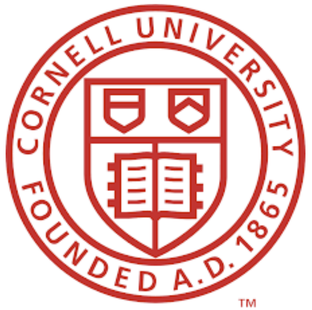
EXPRESSIVE POLICIES FOR MICROSERVICE NETWORKS

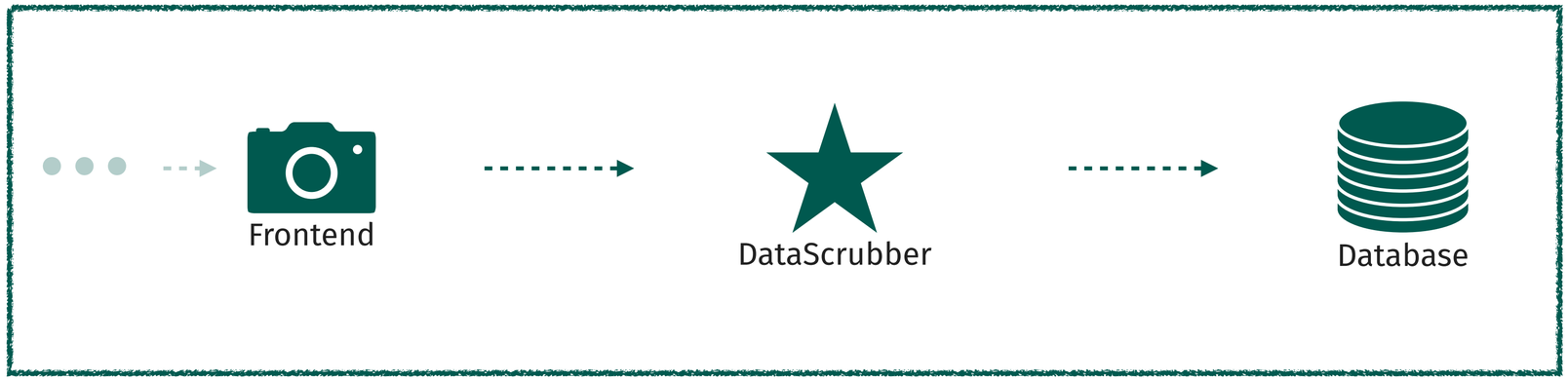
HotNets'23

Karuna Grewal, Brighten Godfrey, Justin Hsu



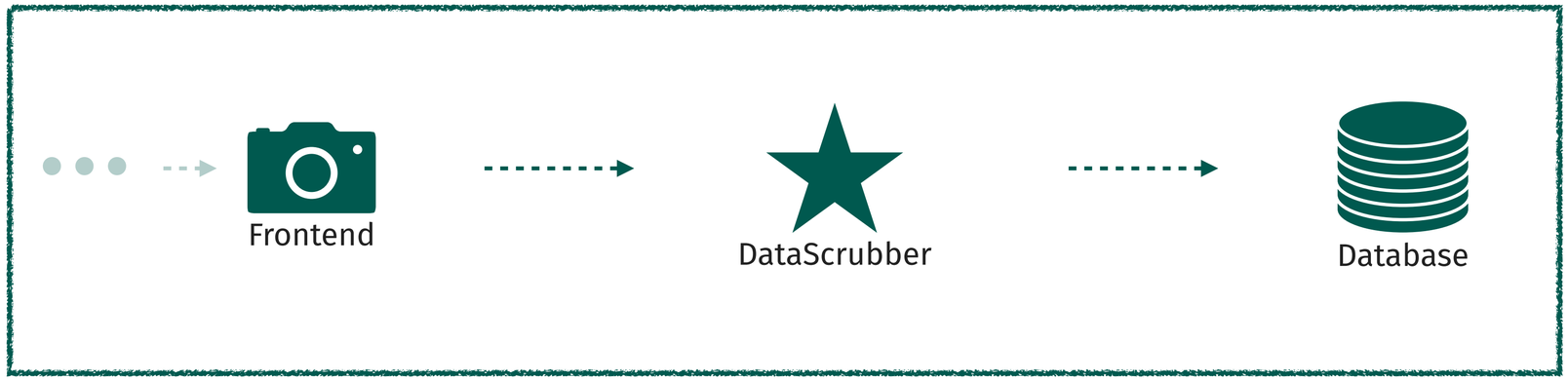
SAFETY PROPERTIES IN MICROSERVICES

Example: Data Scrubbing



SAFETY PROPERTIES IN MICROSERVICES

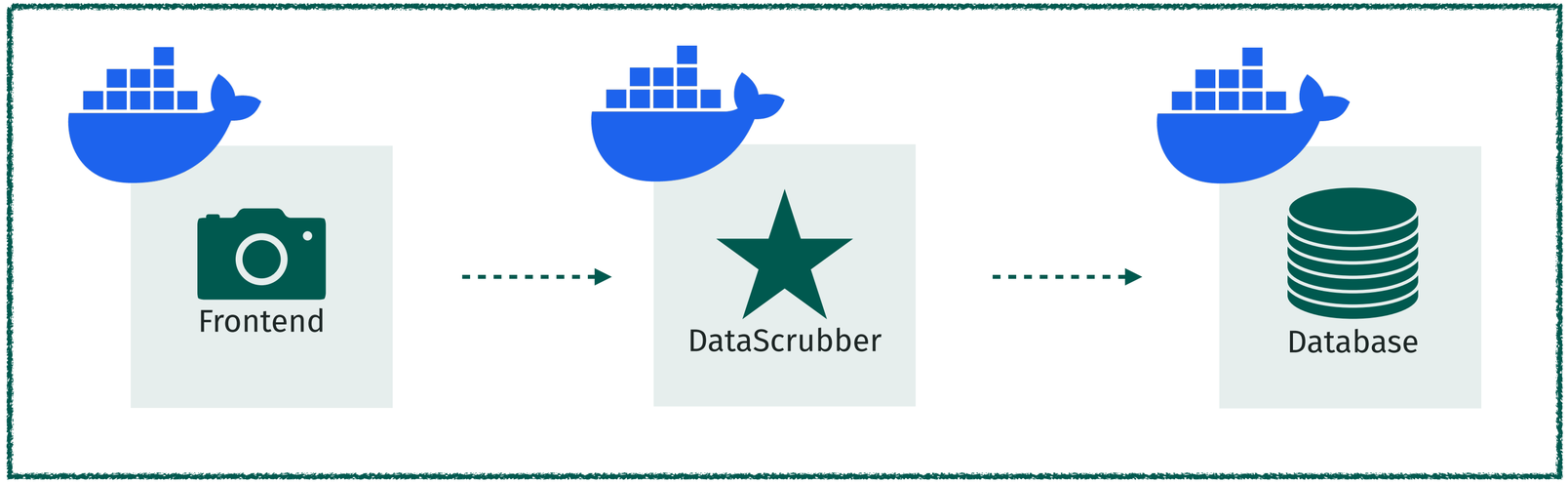
Example: Data Scrubbing



GOAL: To prevent leaking personally identifiable information

EXISTING ENFORCEMENT TECHNIQUES

Deployment partitioned across multiple containers



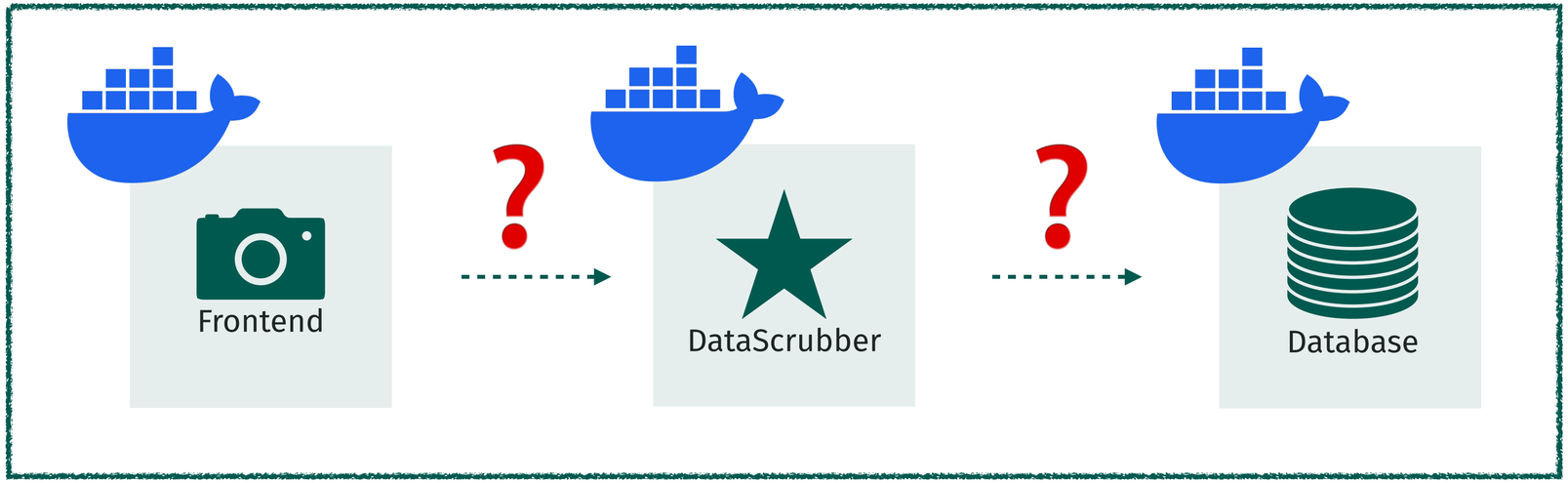
Kubernetes

Service Meshes, like Istio, Cilium

EXISTING ENFORCEMENT TECHNIQUES

Deployment partitioned across multiple containers

Can only express point-to-point policies



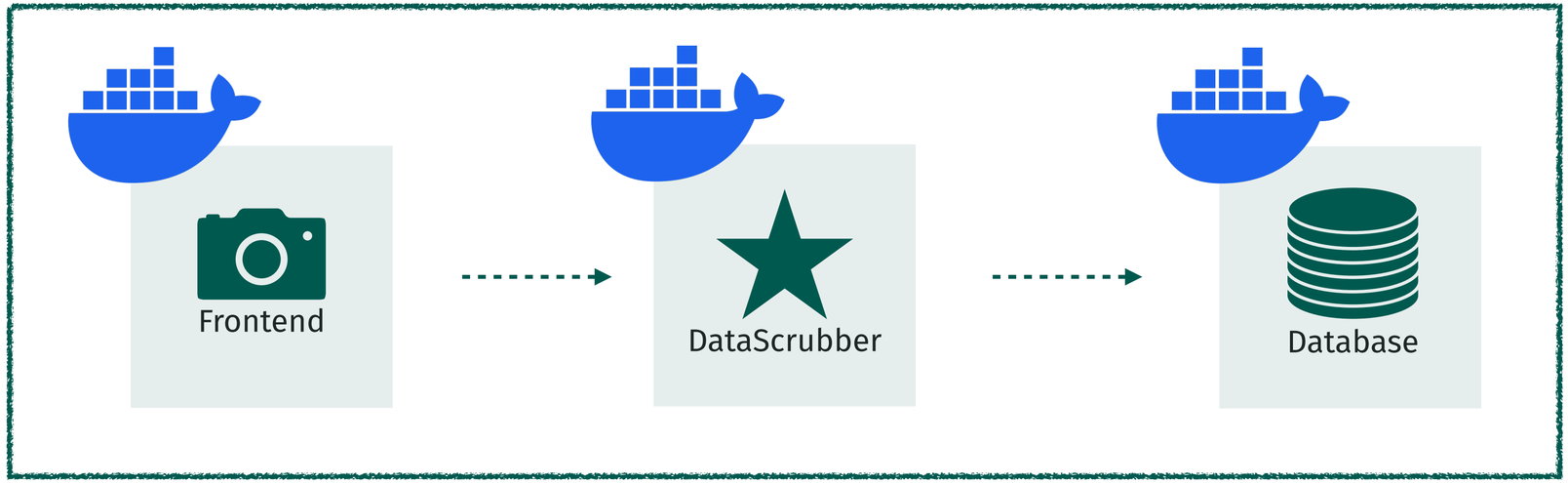
Kubernetes

Service Meshes, like Istio, Cilium

EXISTING ENFORCEMENT TECHNIQUES

Deployment partitioned across multiple containers

Cannot enforce this data scrubber policy



Kubernetes

Service Meshes, like Istio, Cilium

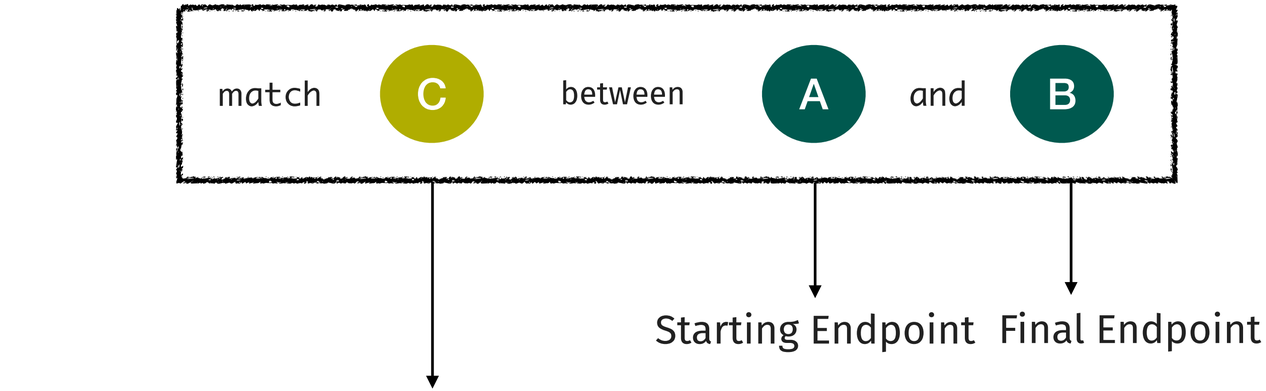
CONTRIBUTIONS

- **Expressive policy language** for intermediate service interactions.
- **Service layer networking for policy enforcement**.

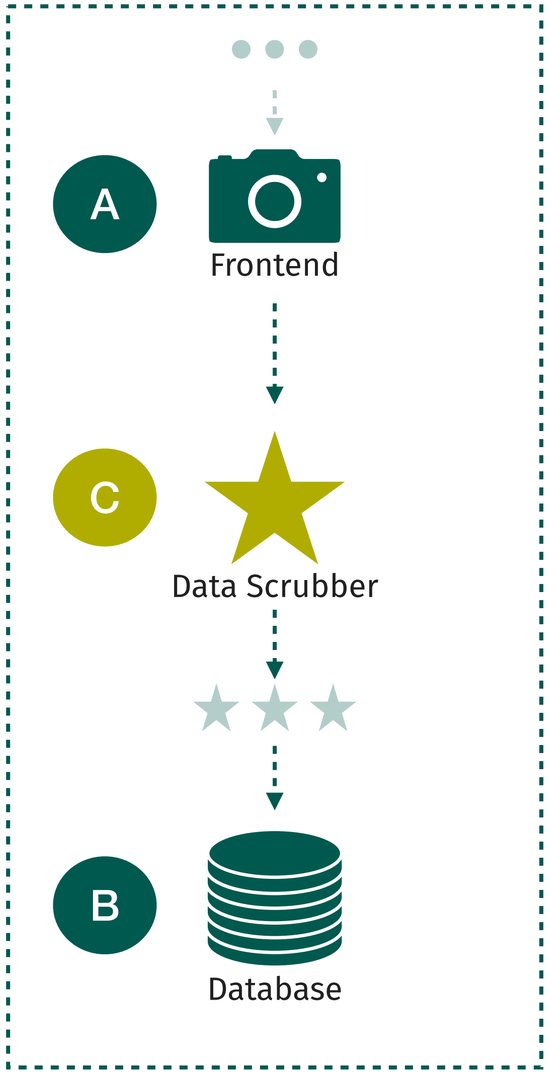
1. Expressive Policy Language

Expressing Policies on Intermediate Service Interactions

Generally,



Expressing Policies on Intermediate Service Interactions



Generally,

match C between A and B

Starting Endpoint

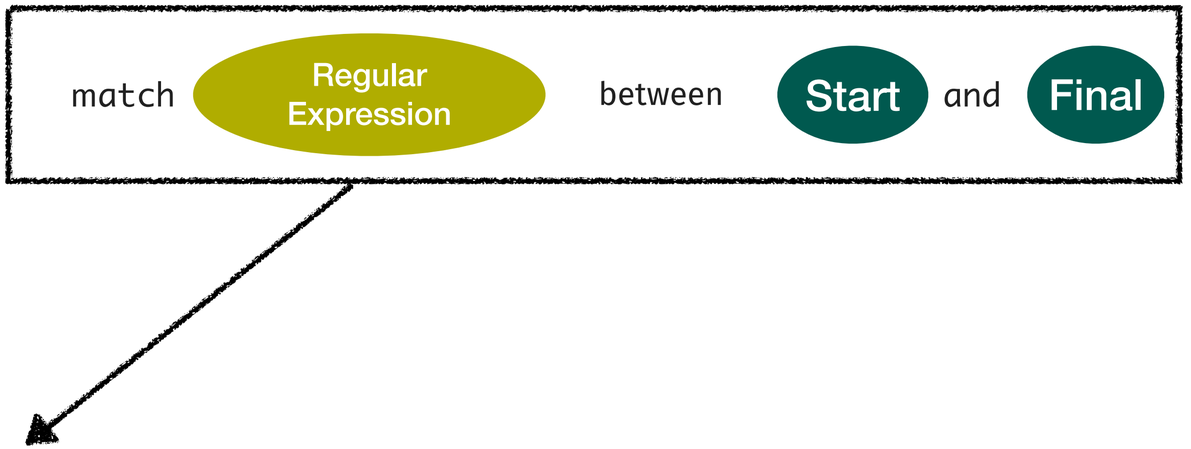
Final Endpoint

Condition on intermediate service interactions

Frontend accesses **Database** after going through some sequence **Data Scrubber**

POLICY LANGUAGE

POLICY LANGUAGE



empty

Only direct access

POLICY LANGUAGE

empty

(Any Endpoint)*

Only direct access

Unrestricted direct/ indirect access

POLICY LANGUAGE

empty

(Any Endpoint)*

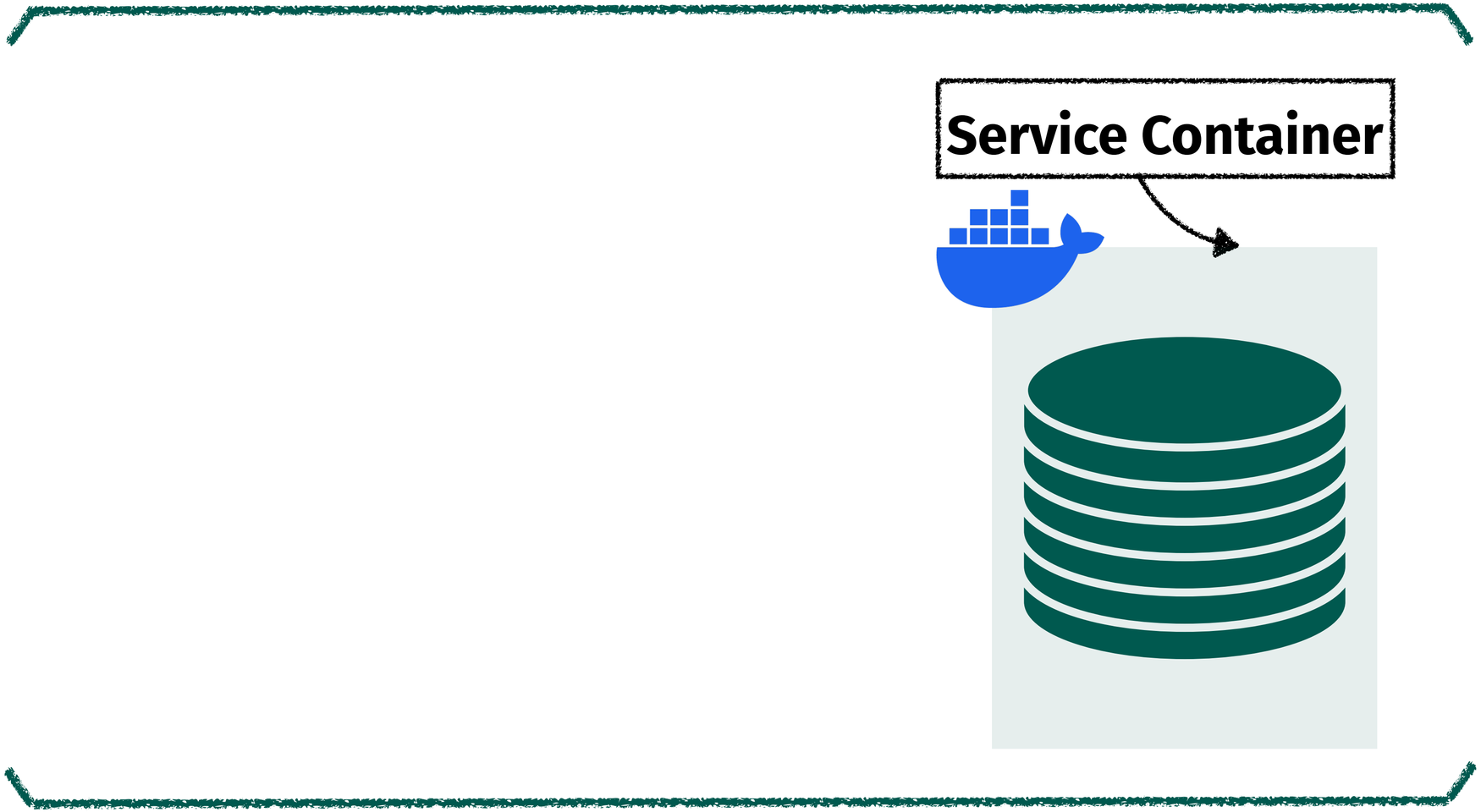
Only direct access

Unrestricted direct/ indirect access

Use cases: regional policy compliance, traffic management scenarios, like A/B testing.

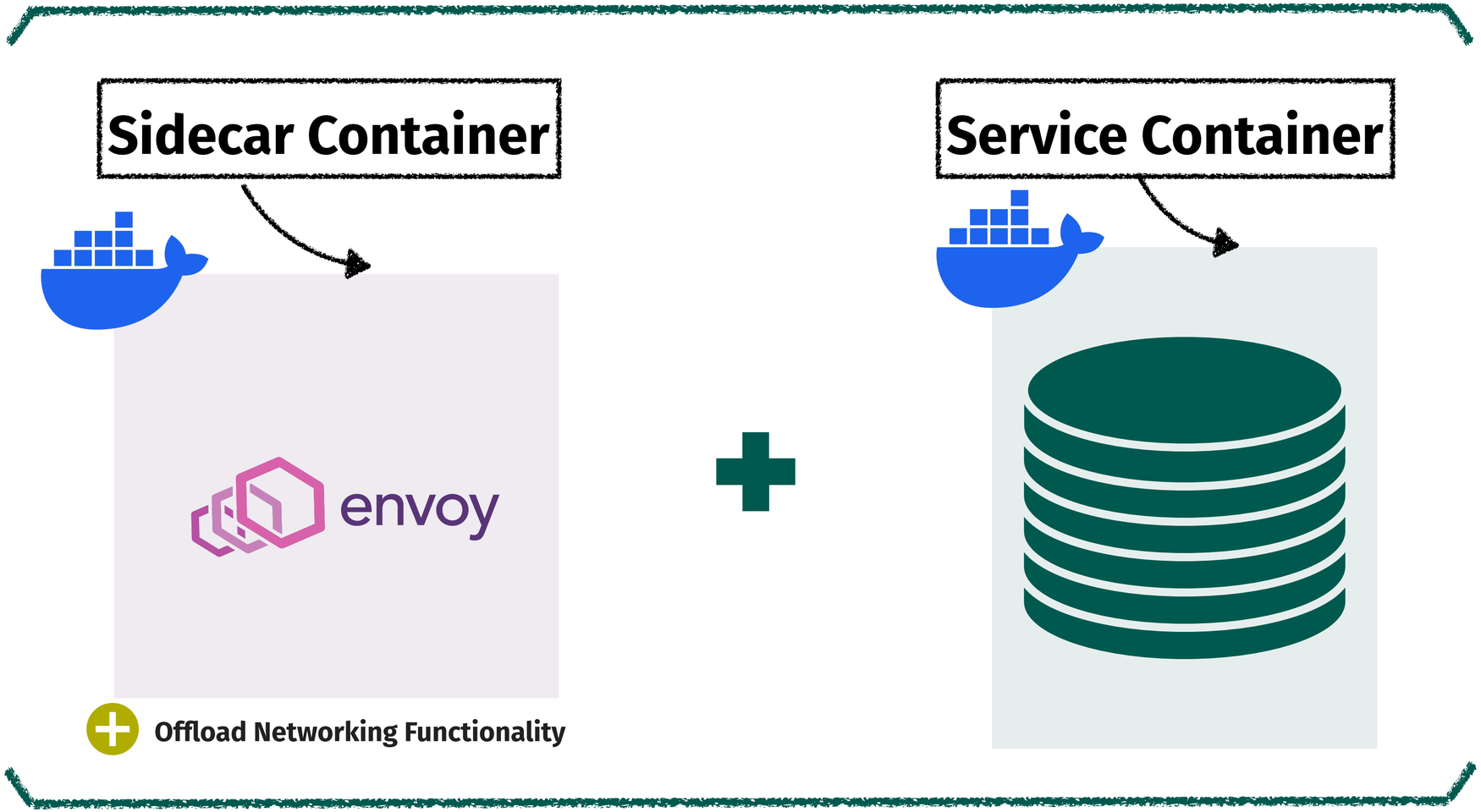
2. Service Layer Enforcement

SERVICE MESH REFRESHER



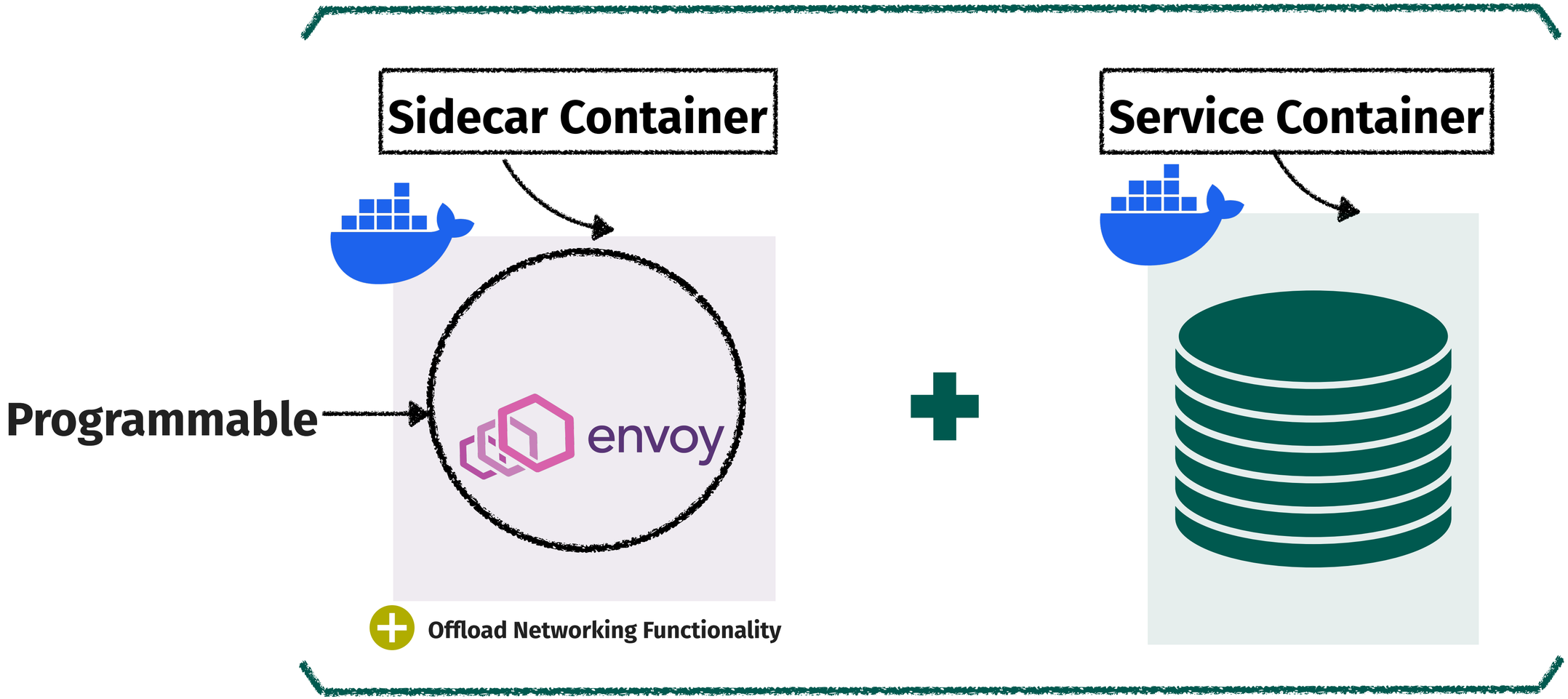
Application on Service Mesh

SERVICE MESH REFRESHER



Application on Service Mesh

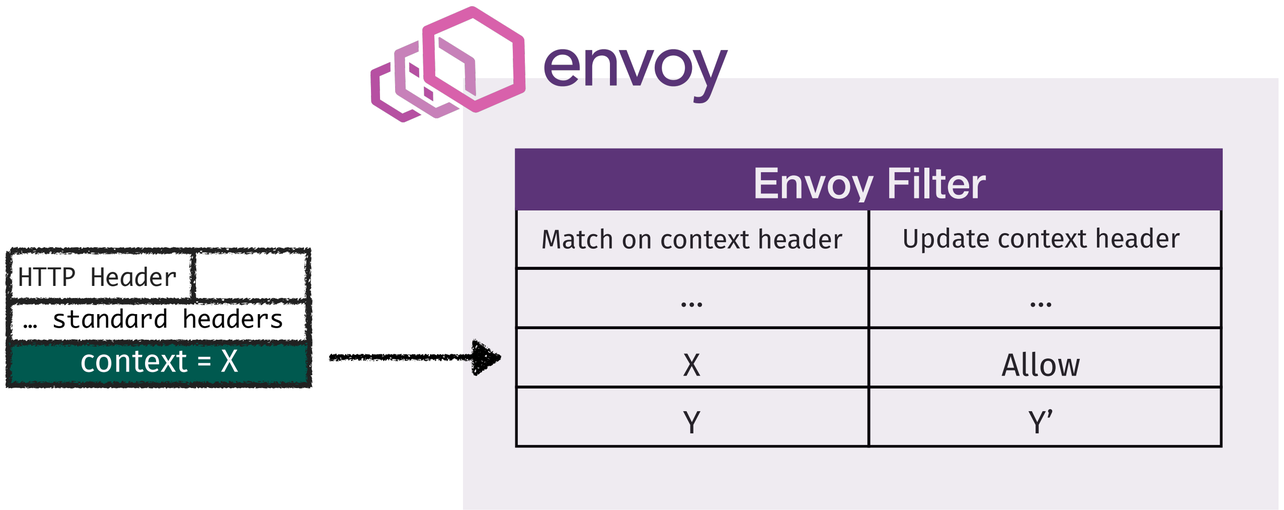
SERVICE MESH REFRESHER

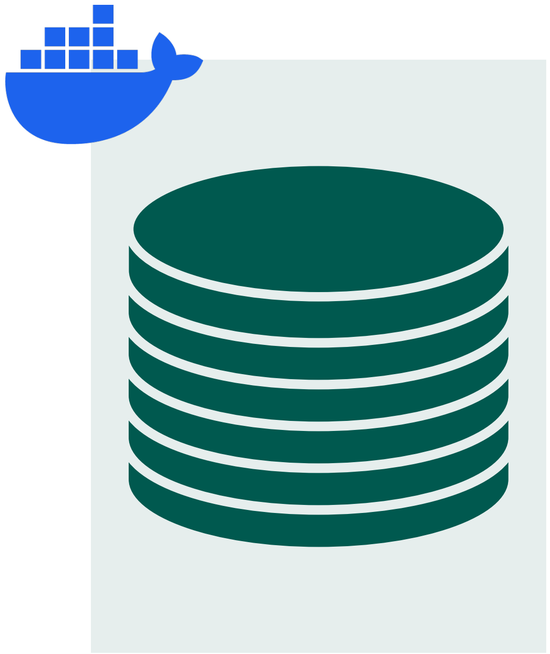


Application on Service Mesh

POLICY CHECKING—ENVOY FILTERS

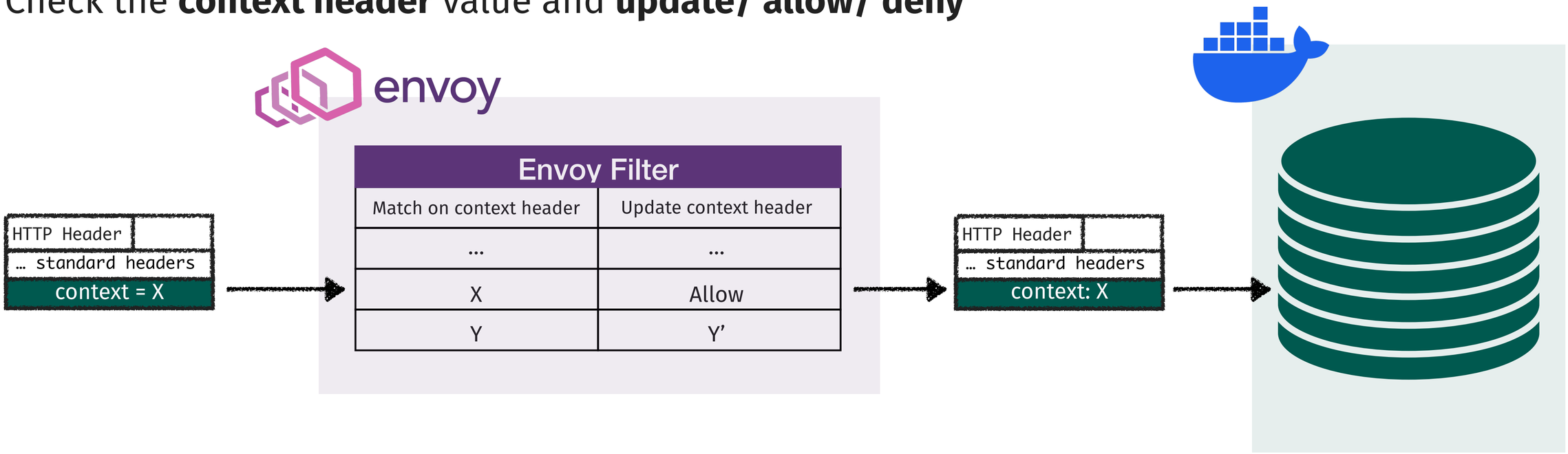
Check the **context header** value and **update/ allow/ deny**





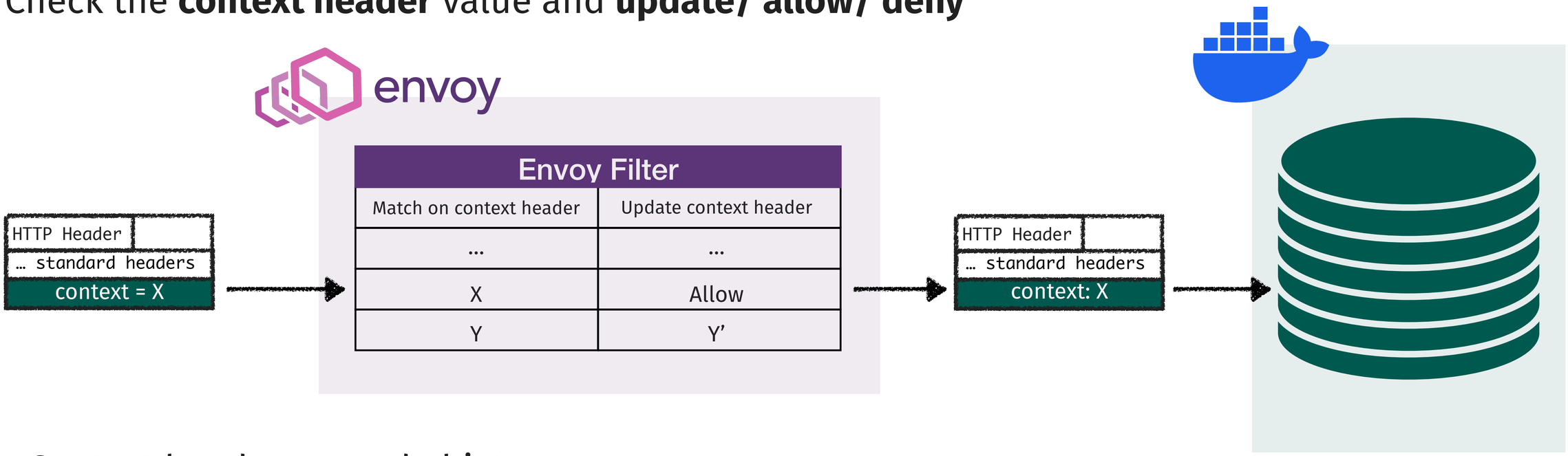
POLICY CHECKING—ENVOY FILTERS

Check the **context header** value and **update/ allow/ deny**



POLICY CHECKING—ENVOY FILTERS

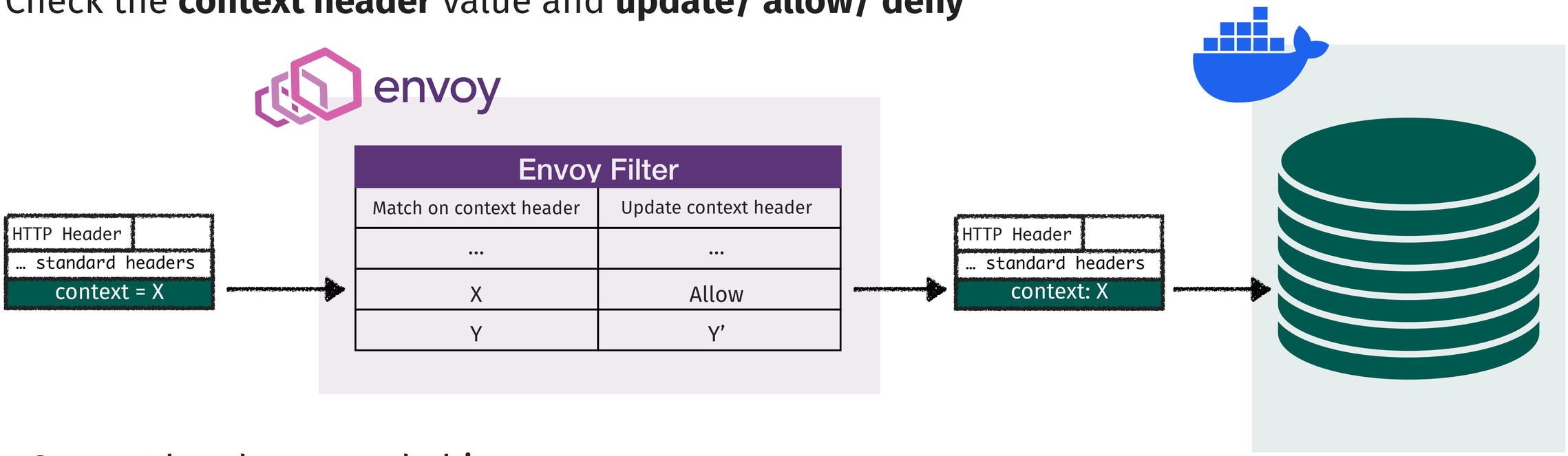
Check the **context header** value and **update/ allow/ deny**



Context headers encode history

POLICY CHECKING—ENVOY FILTERS

Check the **context header** value and **update/ allow/ deny**



Context headers encode history

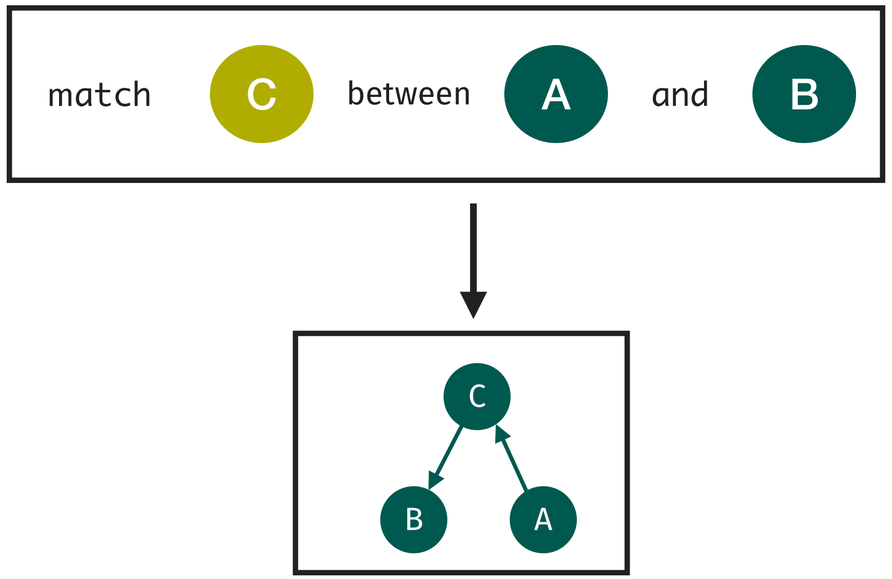
Example context header value:

X = Initiated by Frontend + Data Scrubber happened after Frontend

HOW DO WE WRITE THE FILTERS?

Policy

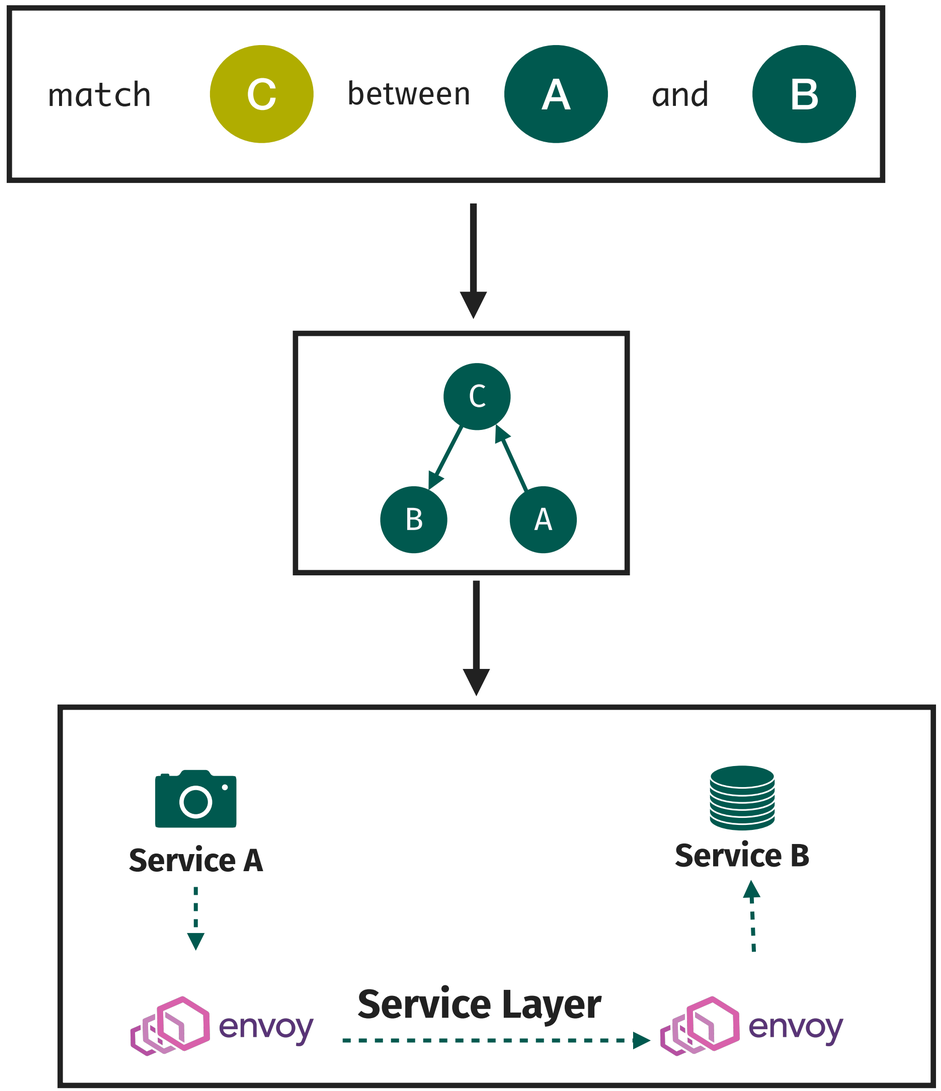
HOW DO WE WRITE THE FILTERS?



Policy

Automata

HOW DO WE WRITE THE FILTERS?



Policy

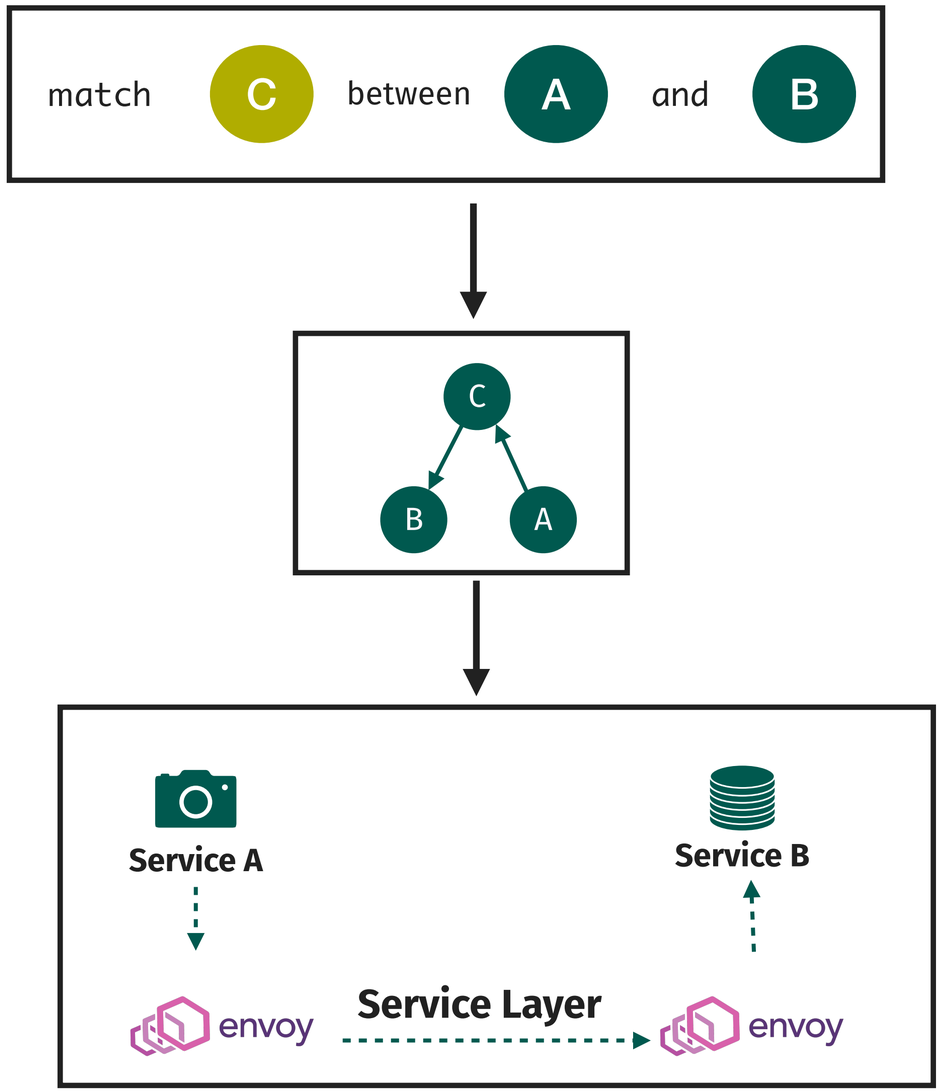
↓

Automata

↓

Envoy Filters

HOW DO WE WRITE THE FILTERS?



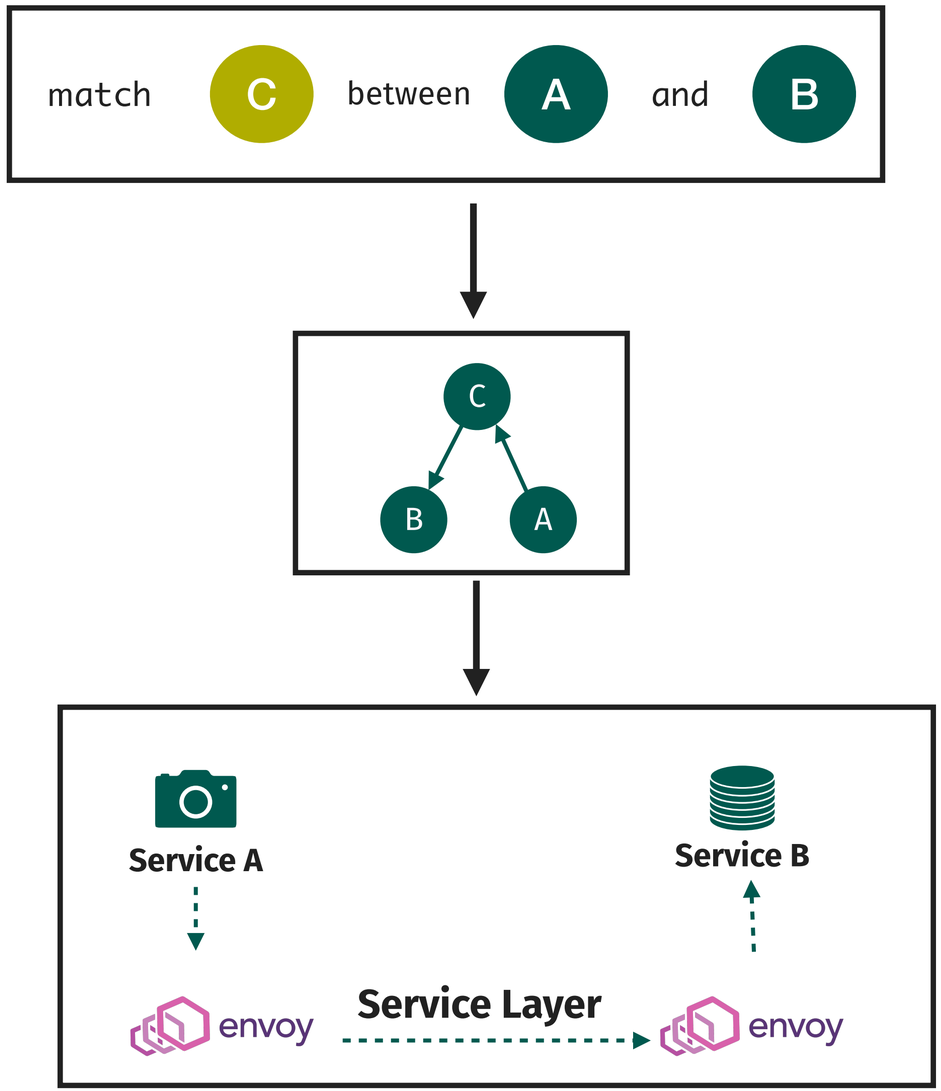
Policy

Automata

Envoy Filters

Automata states = Context header values

HOW DO WE WRITE THE FILTERS?



Policy

Automata

Envoy Filters

Automata states = Context header values

Running the filters is equivalent to running the automata

OPEN QUESTIONS AND ONGOING WORK

Enforcement Mechanism

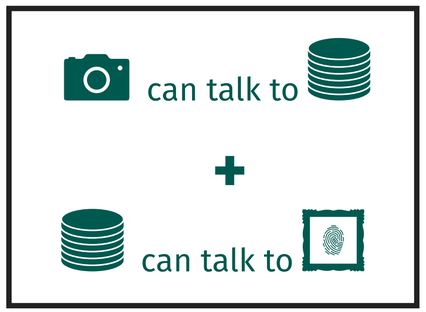
Can we reduce the reliance on trusting the application for context propagation?

Language Design

How to efficiently compose the growing set of policies to avoid blowing up the filters?

Related ongoing work (at UT Austin): Application-tailored Communication with xMesh. NSDI'23 Poster (Saxena et al.)

EXPRESSIVE POLICIES FOR MICROSERVICE NETWORKS



Scope of microservice safety properties

Expressive Declarative Policies

Enforcement at service mesh

More details in the paper!