solo.io

Using Envoy Proxy as your Gateway to Service Mesh

@christianposta

Christian Posta

@christianposta

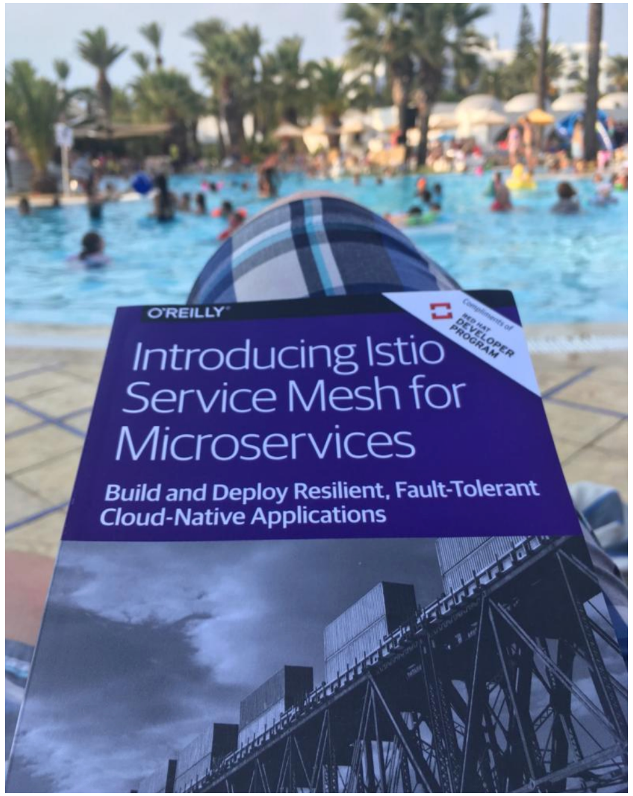
christian@solo.io

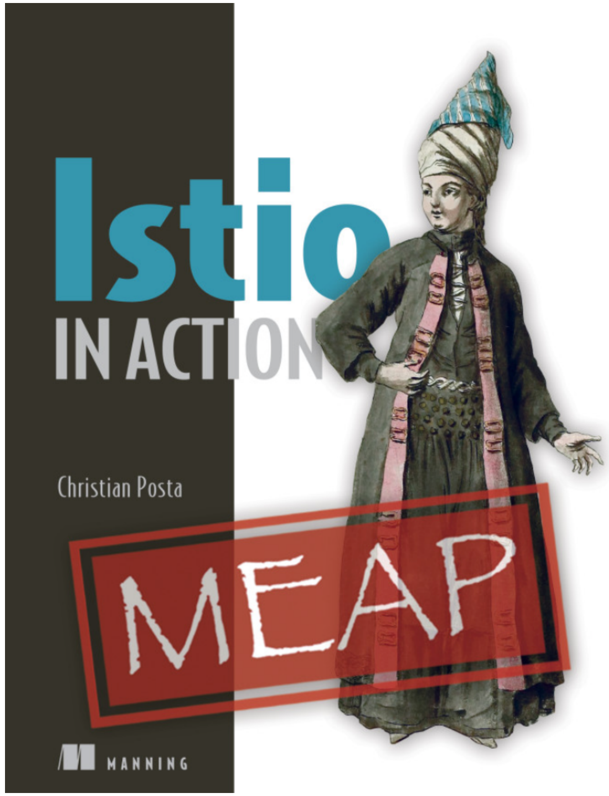
https://blog.christianposta.com

https://slideshare.net/ceposta

- Field CTO @ Solo.io
- Author of a few books
- Contributor to many open-source projects
- Architect, blogger, speaker, mentor, leader

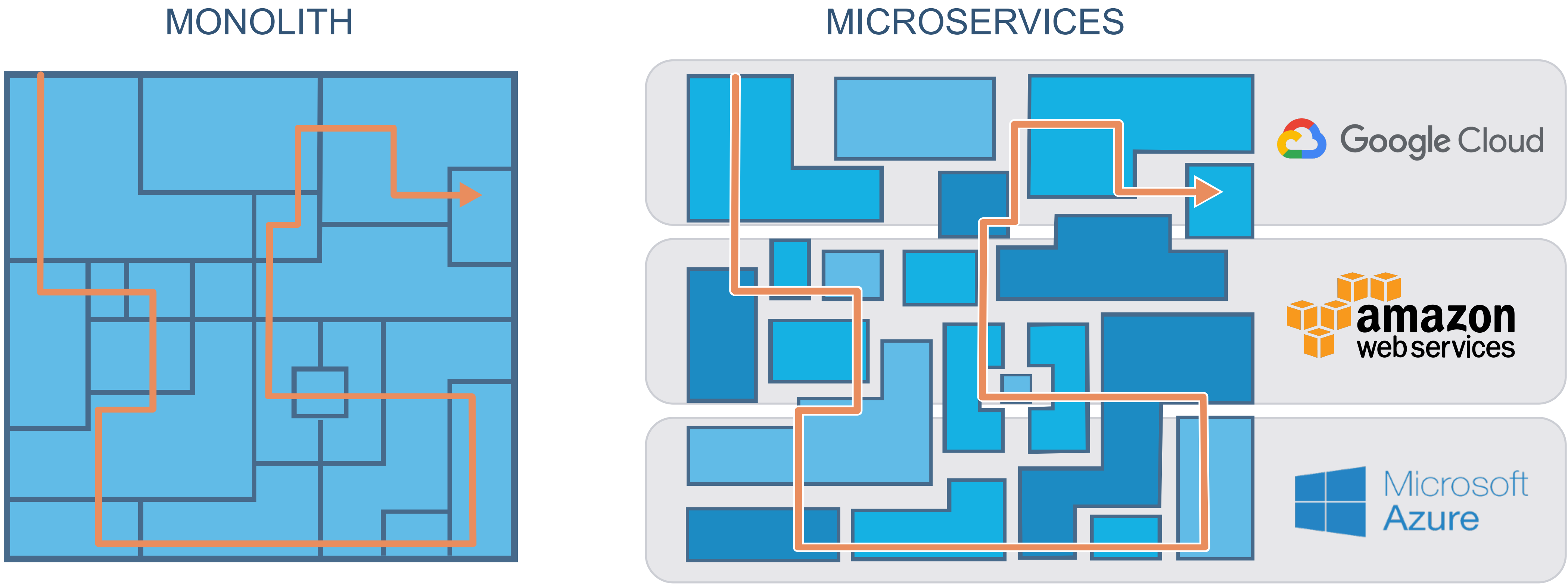
My books


O'REILLY
Compliments of RED HAT DEVELOPER PROGRAM
Introducing Istio Service Mesh for Microservices
Build and Deploy Resilient, Fault-Tolerant Cloud-Native Applications



https://bit.ly/istio-in-action

Why you might be interested?



HOW DO YOU MANAGE **APIs**?

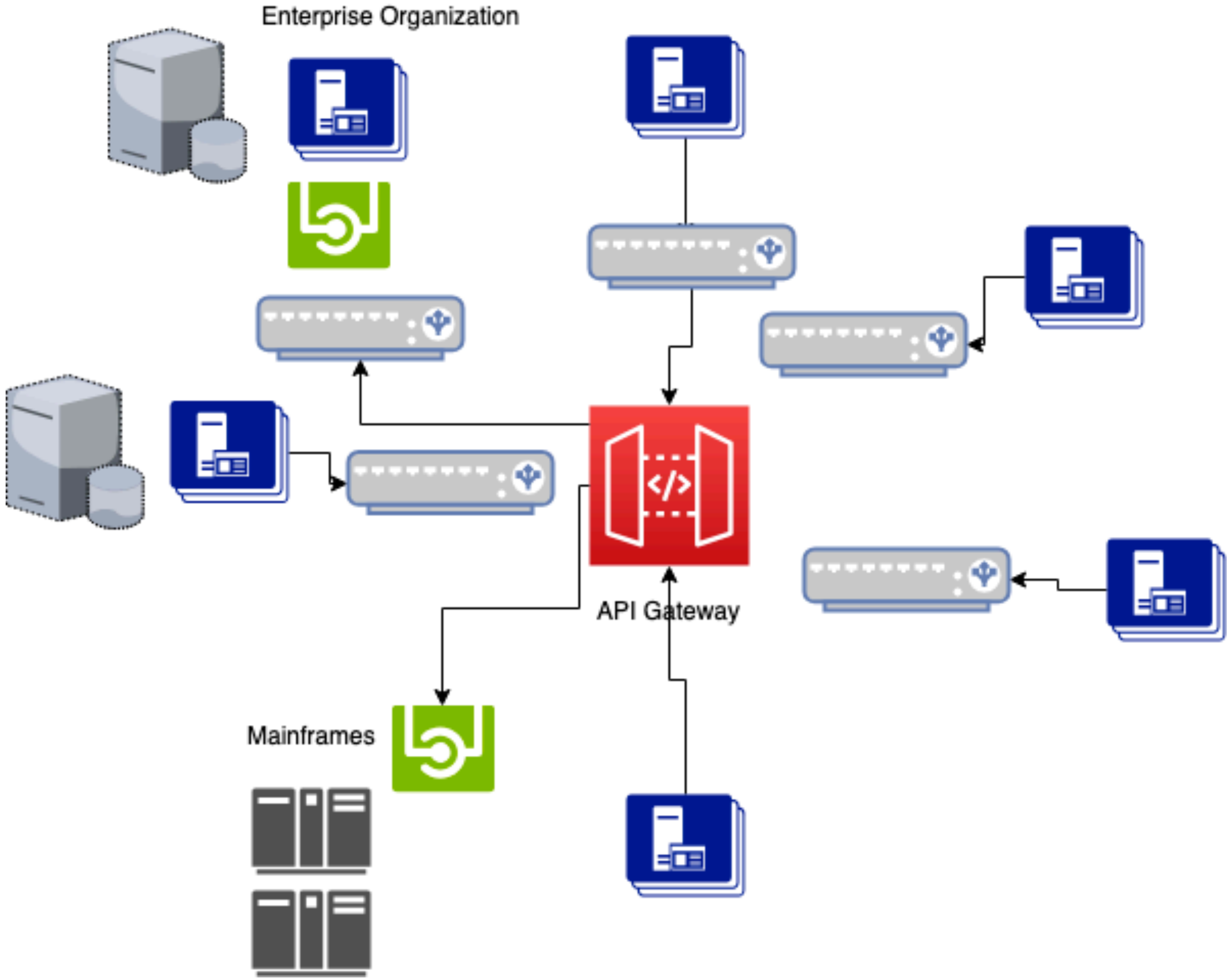
HOW CAN ENFORCE **SECURITY**?

HOW DO YOU **OBSERVE**?

Why you might be interested?

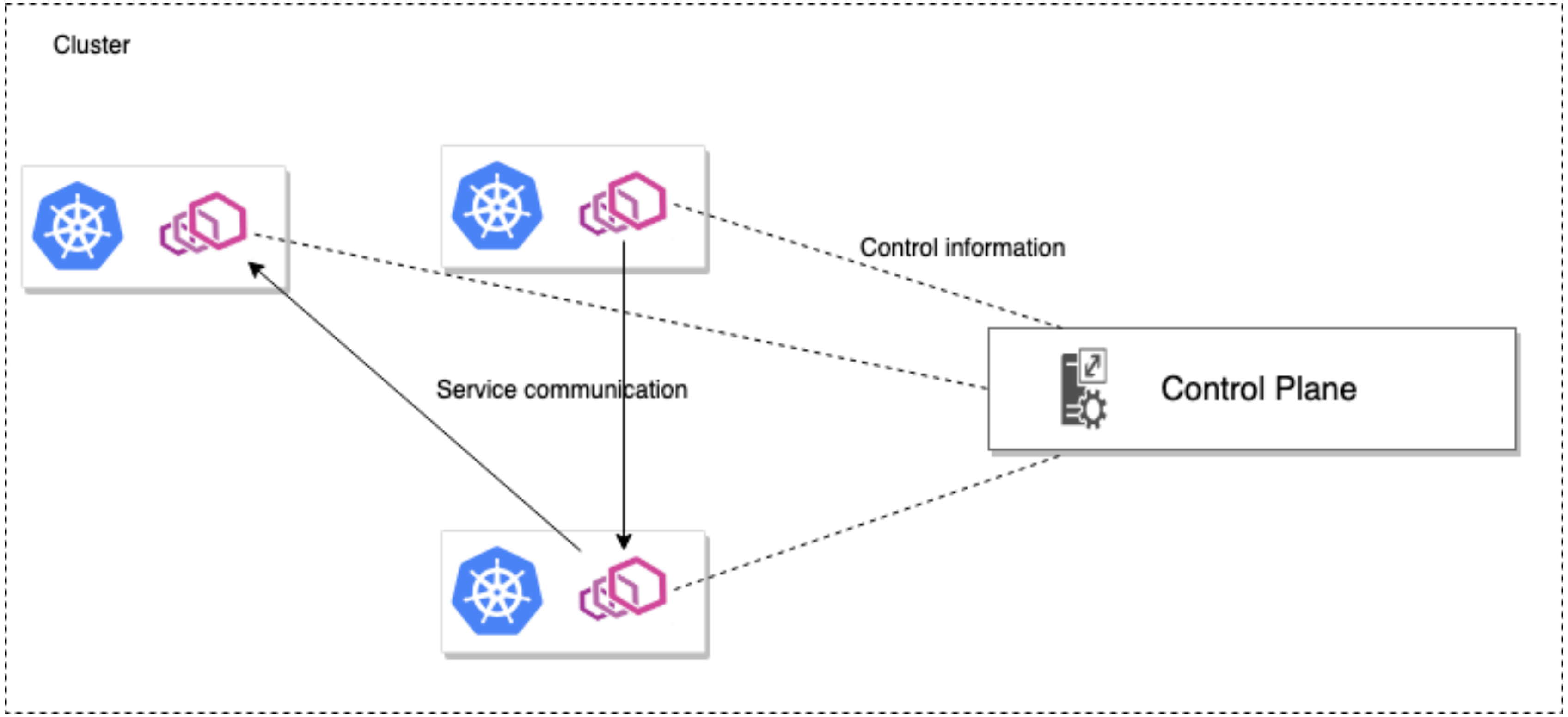
- Augment, complement, replace existing API infrastructure
- Support a microservices, cloud environment
- Need better traffic control and observability
- As little disruption as possible, target multiple compute
- Improve security posture

Why you might be interested?



solo.io

A service mesh to the rescue!



Challenges of adopting a service mesh

Do you need a service mesh?

- Do you have a mix of application languages or frameworks?
- Large deployment of microservices on cloud infrastructure?
- A lot of east/west, service-to-service communication?
- Struggling to implement application network observability?
- Have you mastered your existing infrastructure stack?

Challenges of adoption

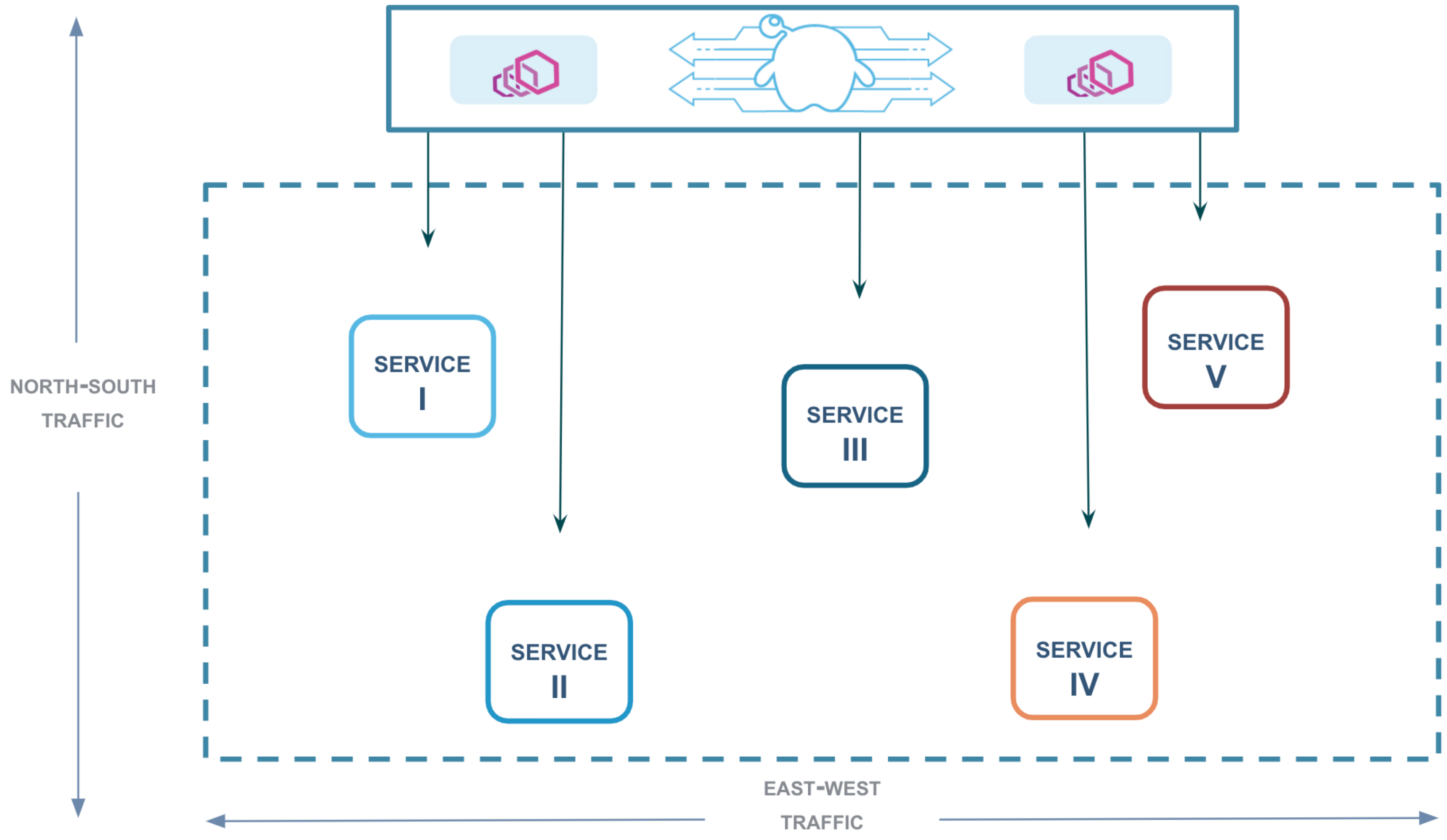
- Which one to choose?
- Who's going to support it?
- Multi-tenancy issues within a single cluster?
- No good way to manage multiple clusters?
- Fitting with existing services (sidecar lifecycle, race conditions, etc)
- What's the delineation between developers and operations?
- Non container environments / hybrid env?
- Centralization vs decentralization

How to get there?

Start with a gateway approach

- Start at the edge
- Start with one proxy, grow to more
- Pick a subset of traffic applications
- Get demonstrable value from it
- Data plane matters
- Pick something that lets you iteratively adopt service mesh

"Edge" concerns, North-South vs East-West



@christianposta

solo.io

“Edge” concerns, North-South vs East-West

Capability	Service Mesh	Edge
Traffic Control		
Traffic Routing		
TLS/mTLS		
Network Observability		
Policy Enforcement		

“Edge” concerns, North-South vs East-West

Capability	Service Mesh	Edge
OAuth/OIDC	✗	
Web Application Firewall	✗	
Message transformation	✗	
Request/response caching	✗	
Domain-specific rate limit	✗	
HMAC, request path security	✗	
Understand API surface, intended decoupling	✗	

Envoy proxy as a gateway

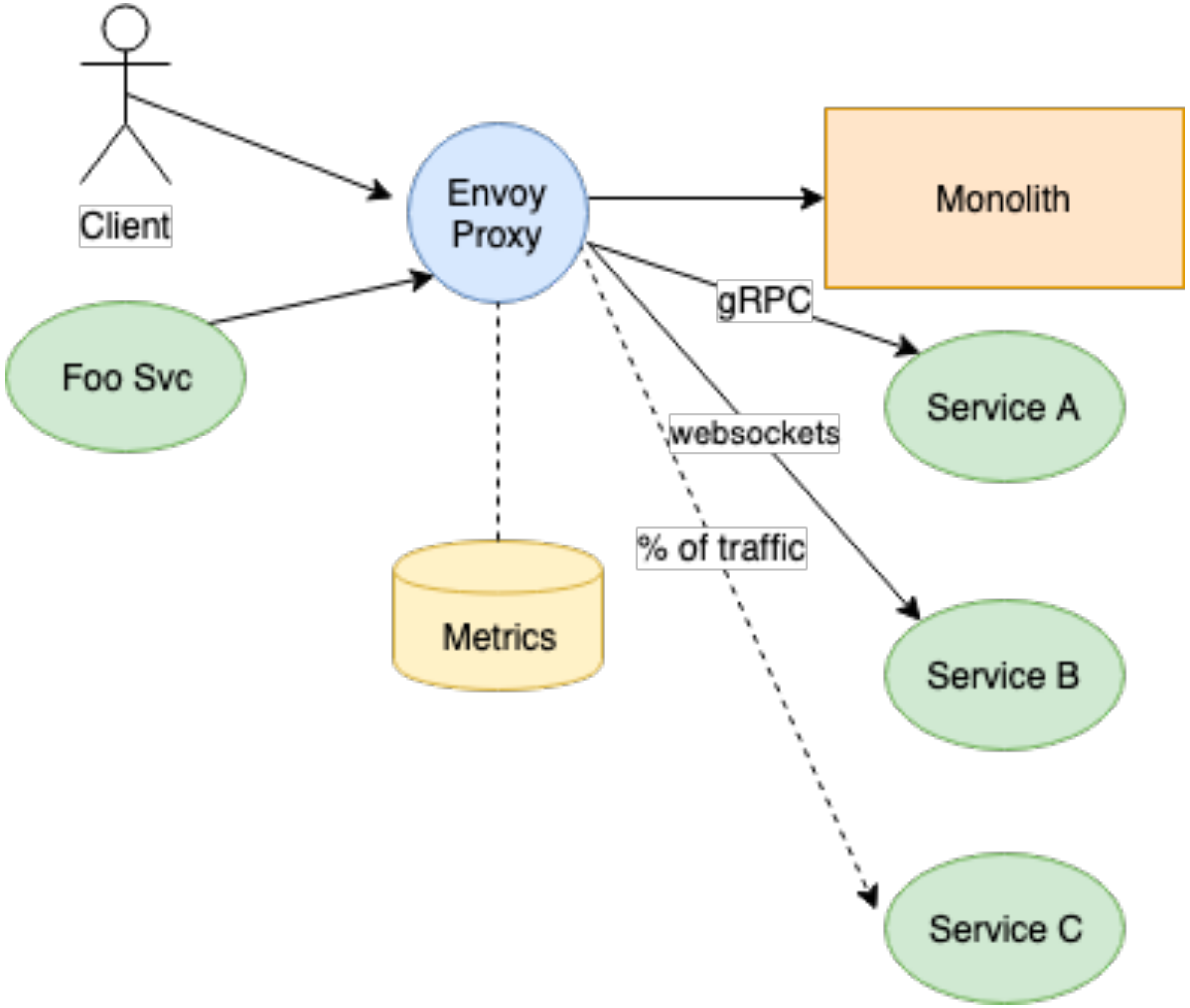
Meet Envoy Proxy

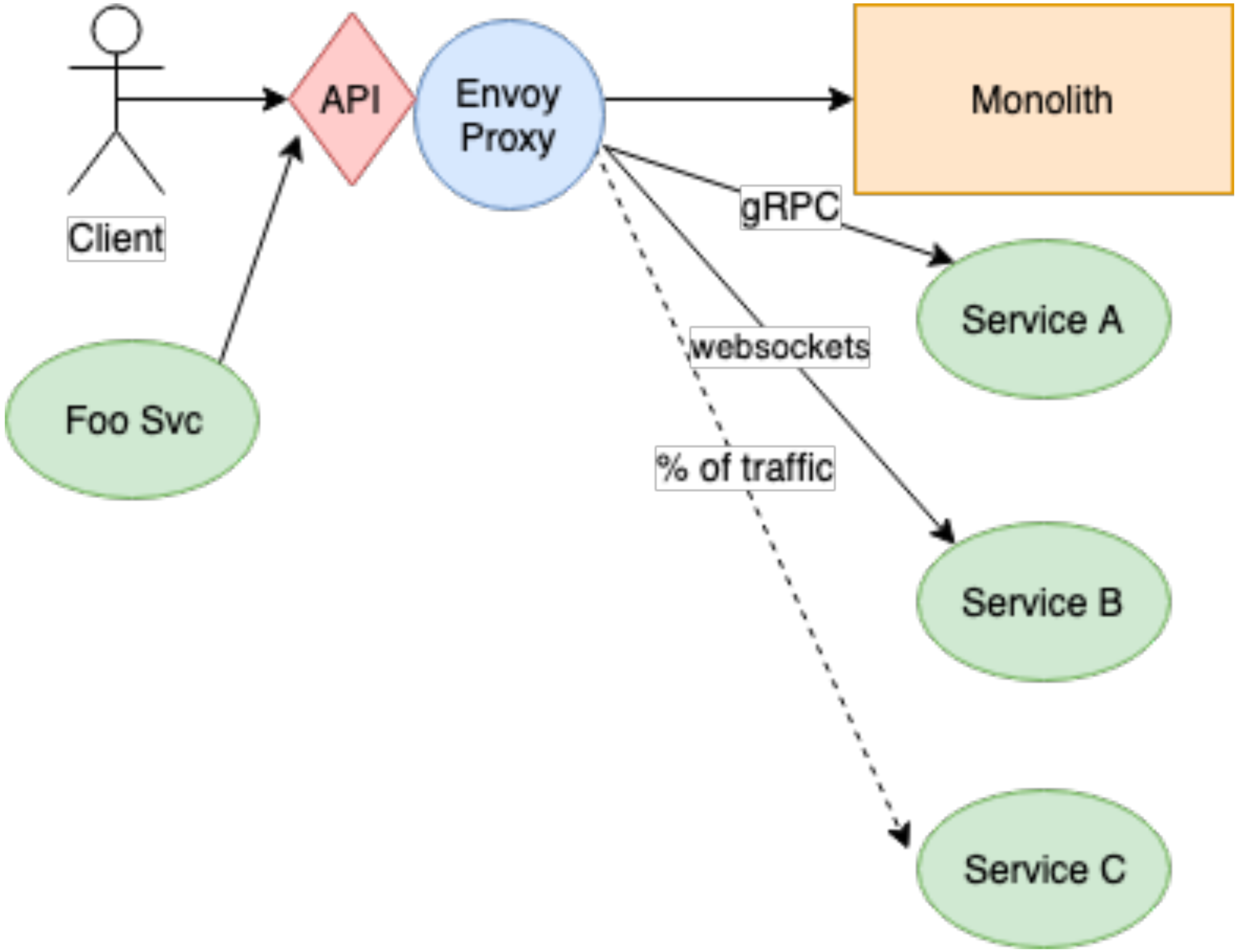
http://envoyproxy.io

ENVOY IS AN OPEN SOURCE EDGE AND SERVICE PROXY, DESIGNED FOR CLOUD-NATIVE APPLICATIONS

Envoy Proxy implements:

- zone aware, least request load balancing
- circuit breaking
- outlier detection
- retries, retry policies
- timeout (including budgets)
- traffic shadowing
- rate limiting
- access logging, statistics collection
- Many other features!


Client
Envoy Proxy
Monolith
Foo Svc
gRPC
Service A
websockets
Metrics
% of traffic
Service B
Service C


Client
API
Envoy Proxy
Monolith
gRPC
Service A
websockets
% of traffic
Service B
Service C
Foo Svc

Open source edge gateway built on Envoy

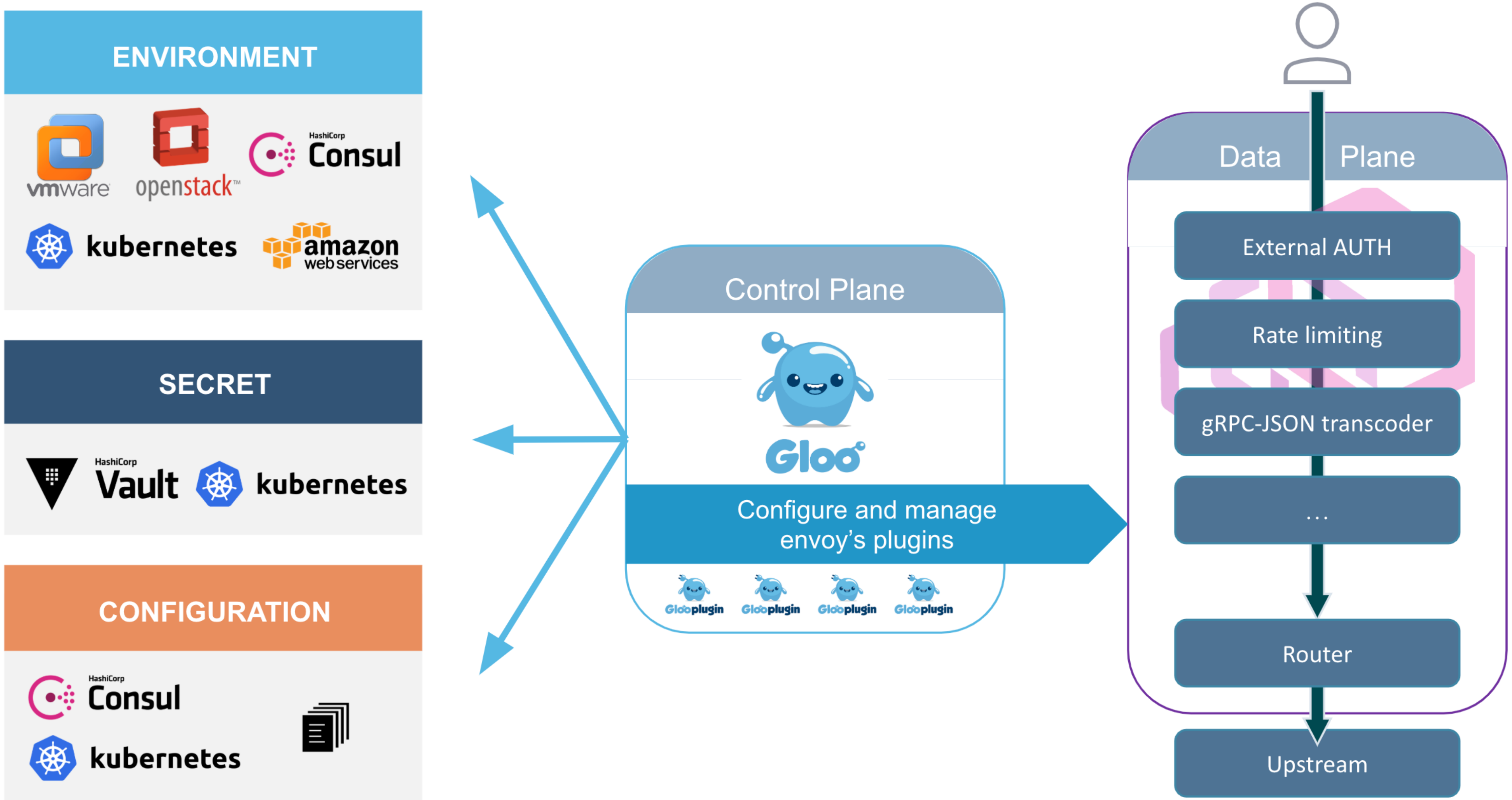
https://github.com/solo-io/gloo

What is Gloo?

- Enterprise Envoy Proxy
- API-level routing, decoupling
- Complements any service mesh
- Traffic control, canary releases
- OAuth flows
- TLS termination, passthrough, mTLS
- Rate limiting, Caching
- Request/Response transformation
- Kubernetes CRDs (when deployed to Kubernetes)

https://gloo.solo.io

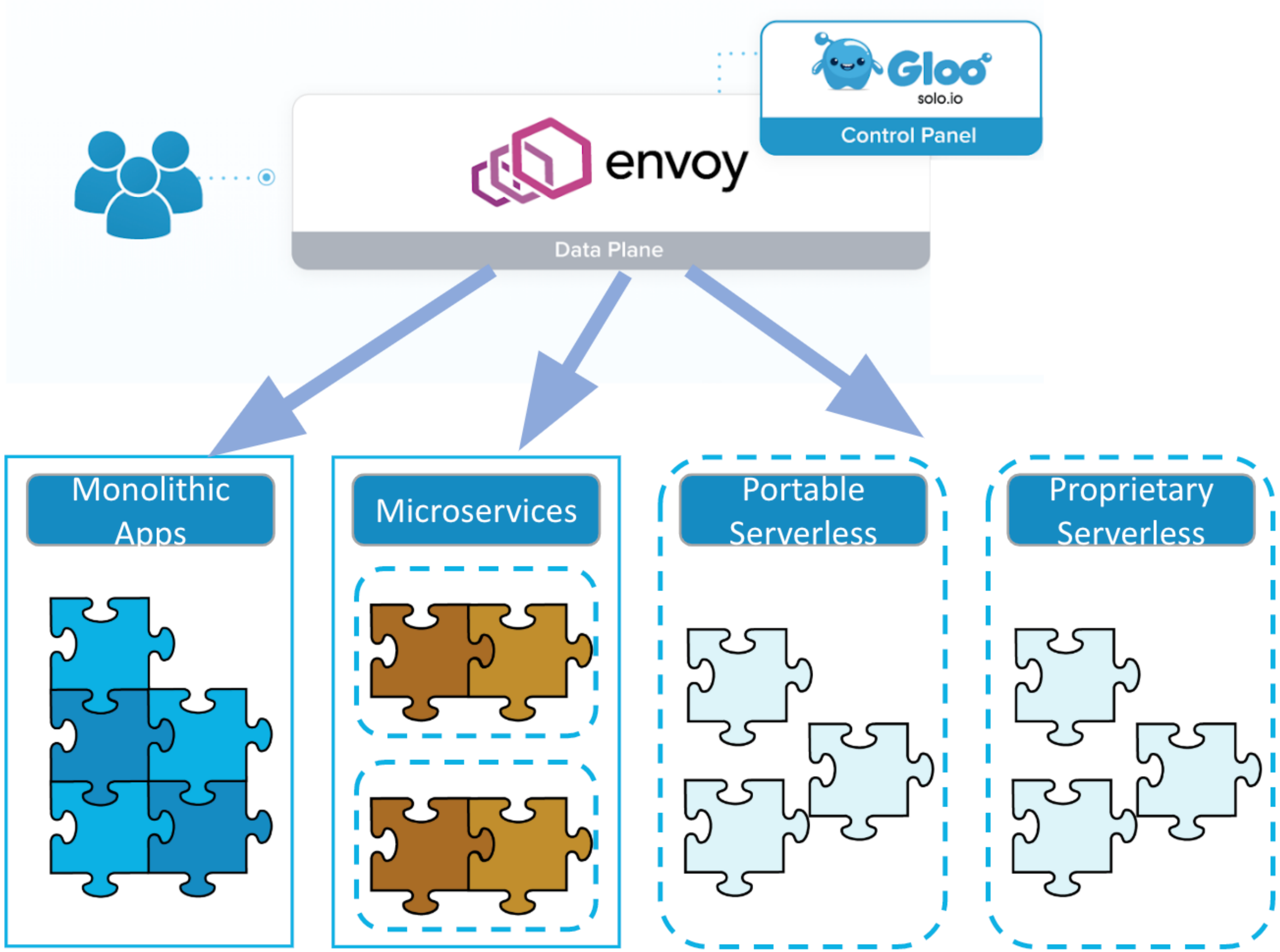
Edge Gateway built on Envoy



Gloo adds these to service mesh!

Capability	Service Mesh	Edge
OAuth/OIDC	✗	
Web Application Firewall	✗	
Message transformation	✗	
Request/response caching	✗	
Domain-specific rate limit	✗	
HMAC, request path security	✗	
Understand API surface, intended decoupling	✗	

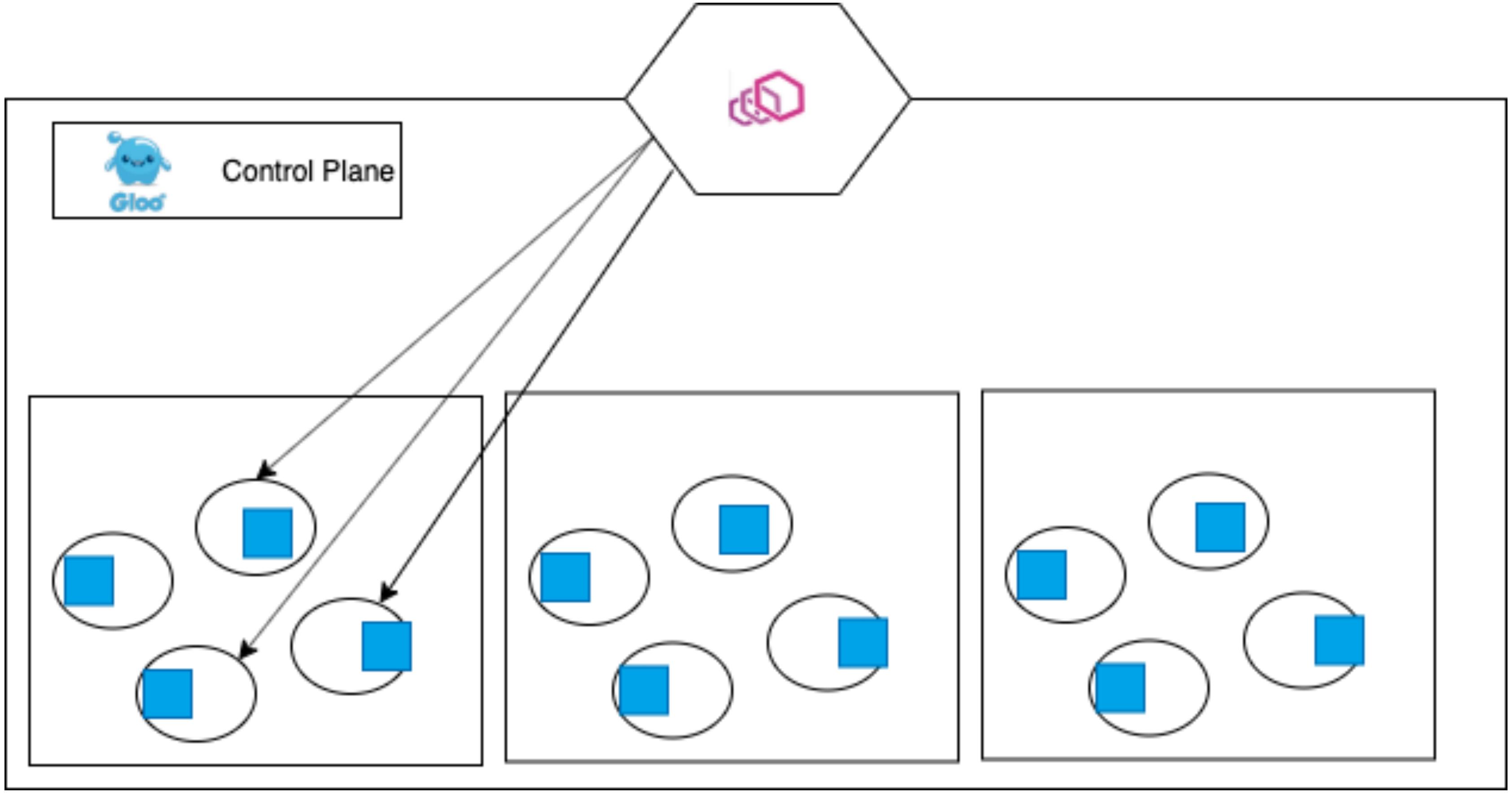
Demo!



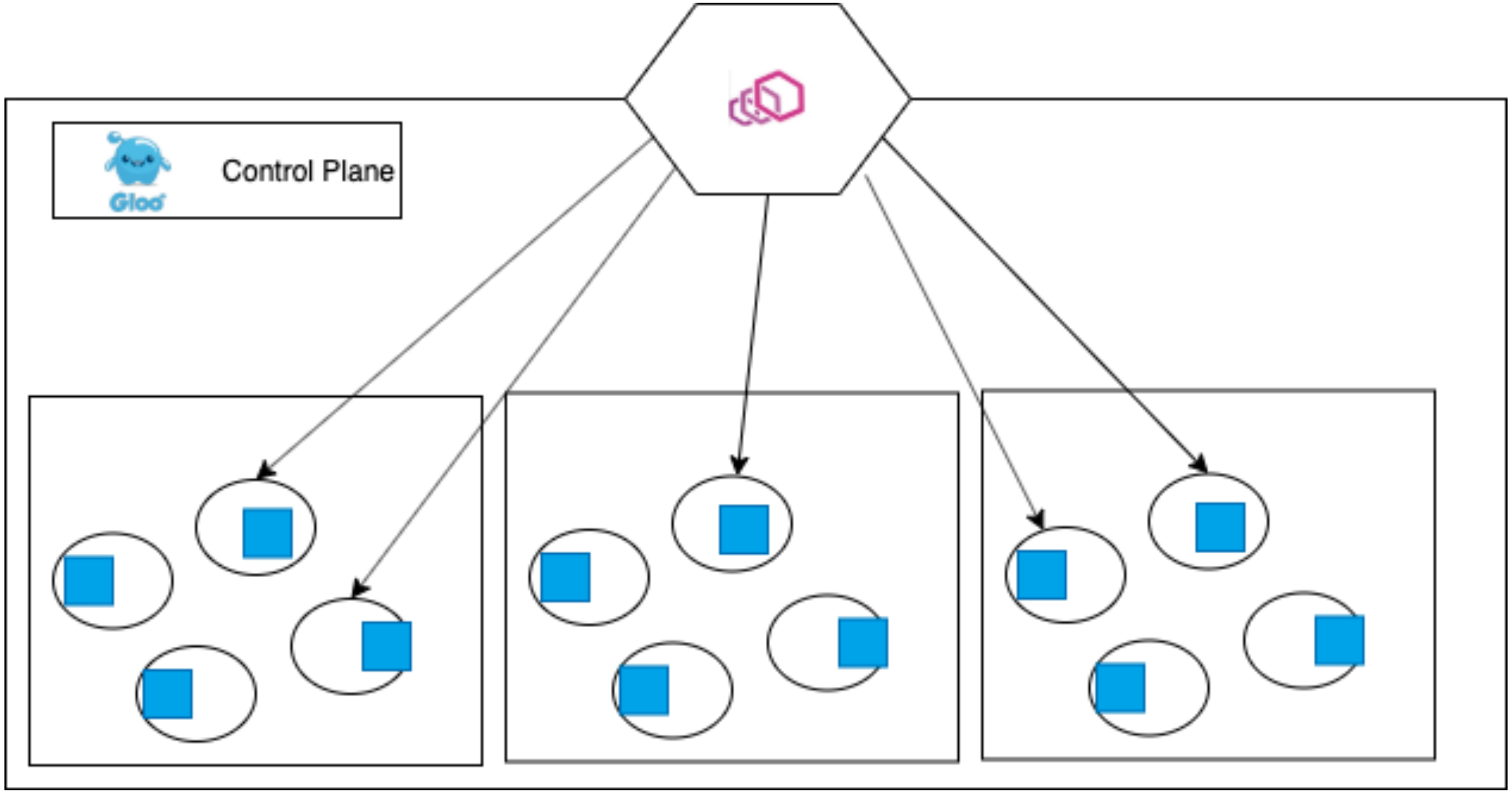
@christianposta

Gateway adoption patterns (waypoint architecture) on the journey to service mesh

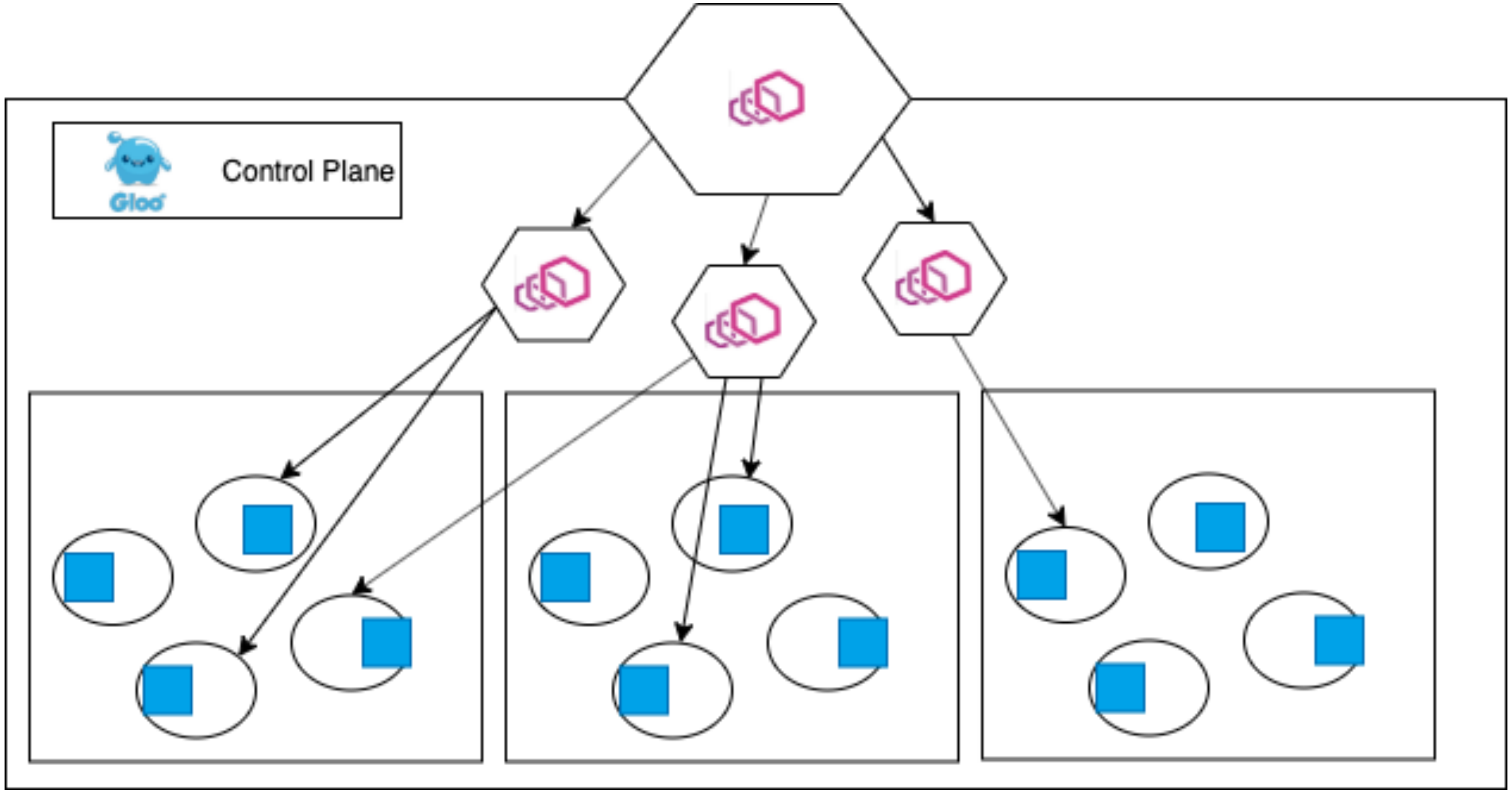
Start with single proxy



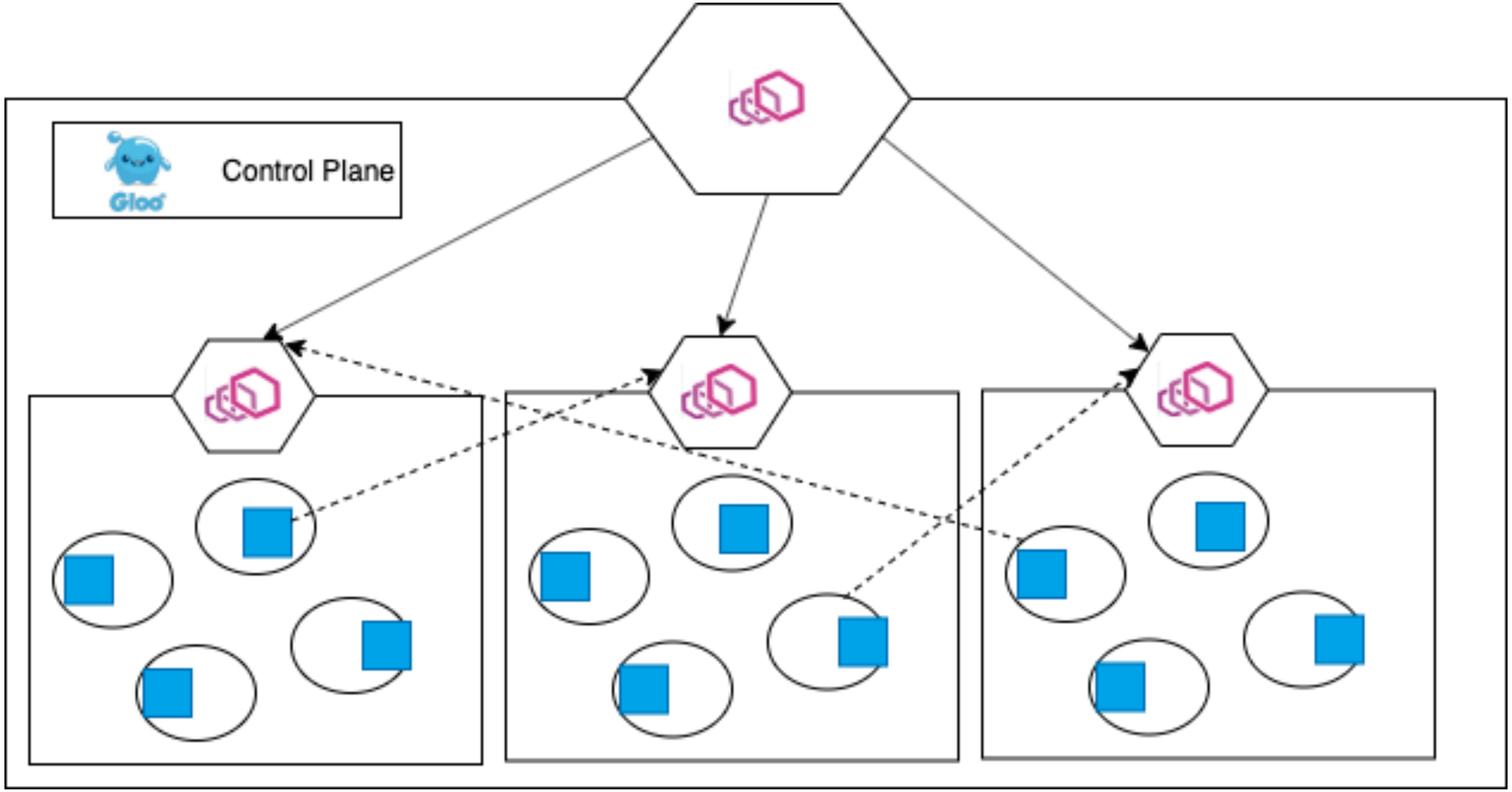
Start with single proxy



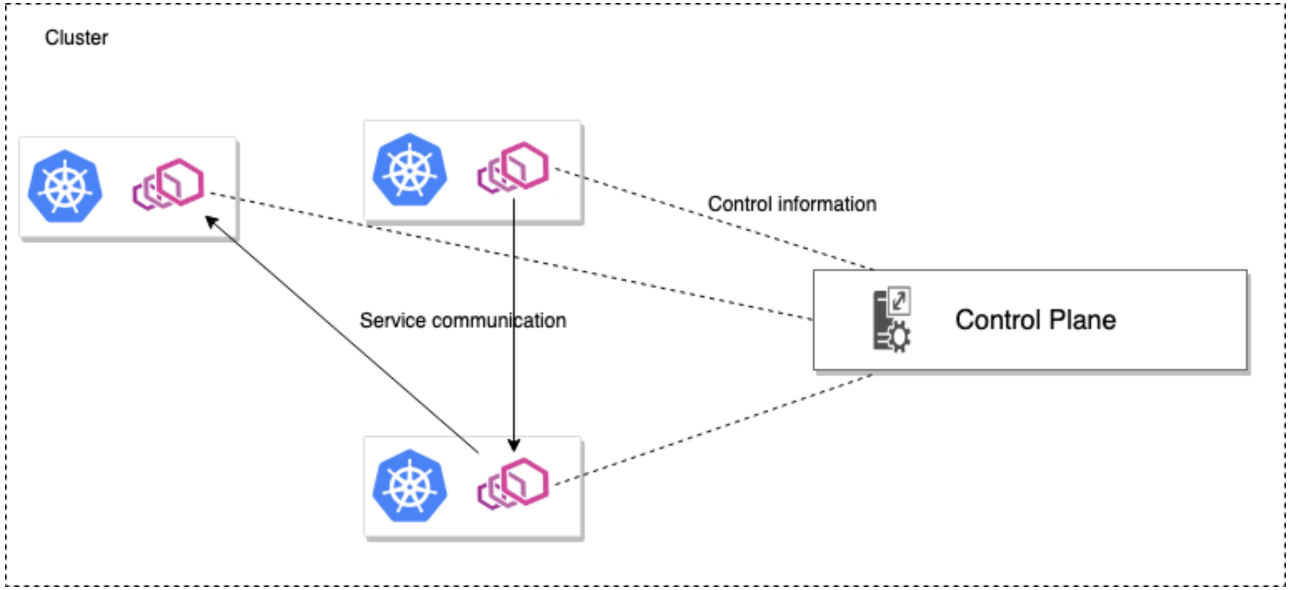
Bring in decoupling points (multi-tier gateway)

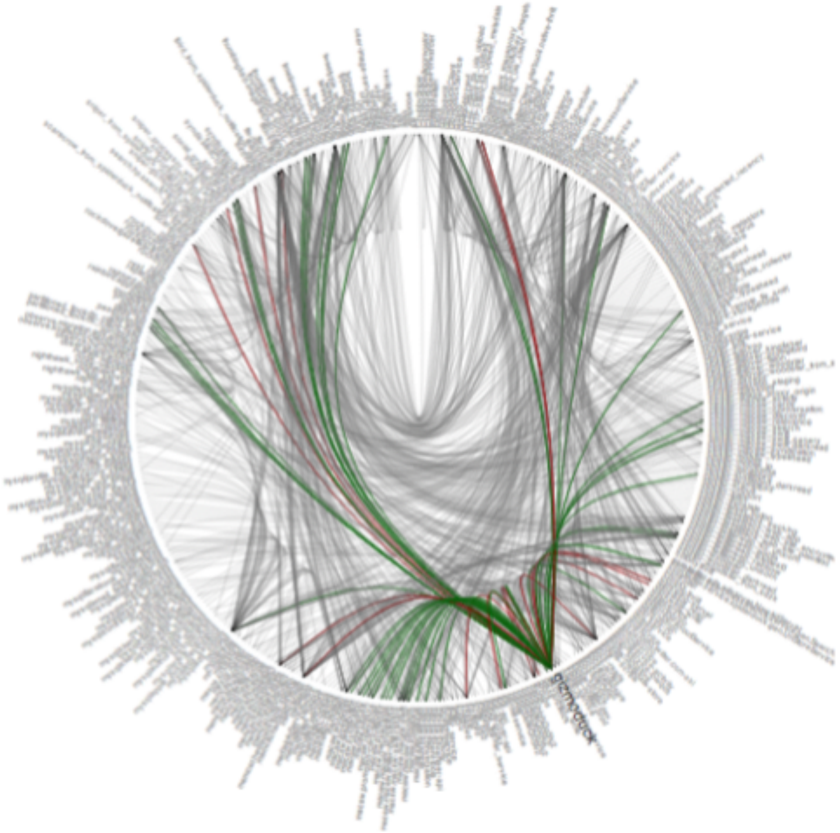


Gateway per product/domain/bounded context

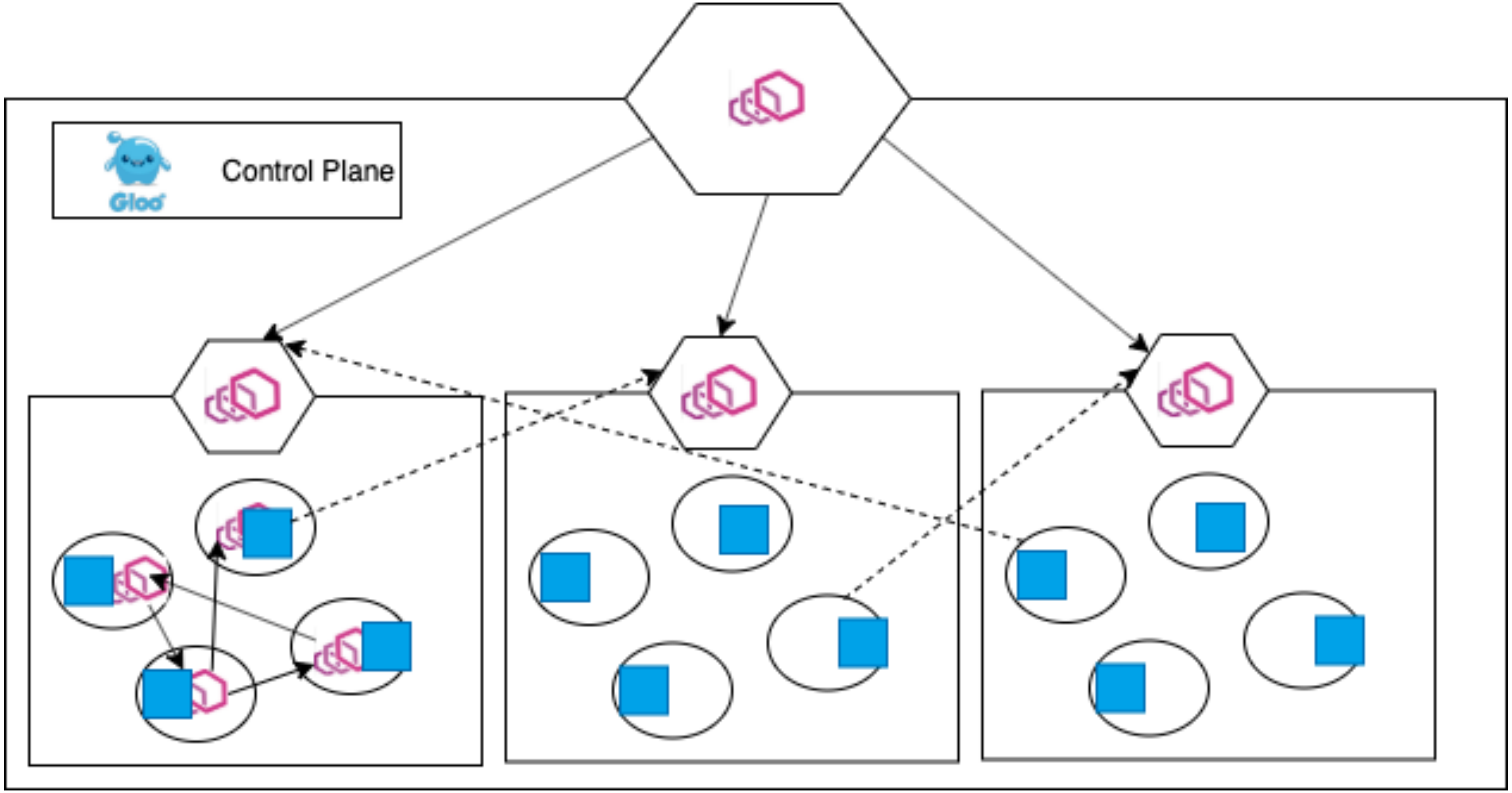


Push gateways down as you grow, avoid death star architecture!

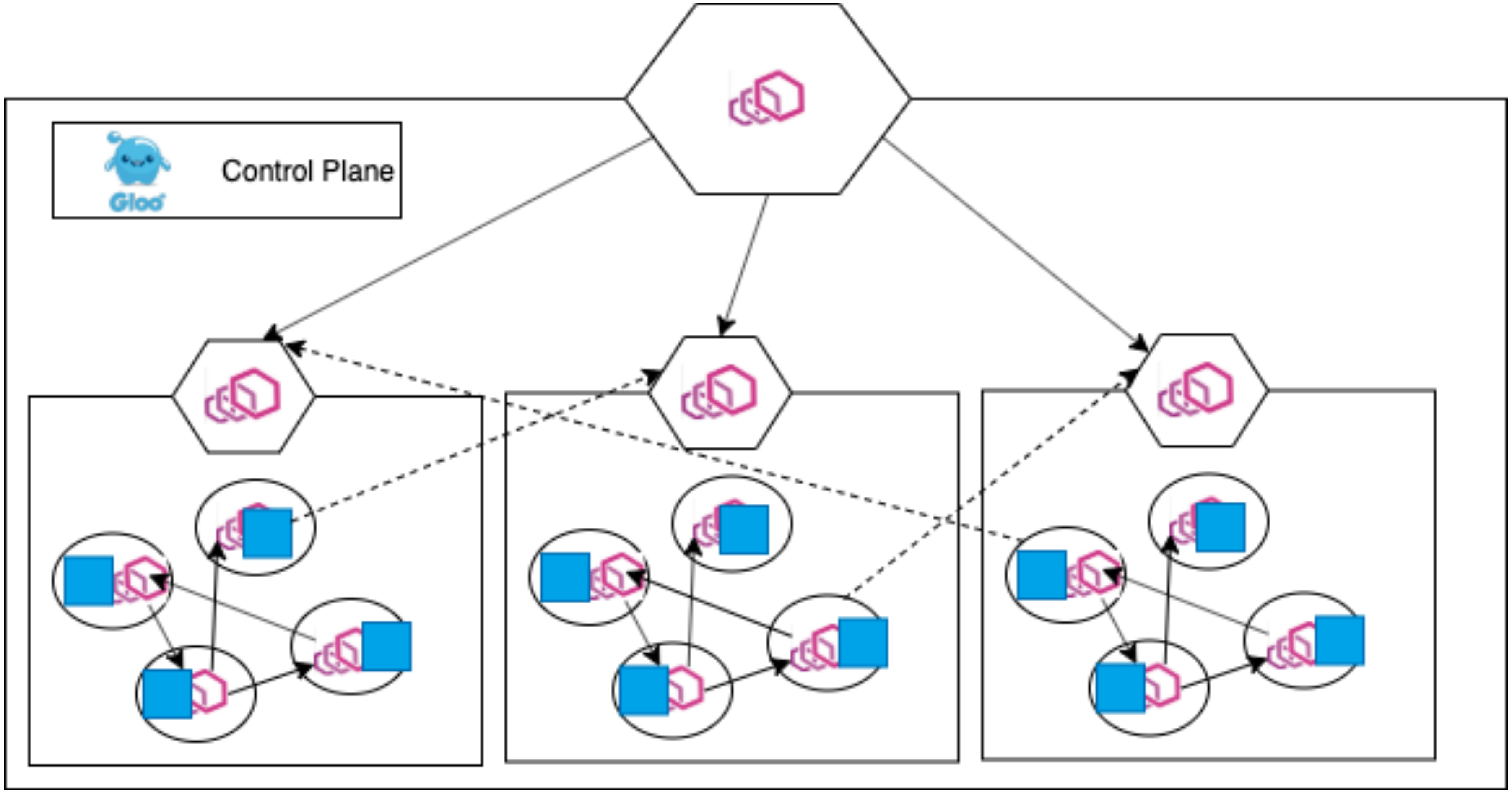




Push gateways down as you grow, avoid death star architecture!



Push gateways down as you grow, avoid death star architecture!



Solo.io projects:

Gloo	https://docs.solo.io/gloo/latest/
Gloo Enterprise	https://www.solo.io/products/gloo/
Service Mesh Hub	http://servicemeshhub.io
Autopilot	https://docs.solo.io/autopilot/latest/
Squash Debugger	https://squash.solo.io

The Service Mesh Journey with Solo.io

Solo.io connects and manages the world's applications with APIs and service mesh across any infrastructure.

solo.io

@soloio_inc