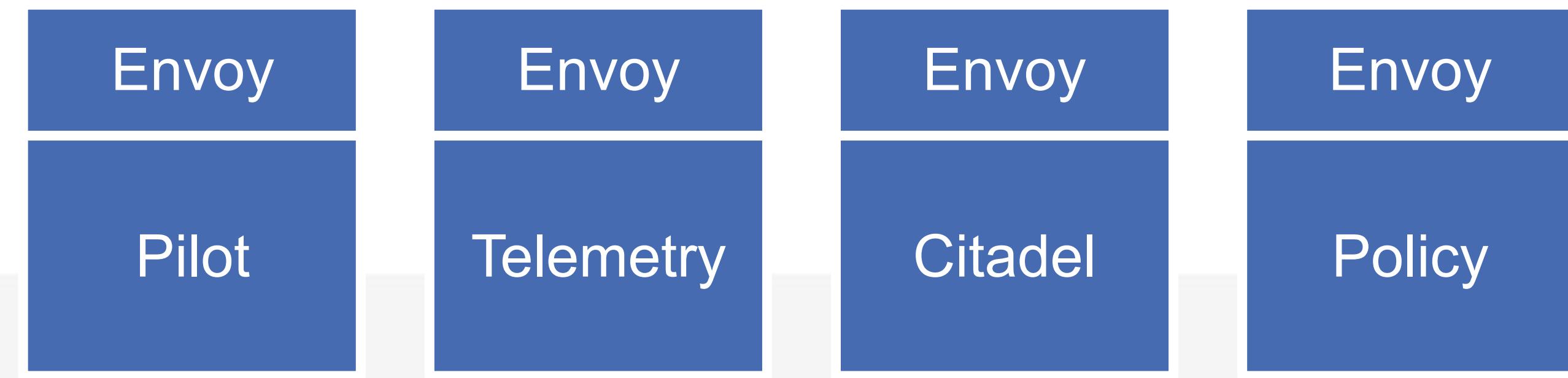
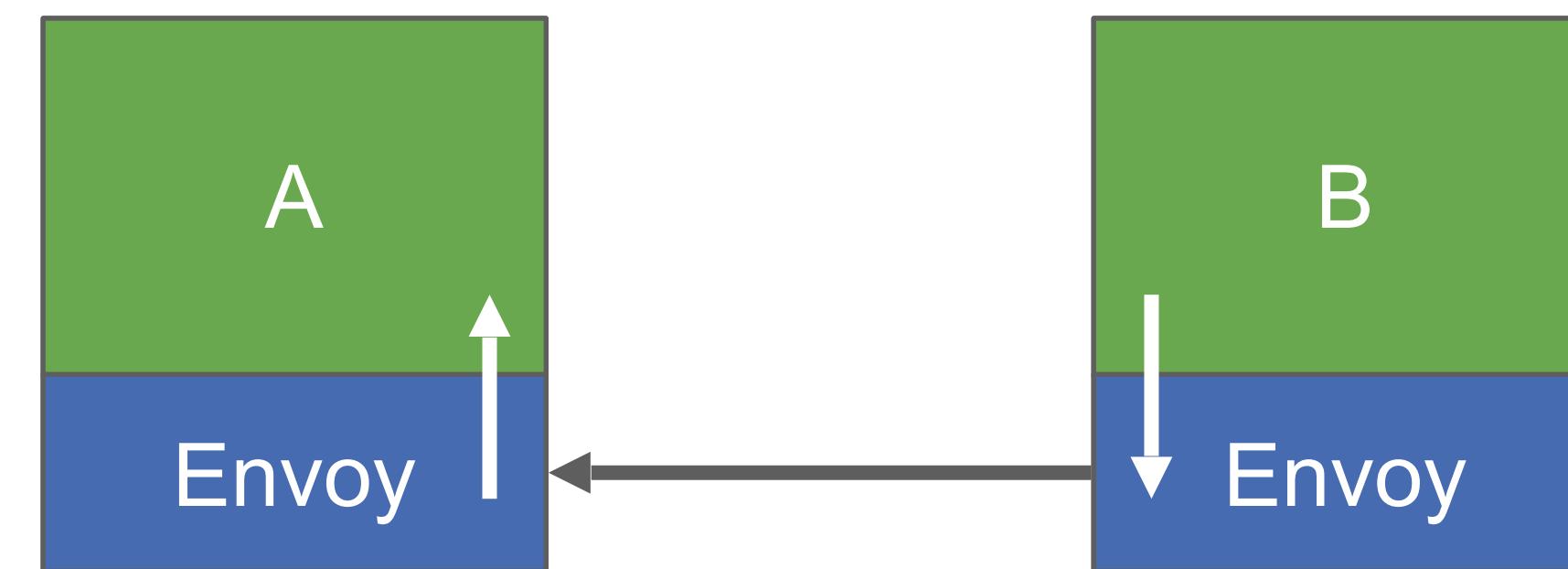
How does it work?

B's sidecar forwards the call to B



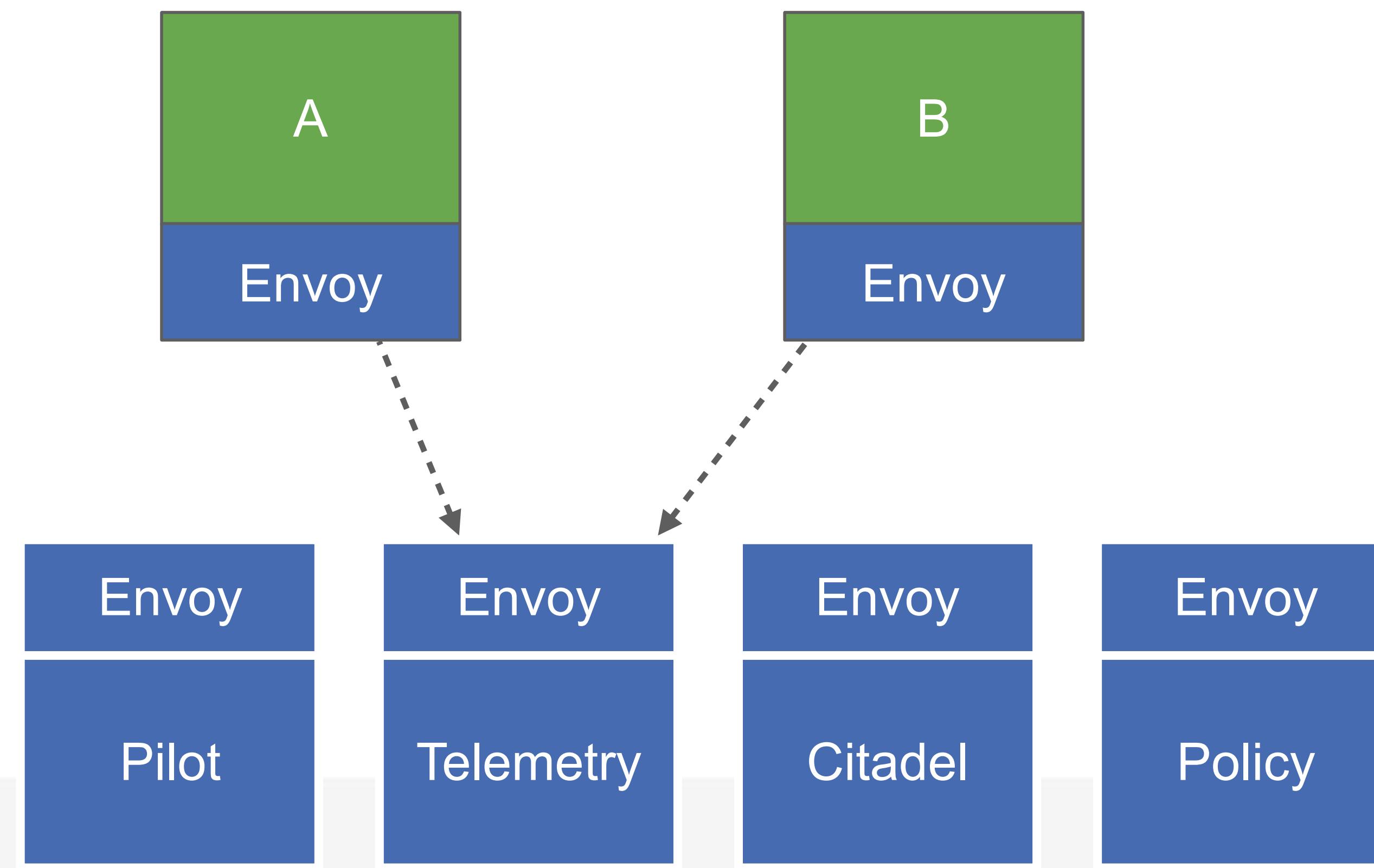
How does it work?

B's response is sent back

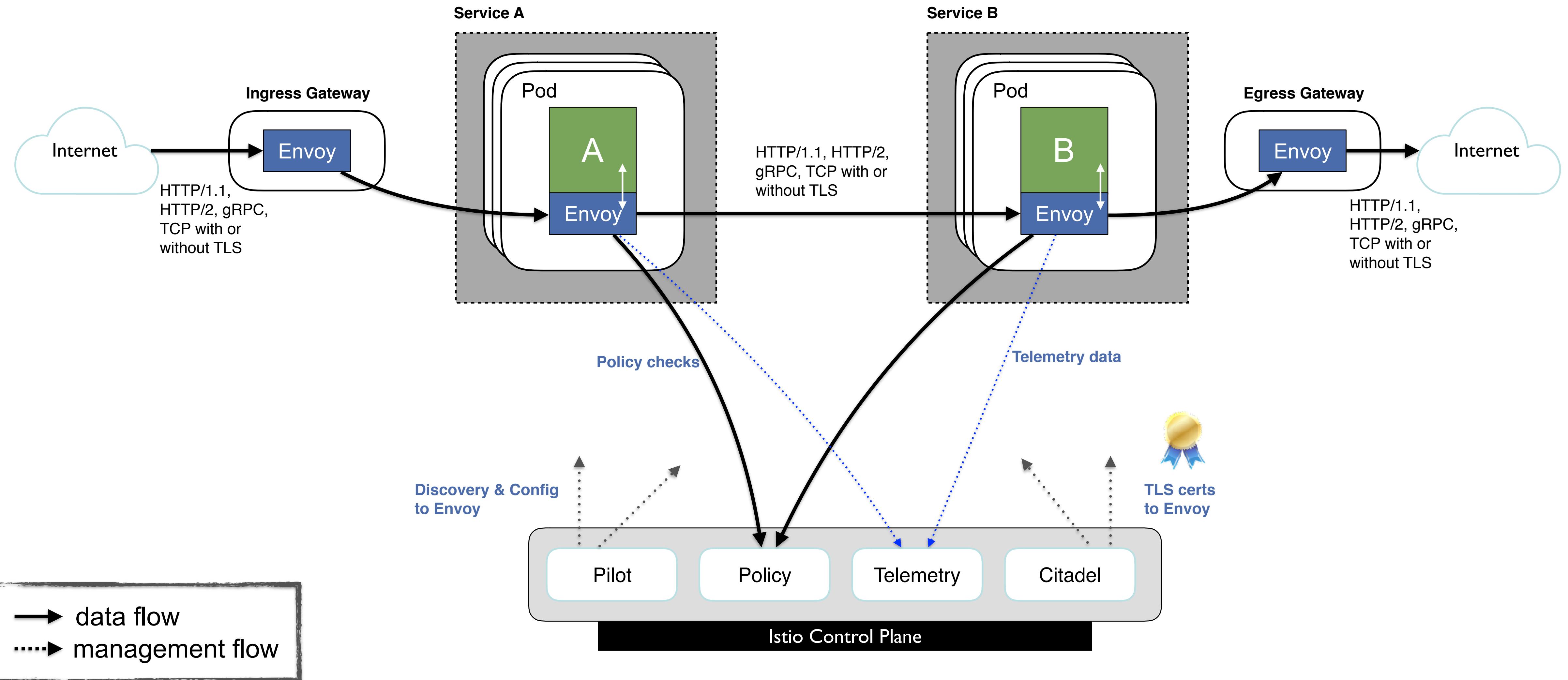


How does it work?

Both sidecars report telemetry data



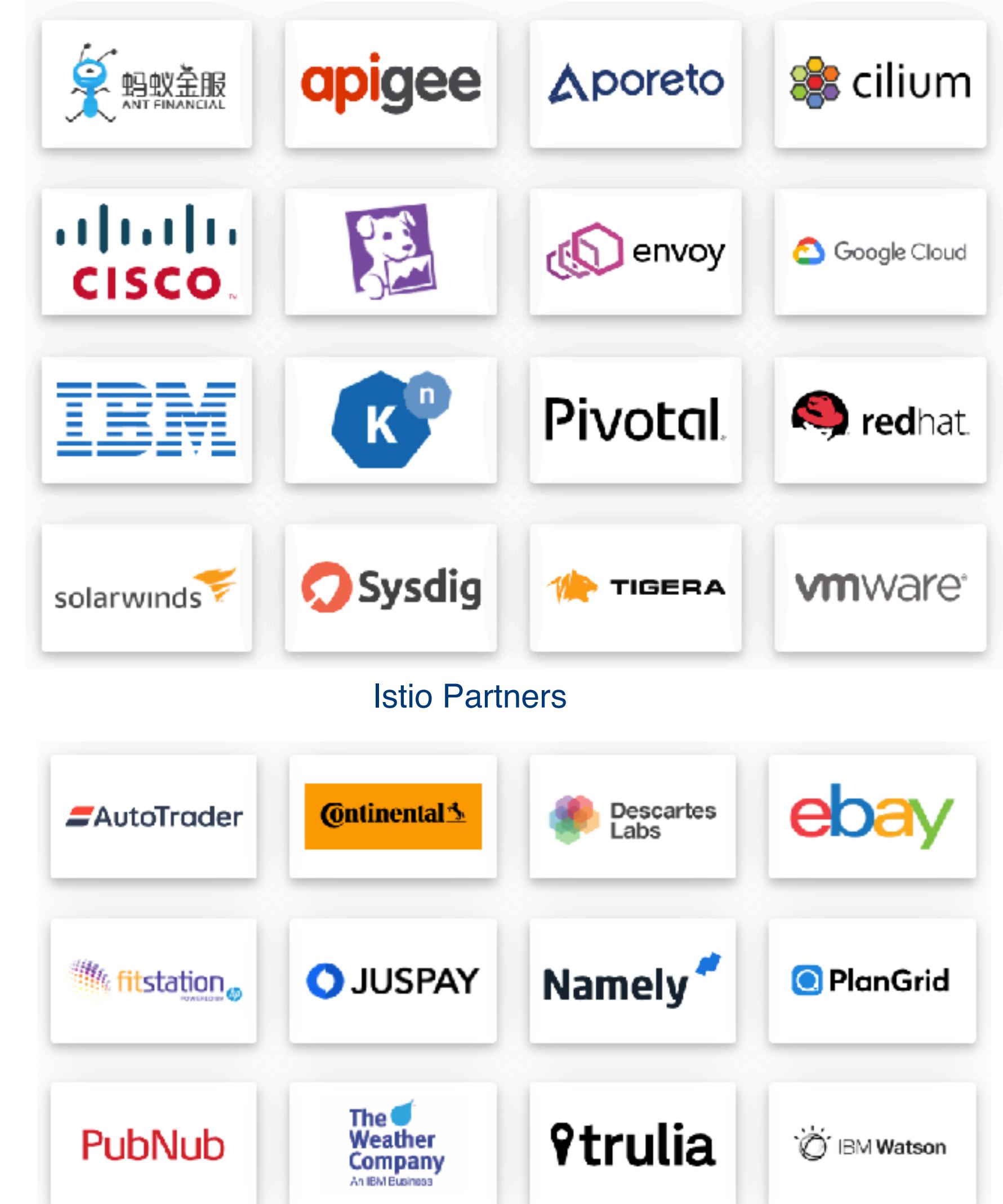
Wire it up together



Where we are

Istio 1.0

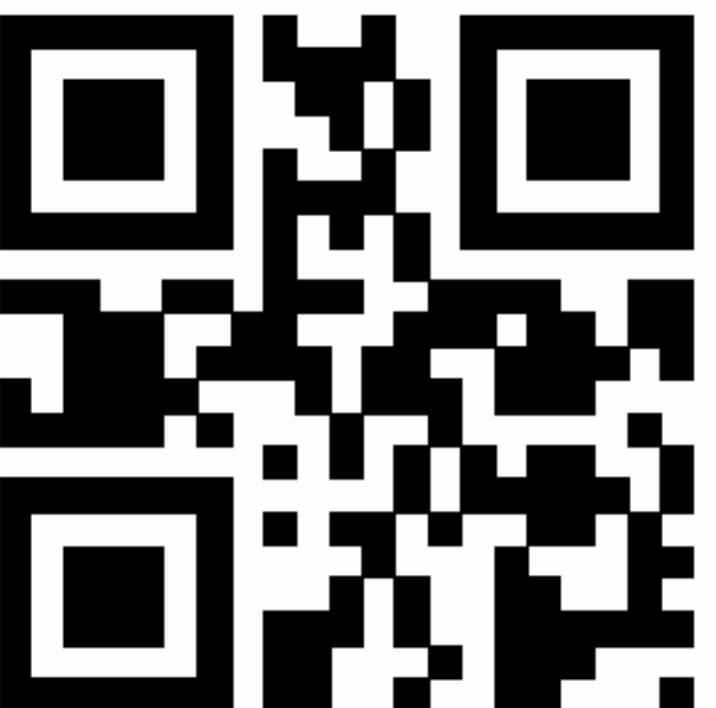
- ⛵ After ~2 years of work
- ⛵ ~200 developers
- ⛵ IBM, Google, VMWare, Cisco, Red Hat, Tigera, others...
- ⛵ Many adapters
- ⛵ Many customers



Istio in Action

Istio 1.0 highlights

- ❶ Flexible and highly configurable setup
- ❶ Minimal Istio
- ❶ Revamped traffic management model reaches Beta
- ❶ Multicluster support reaches Beta
- ❶ Incremental mTLS support
- ❶ Out of process Mixer adapter support
- ❶ Optimized authorization policies implementation
- ❶ Scalability and performance improvements
- ❶ 1.0.x addresses some critical issues found in 1.0



Istio has many features

Provides a network for **services**:

- ❶ Security
- ❶ Policy Enforcement
- ❶ Resiliency
- ❶ Traffic Control
- ❶ Observability

Key Features:

- ❶ Service authn and identity
- ❶ Authorization
- ❶ Rate limiting
- ❶ Load balancing / shedding
- ❶ Retries and circuit breaking
- ❶ Fine-grained routing
- ❶ Metrics and logs generation
- ❶ Request tracing
- ❶ Fault injection



The Weather Company

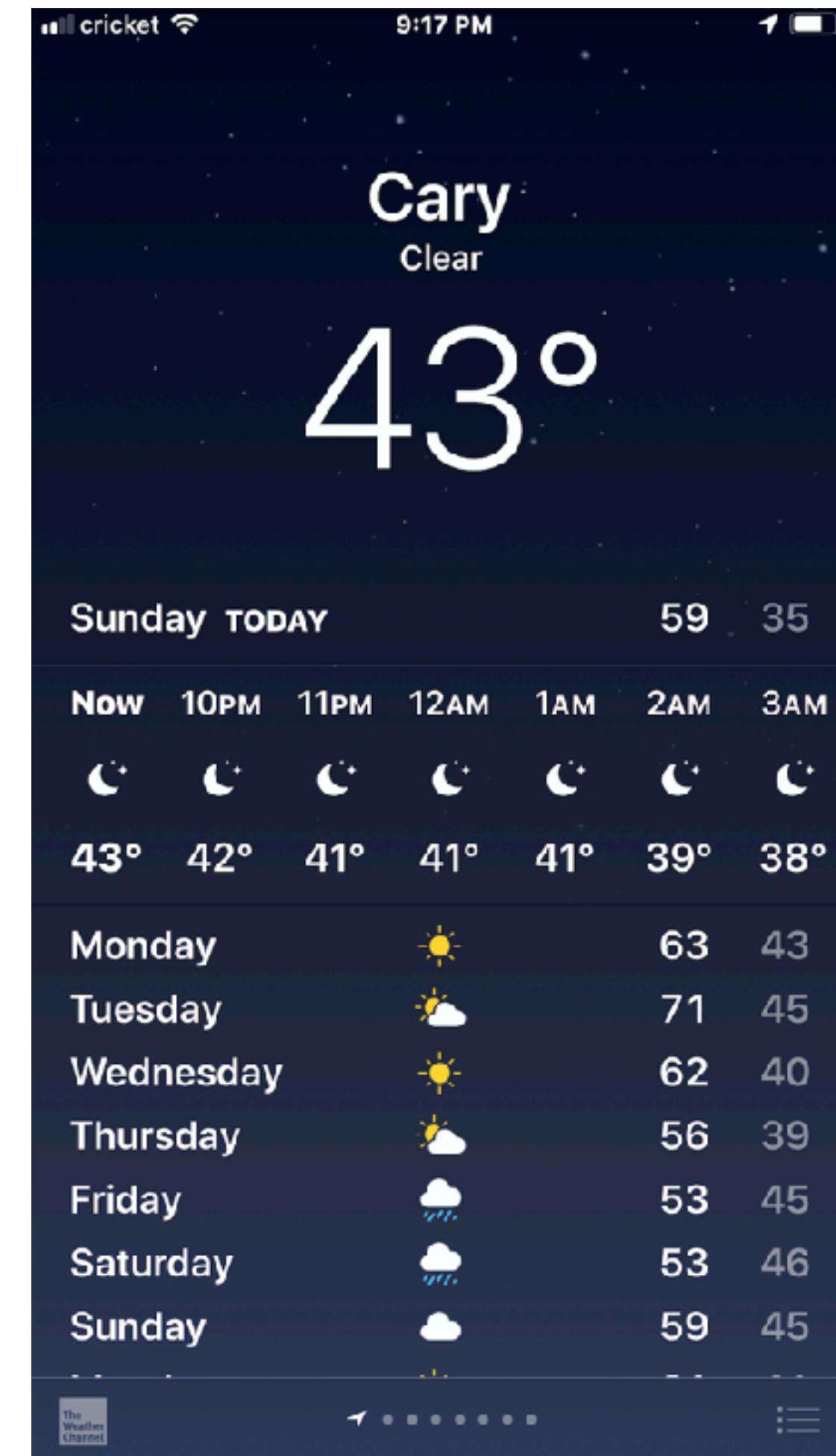


Why Istio?

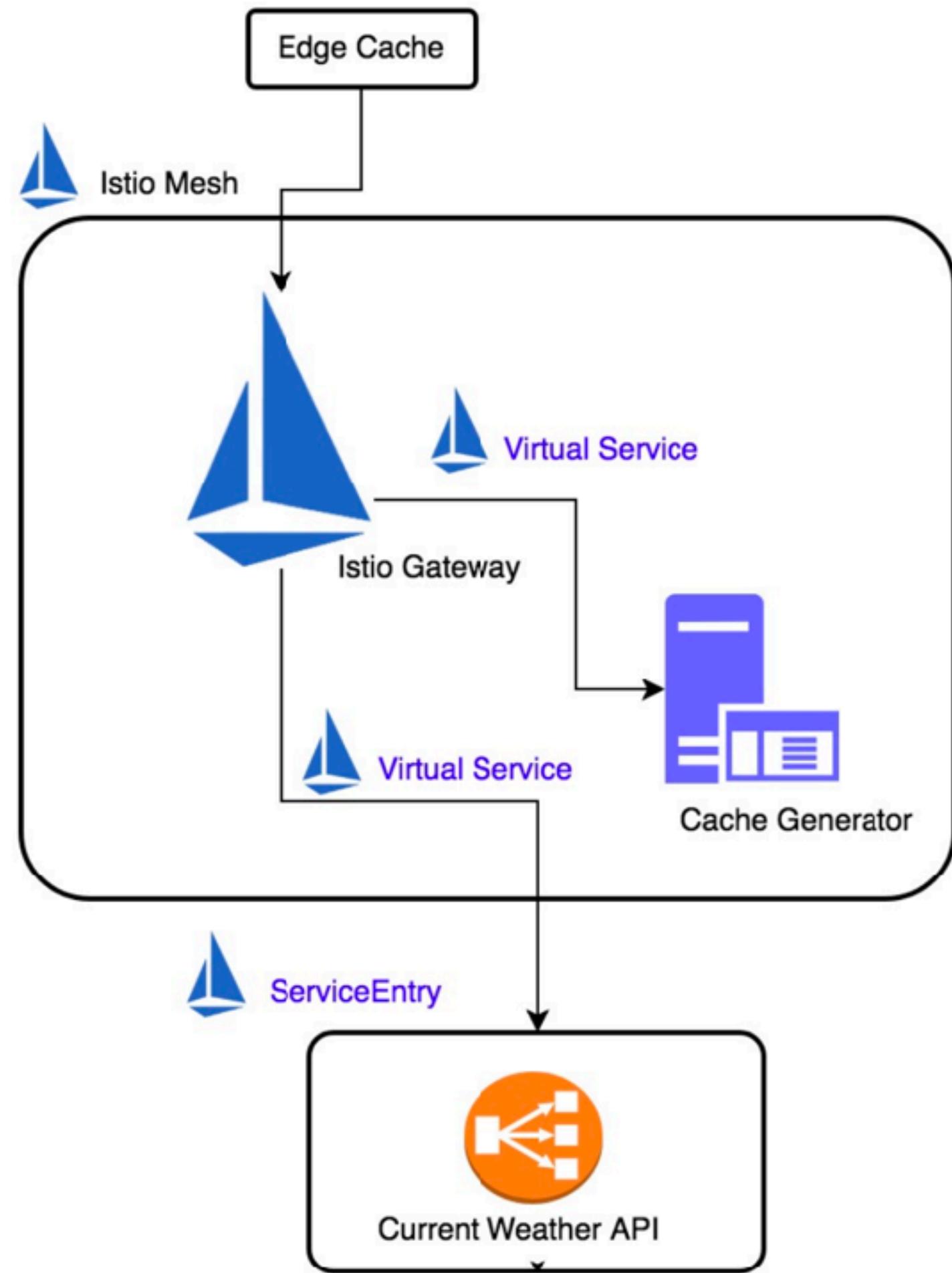
- ❶ Single platform for many tools
- ❶ Significantly less code to write
- ❶ Enterprise support and testing
- ❶ Cloud agnostic
- ❶ Cloud Native

Journey

- ❶ Using Istio for months
- ❶ api.weather.com not in production yet
- ❶ Some smaller APIs using Istio in production
- ❶ Actively working on 1.0 support
- ❶ Exploring traffic routing/shifting support



The Weather Company



Istio allows us to ...

- ➊ Resolve issues faster
- ➋ Mitigate backend issues
- ➌ Be more transparent
- ➍ Reduce use of custom code
- ➎ Make route changes at will

Nick Nellis talk: <https://www.youtube.com/watch?v=0fKi3NeCsSE>

Service Entries

```
apiVersion:  
networking.istio.io/v1alpha3  
kind: ServiceEntry  
metadata:  
  name: current-weather  
spec:  
  hosts:  
  - current-weather.internal-  
    weather.com  
  ports:  
  - number: 80  
    protocol: http  
    name: http
```

- ❶ 40 service entries
- ❷ Dynamically updateable
- ❸ Can use mTLS
- ❹ Gateways
- ❺ Virtual Services

Nick Nellis talk: <https://www.youtube.com/watch?v=0fKi3NeCsSE>

Routing - Cache Intercept

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: current-weather-cache
spec:
  gateways:
  - istio-gateway
  hosts:
  - api.weather.com
  http:
  - route:
    - destination:
        name: cache-generator
    - match:
    - uri:
        exact: /v3/wx/observations/current
```

Virtual service is new route rule

Dynamically updateable

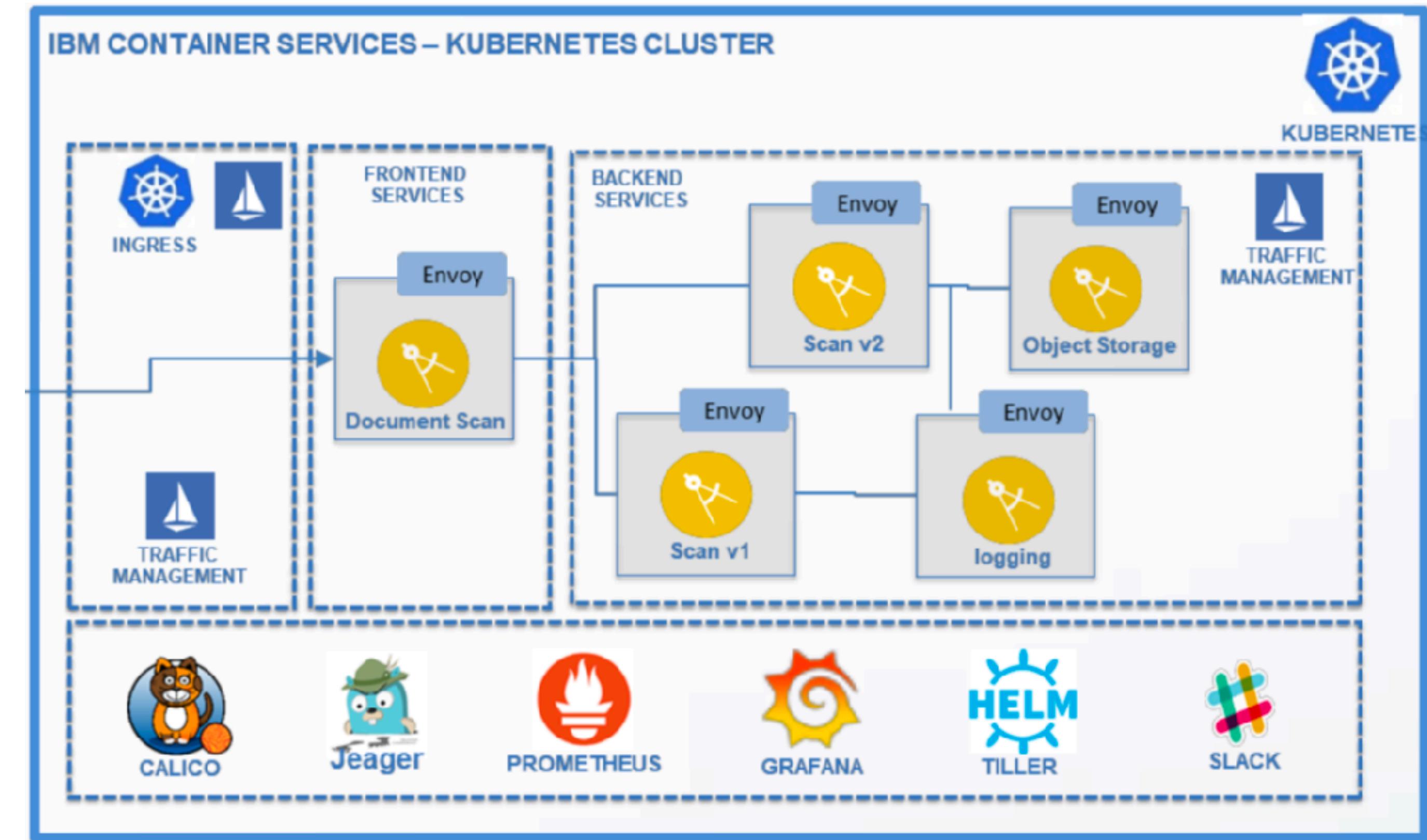
Gateway selector

Nick Nellis talk: <https://www.youtube.com/watch?v=0fKi3NeCsSE>

Large Airline

Leverage Traffic Management:

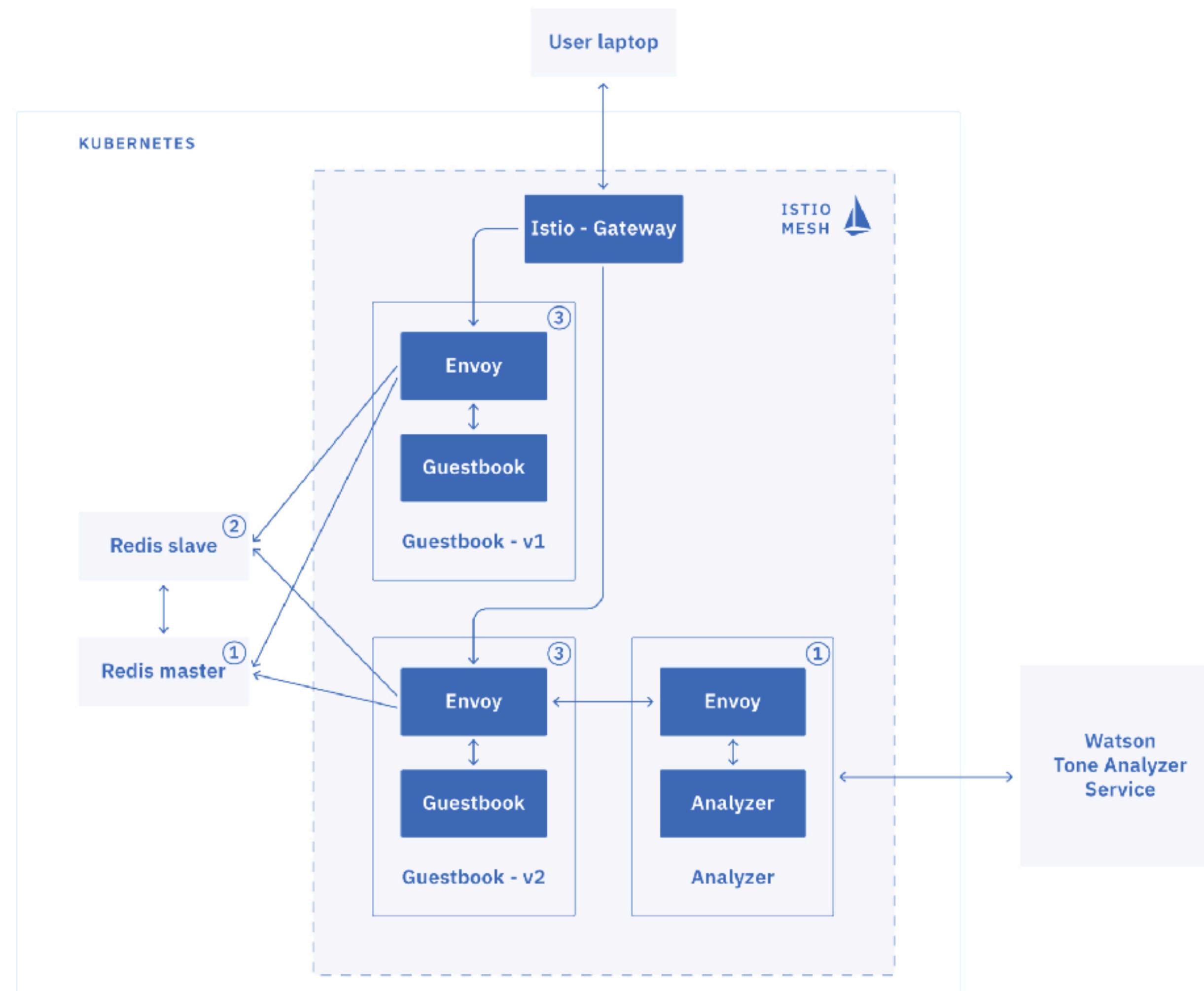
- ❶ Request Routing
- ❷ Circuit Breaking
- ❸ Route Rule
- ❹ Traffic mirroring
- ❺ Egress-Ingress Policies



Let us see it live!



The Guestbook Application



Guestbook - V1

very nice day today

Guestbook - V2

excited about today's conference : No Tone Detected

excited about today's conference : Joy (✿^‿^)

excited about today's conference

<https://github.com/IBM/guestbook>

What You Have Seen

- ❶ Service entry
- ❶ Ingress gateway
- ❶ Traffic management
- ❶ Distributed tracing
- ❶ Istio Mesh Metrics

Istio 1.1 and beyond

- ⛵ Make Istio easier to use and adopt
- ⛵ Incrementally adopt Istio
- ⛵ Ability to run mixer per node or sidecar
- ⛵ Support non routable L3 network among pods for multiclouds and multiple environments
- ⛵ Continued Performance and Scalability Improvement
- ⛵ Integration with API management
- ⛵ Various bug fixes!



Minimal Istio

The screenshot shows a website for 'Istio Prelim 1.1'. The top navigation bar includes a logo of a sailboat inside a circle, the text 'Istio Prelim 1.1', and links for 'Docs' and 'Blog'. On the left, a sidebar menu lists categories like 'Concepts', 'Setup', and 'Kubernetes', with 'Minimal Istio Installation' under 'Kubernetes'. The main content area features a title 'Minimal Istio Installation' with a Helm icon, a '2 MINUTE READ' note, and a brief description: 'Quick start instructions for the minimal setup and configuration of Istio using Helm. This minimal install provides traffic management features of Istio.' Below this are sections for 'Prerequisites' (with a note to refer to the Quick Start guide) and 'Installation steps'. The first step involves running a command: '\$ kubectl apply -f install/kubernetes/helm/istio/templates/crds.yaml'. The second step is to choose one of two mutually exclusive options. A final section, 'Option 1: Install with Helm via helm template', is shown.

Minimal Istio Installation

2 MINUTE READ

Quick start instructions for the minimal setup and configuration of Istio using Helm. This minimal install provides traffic management features of Istio.

Prerequisites

Refer to the [prerequisites](#) described in the Quick Start guide.

Installation steps

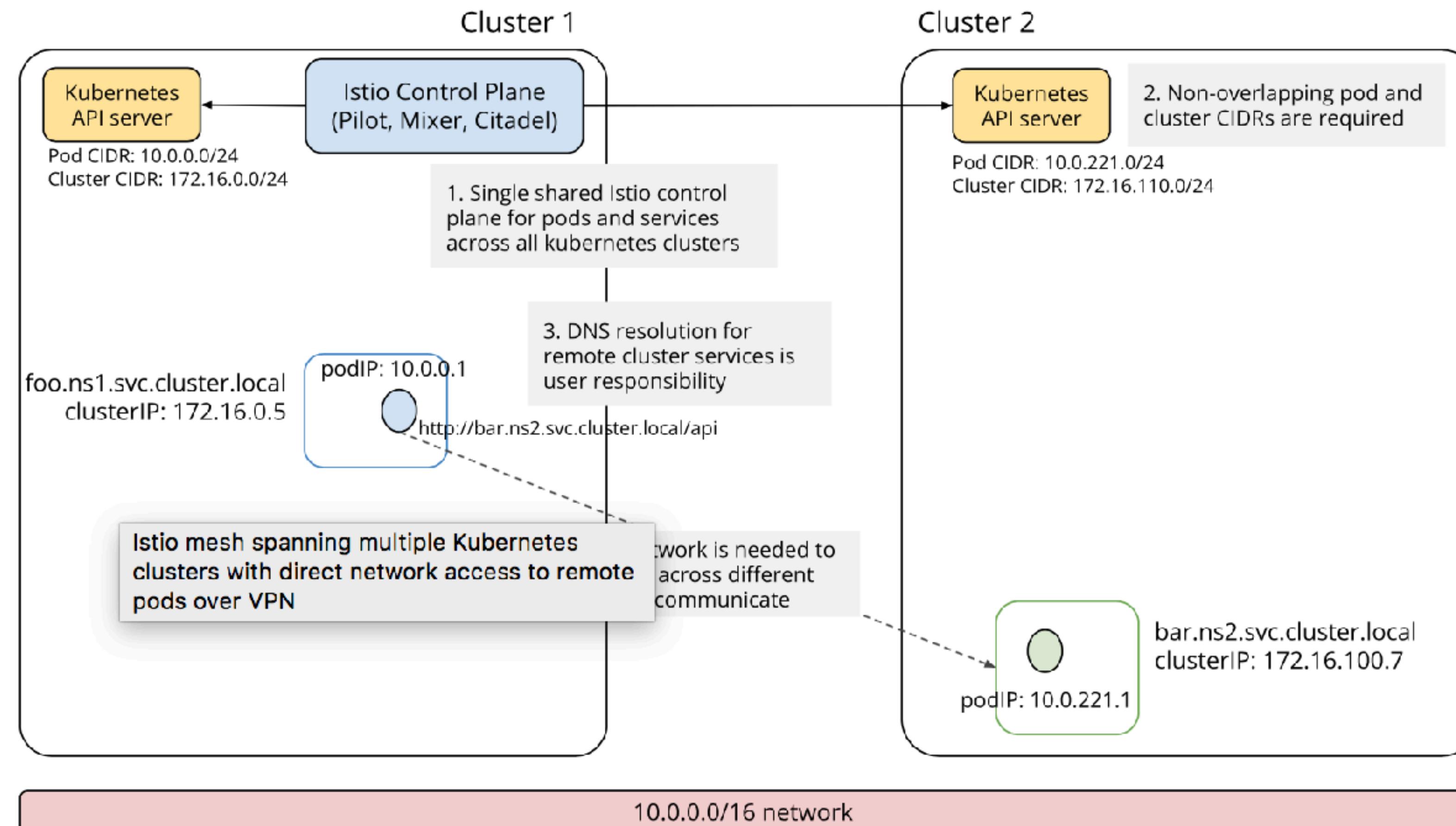
- If using a Helm version prior to 2.10.0, install Istio's [Custom Resource Definitions](#) via `kubectl apply`, and wait a few seconds for the CRDs to be committed in the kube-apiserver:

```
$ kubectl apply -f install/kubernetes/helm/istio/templates/crds.yaml
```
- Choose one of the following two **mutually exclusive** options described below.

Option 1: Install with Helm via `helm template`

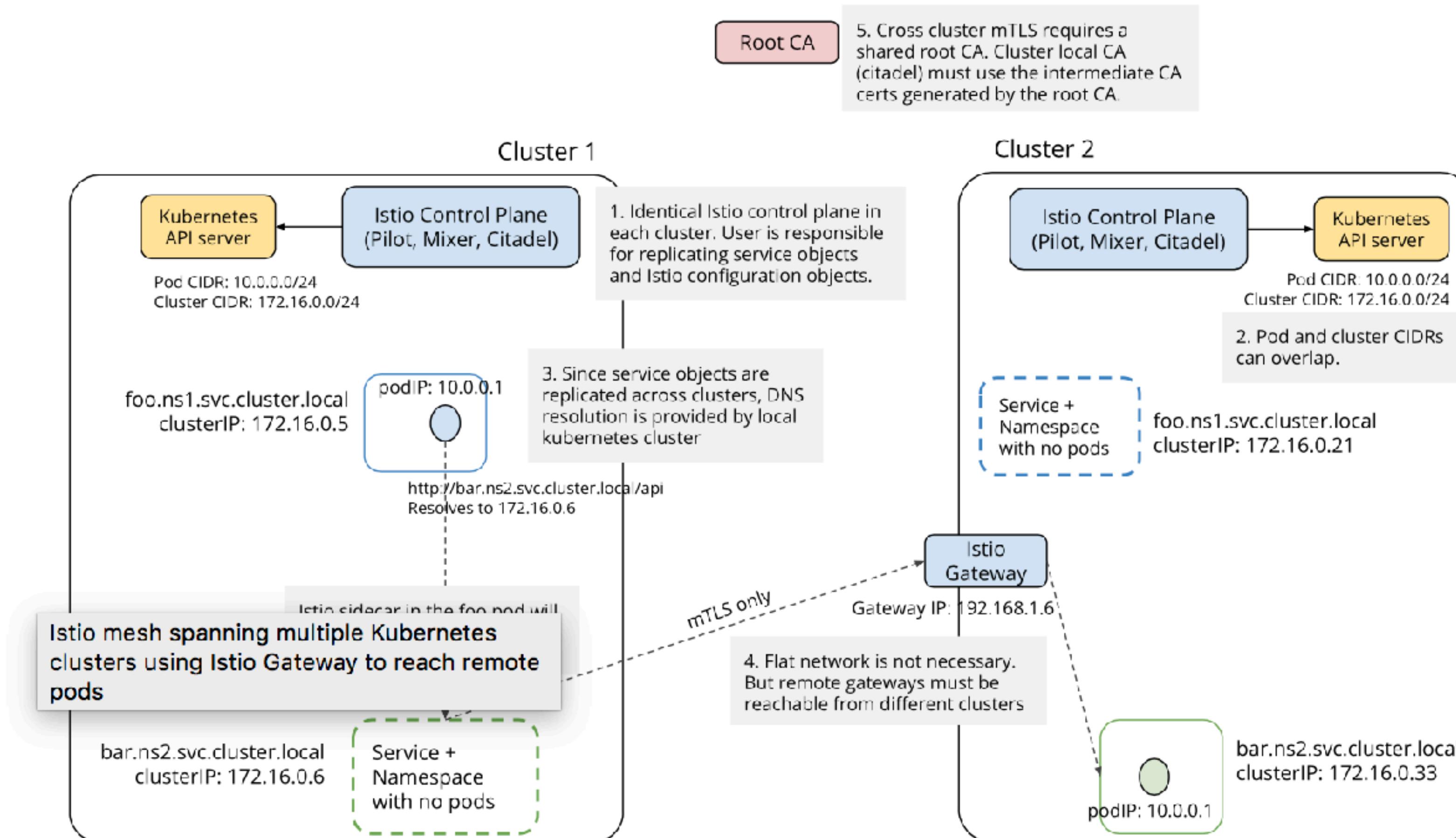
1 Render Istio's core components to a Kubernetes manifest called `istio.yaml`.

Istio Multicluster Single Control Plane



Istio mesh spanning multiple Kubernetes clusters with direct network access to remote pods over VPN

Federated Istio Multicloud via Gateway



Istio mesh spanning multiple Kubernetes clusters using Istio Gateway to reach remote pods



Try Istio today! Thanks!

Contact us:

istio-user@googlegroups.com

istio-dev@googlegroups.com

[@IstioMesh](#)

More information:

istio.io

istio.io/about/community/

github.com/istio