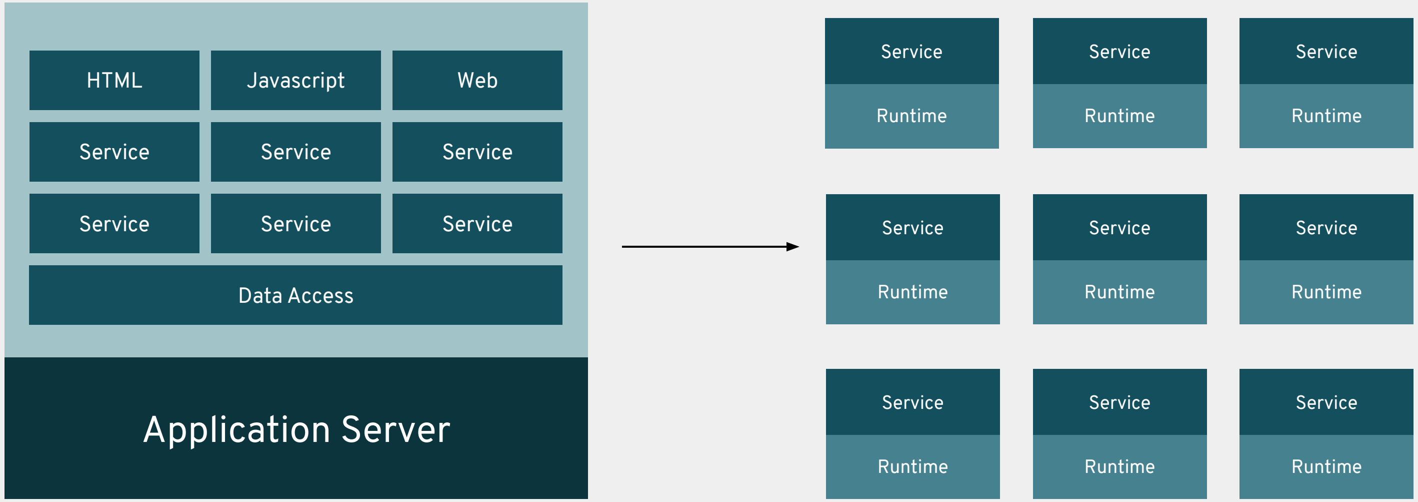
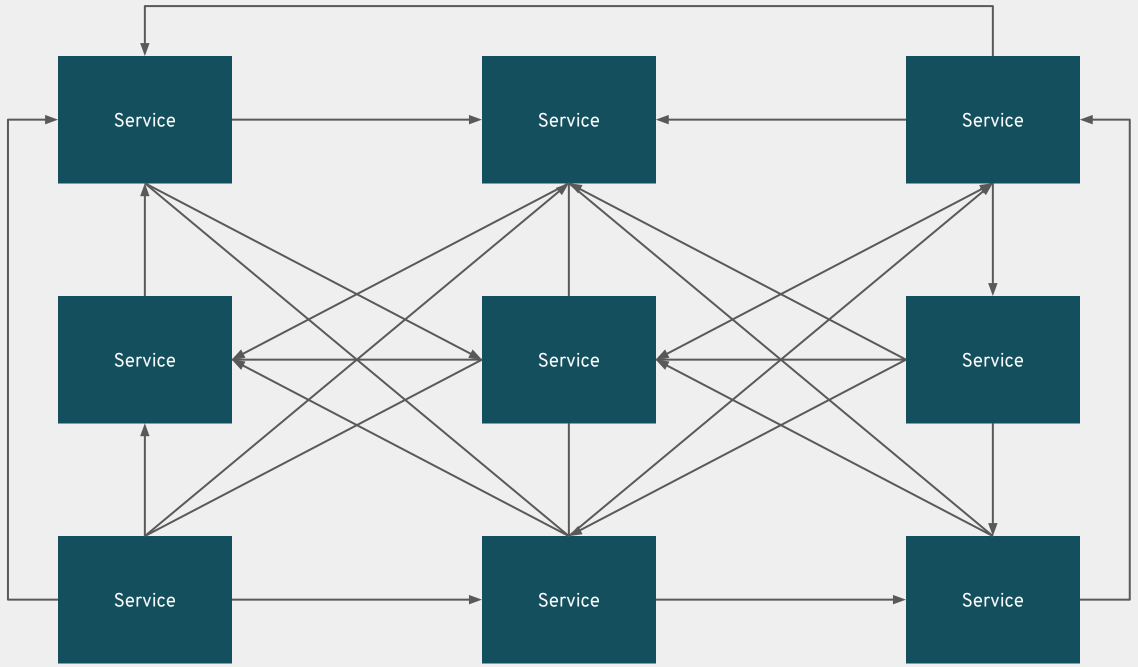
OpenShift 4.x Architecture Workshop

Red Hat Service Mesh - Istio

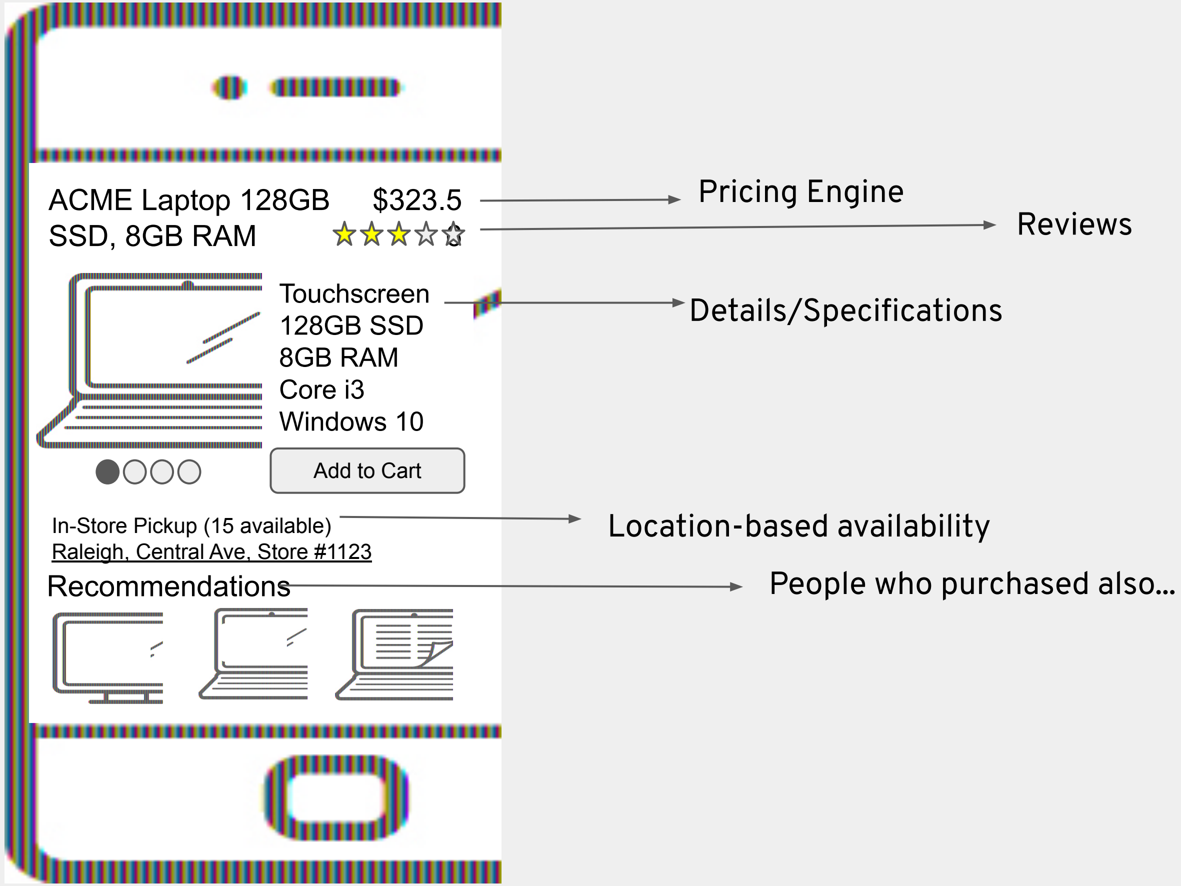
~~MICROSERVICES~~ ARCHITECTURE
DISTRIBUTED



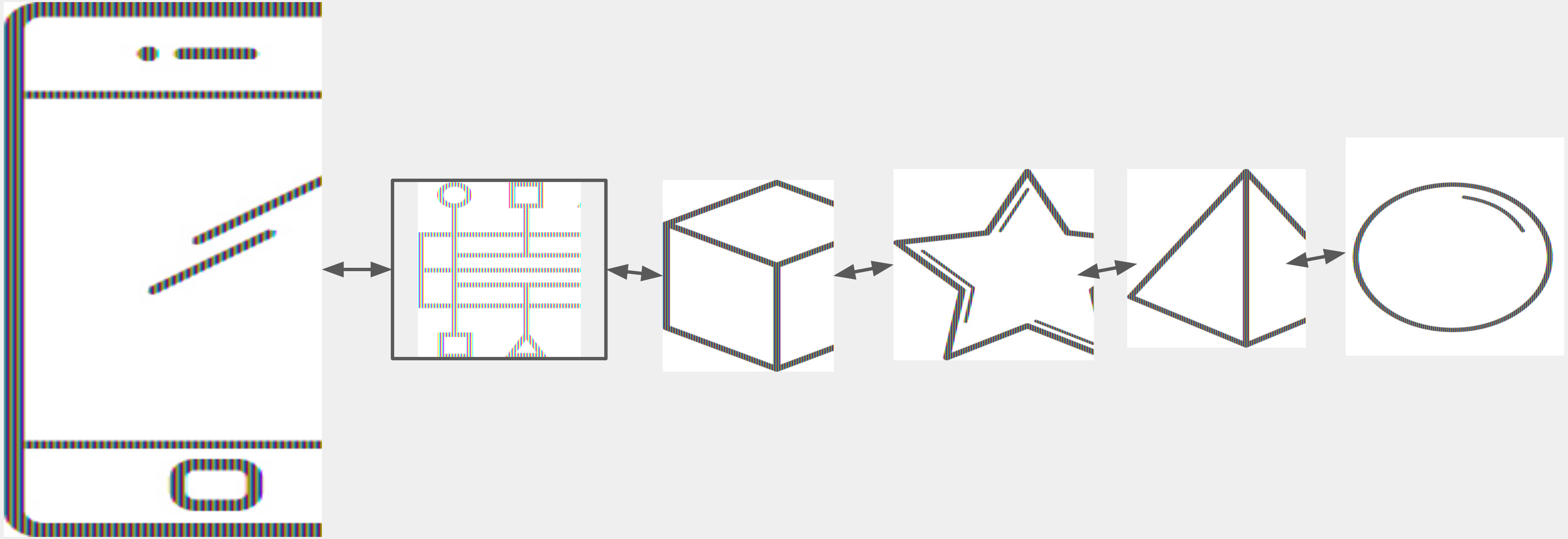
DISTRIBUTED ARCHITECTURE



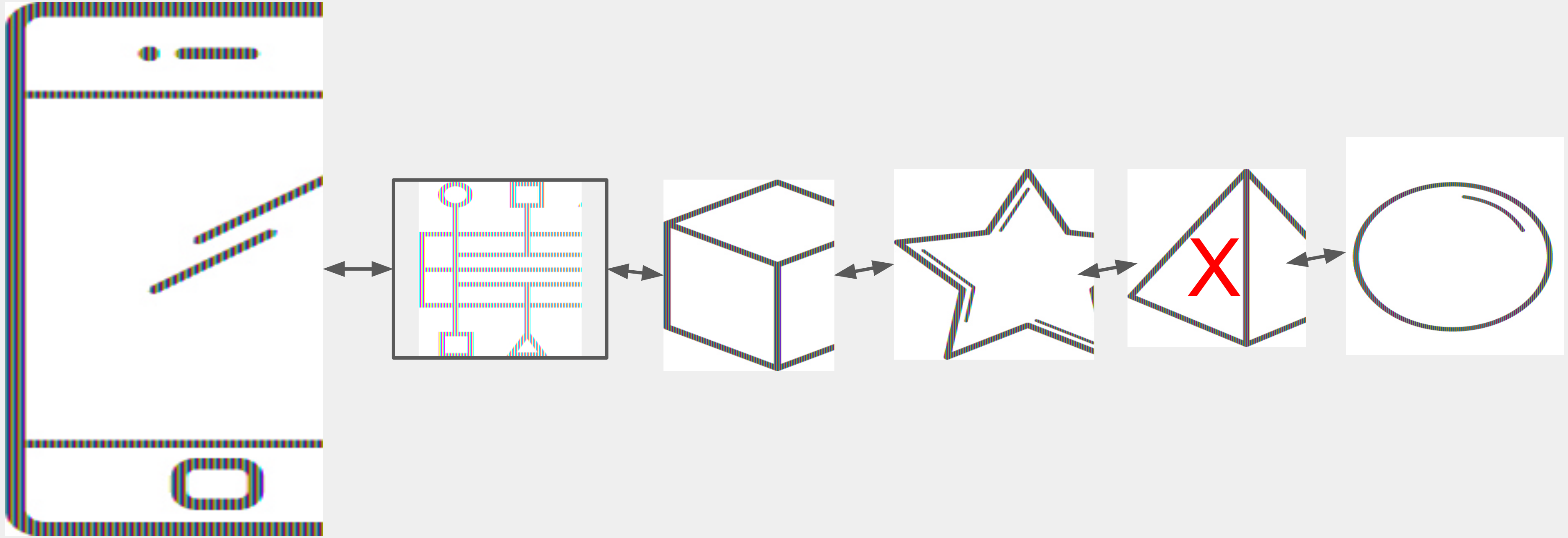
AN EXAMPLE


ACME Laptop 128GB SSD, 8GB RAM
$323.5
Touchscreen
128GB SSD
8GB RAM
Core i3
Windows 10
Add to Cart
In-Store Pickup (15 available)
Raleigh, Central Ave, Store #1123
Recommendations
Pricing Engine
Reviews
Details/Specifications
Location-based availability
People who purchased also...

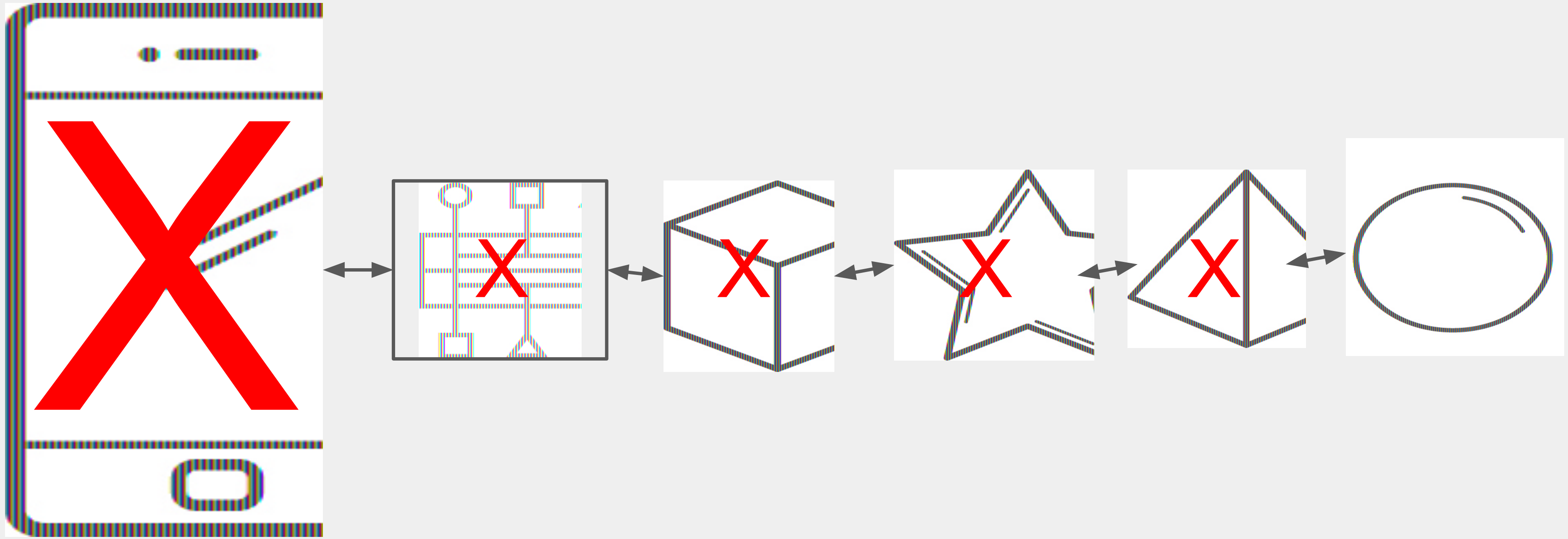
CHAINING



CHAINING (FAILURE)



CHAINING (CASCADING FAILURE)



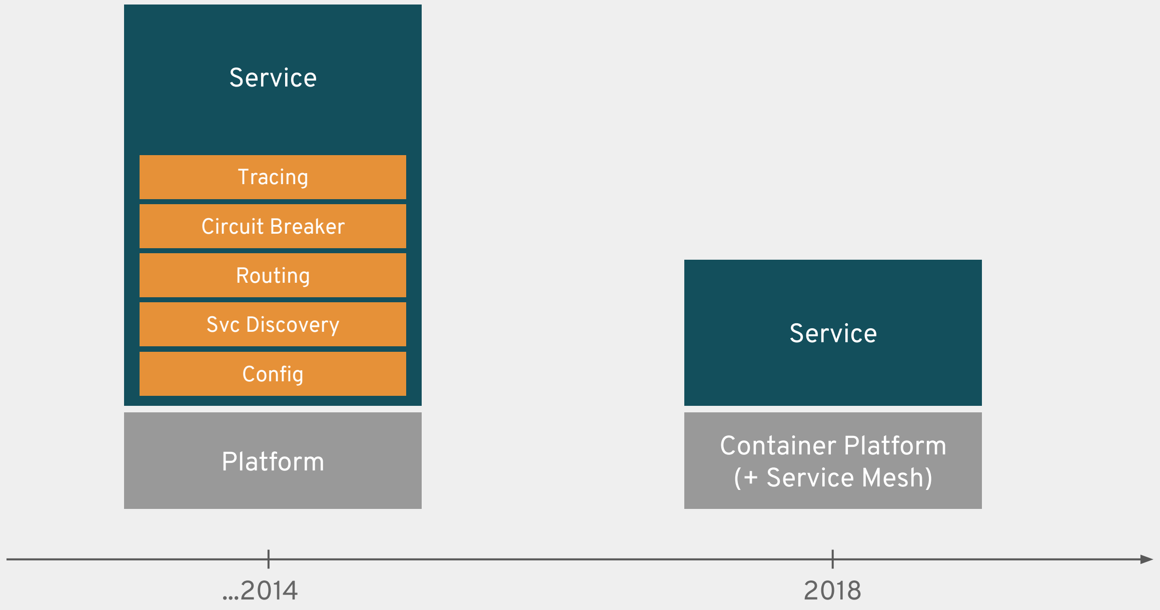
Enter the service mesh

SERVICE MESH

A dedicated network for service-to-service communications

Photo on Visual Hunt

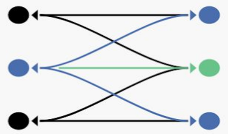
A better way with a service mesh



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

Istio

Connect, secure, control, and observe services.

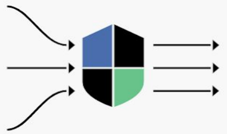


Connect

Intelligently control the flow of traffic and API calls between services, conduct a range of tests, and upgrade gradually with red/black deployments.

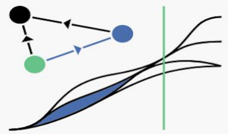
Secure

Automatically secure your services through managed authentication, authorization, and encryption of communication between services.



Control

Apply policies and ensure that they're enforced, and that resources are fairly distributed among consumers.



Observe

See what's happening with rich automatic tracing, monitoring, and logging of all your services.

ISTIO’S CAPABILITIES AT 10,000 FEET

Traffic Management.
Rules and traffic routing lets you control the flow of traffic and API calls between services.

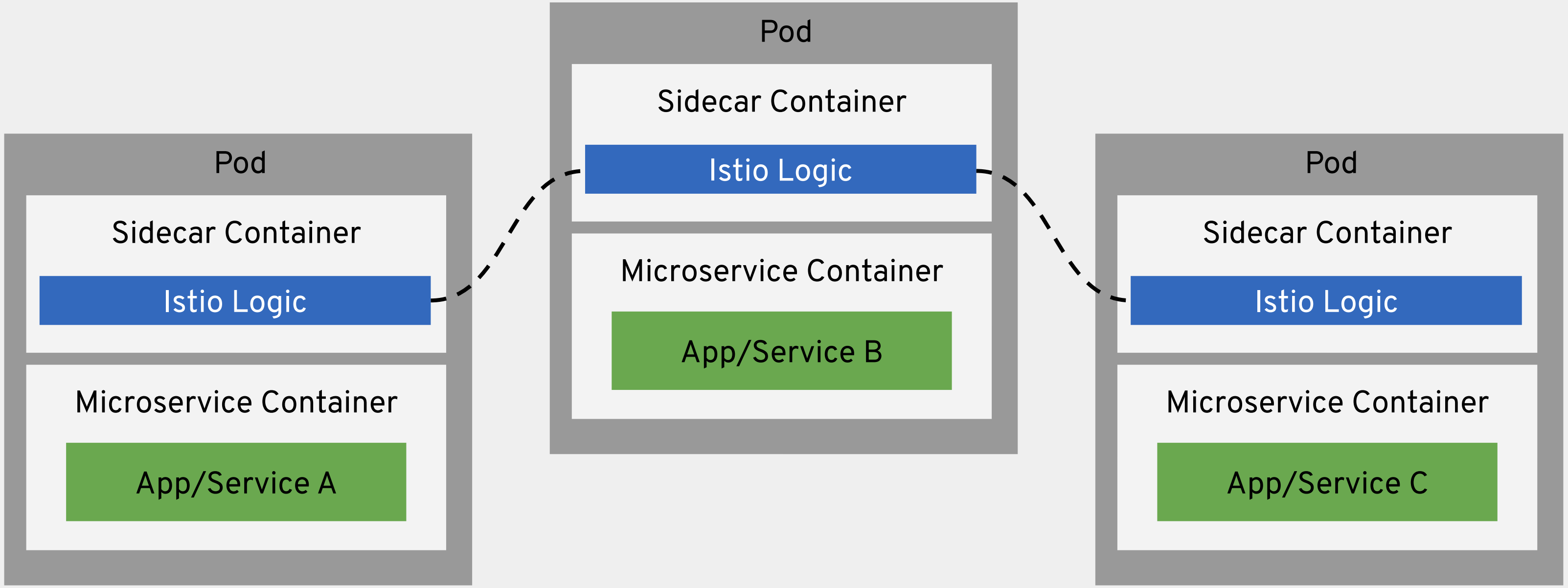
Service Identity and Security.
Enforce consistently across diverse protocols and runtimes with little or no application changes.

Policy Enforcement.
Apply to the interaction between services and ensure they are enforced. Changes are made by configuring the mesh, not by changing application code.

Observability.
Gain understanding of the dependencies between services and the nature and flow of traffic between them, providing the ability to quickly identify and fix issues.

MICROSERVICES WITH ISTIO

connect, manage, and secure microservices *transparently*

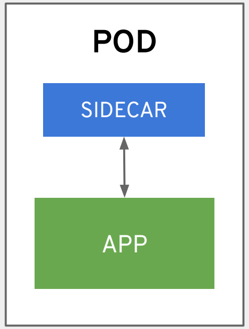


WHAT IS A SIDECAR?

A proxy instance that abstracts common logic away from individual services

SIDECAR PATTERN

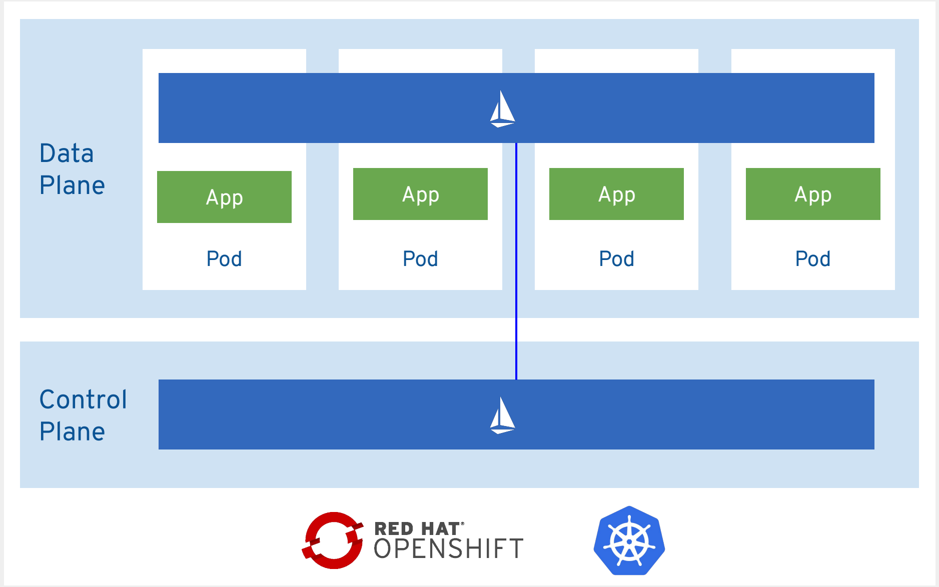
- A utility container in the same pod to enhance the main container's functionality
- Share the same network and lifecycle
- Istio uses an Istio Proxy (L7 Proxy) sidecar to proxy all network traffic between apps



ISTIO PROVIDES BOTH CONTROL AND DATA PLANES

The **data plane** is composed of a set of intelligent proxies (Envoy) deployed as sidecars that mediate and control all network communication between microservices.

The **control plane** is responsible for managing and configuring proxies to route traffic, as well as enforcing policies at runtime.



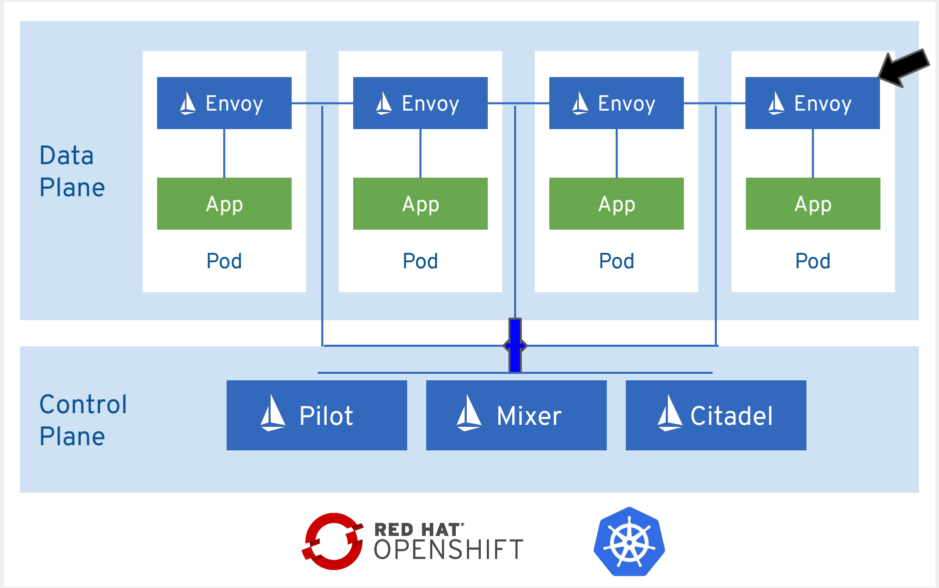
COMPONENTS OF ISTIO

Envoy, originally from Lyft - it's an intelligent proxy. Highly parallel non-blocking, network filtering, service discovery, health checking, dynamically configurable.

Pilot, the component responsible for managing a distributed deployment of Envoy proxies in the service mesh. Intelligent routing, traffic mgmt, resiliency

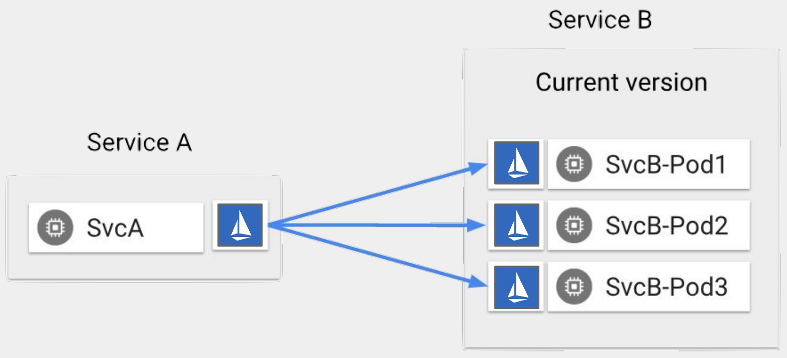
Mixer, which provides the policy and access control mechanisms within the service mesh. Monitoring, reporting, quotas - plugin-based.

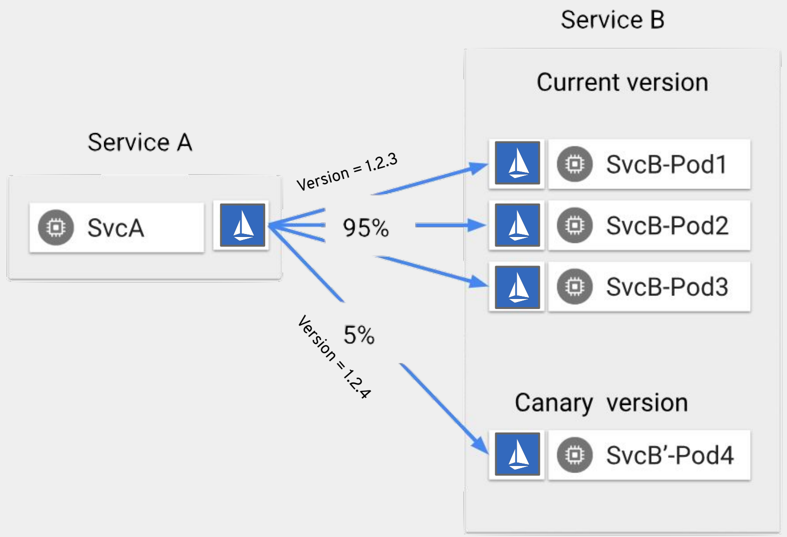
Citadel, control service-service traffic based on origin and user. Key mgmt certificate authority.



It's the sidecar

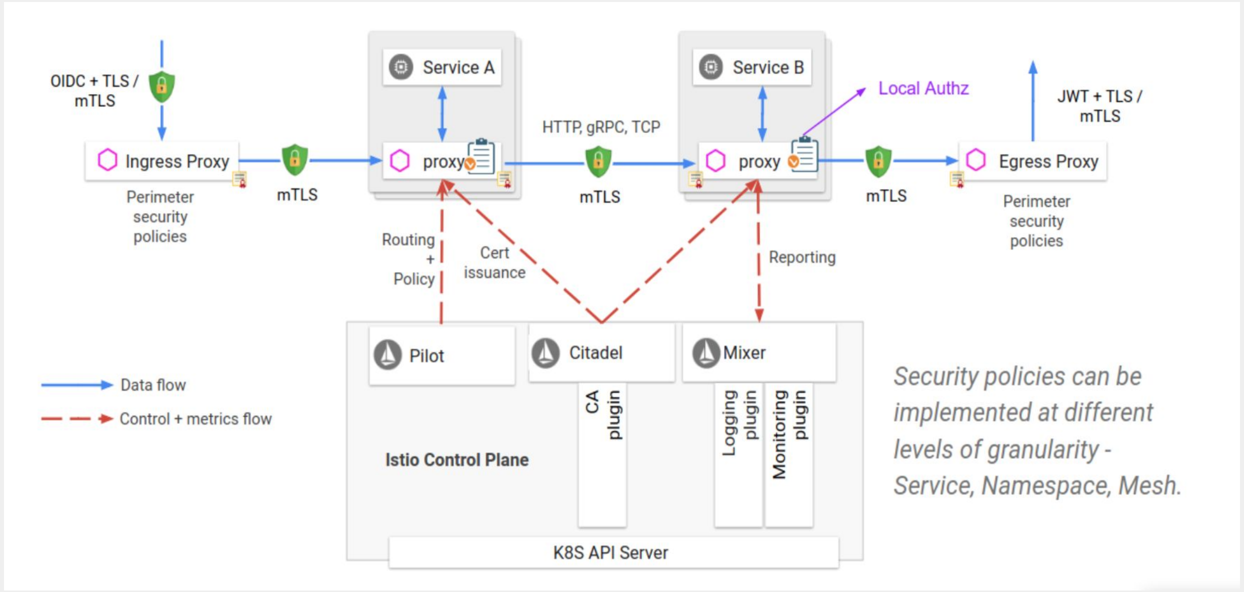
WHAT DOES CONNECT MEAN?





Discovery and Routing: Decoupled from infrastructure, load balancing modes, dynamic routing...
Advanced Deployments: A/B testing, gradual rollouts, canary releases, mirroring...
Failure, Health, and Testing: timeouts, retries, circuit breakers, fault injection, active health checks...

HOW DO YOU SECURE SERVICES?

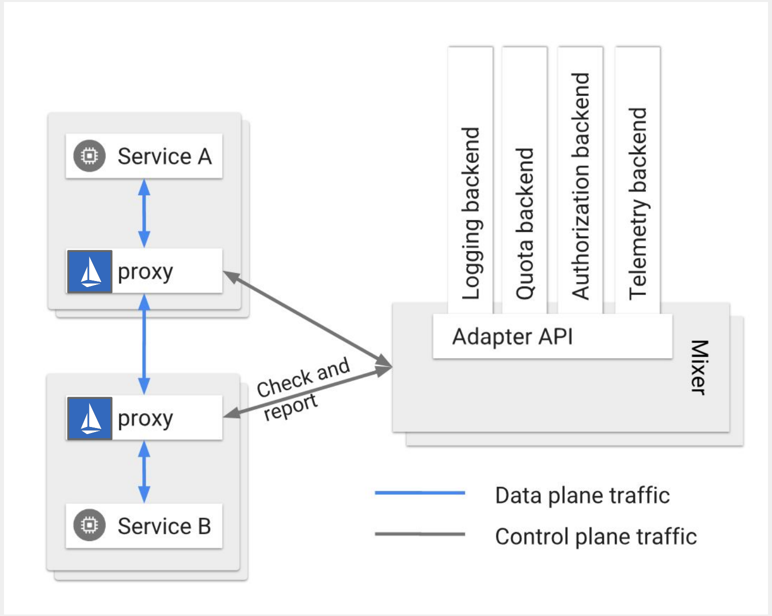


Security by default
no changes needed for application code and infrastructure

Defense in depth
integrate with existing security systems to provide multiple layers of defense

Zero-trust network
build security solutions on untrusted networks

WHAT CAN YOU CONTROL?



Restrict to 2 requests per second per IP :

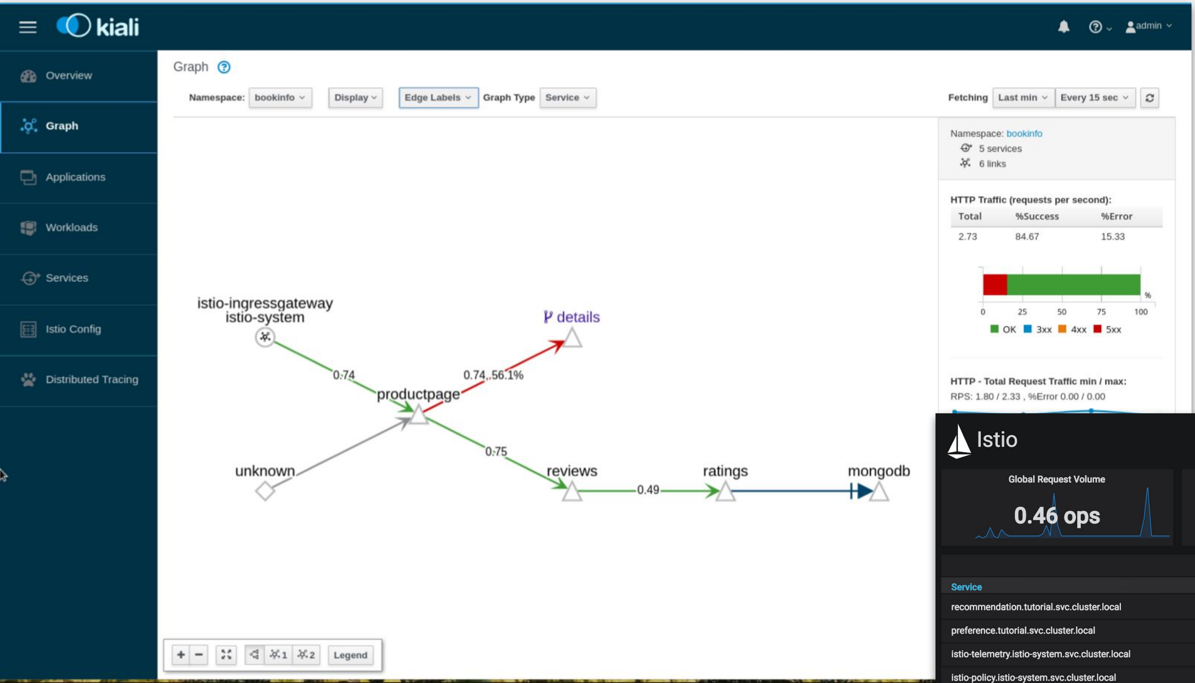
```
quotas:
  - name: requestcount.quota.istio-system
      overrides:
      - dimensions:
              destination: someservice
              maxAmount: 2
```

Exempt if:

```
match(request.headers["cookie"], "user=*") == false
```

Set and Check Policy: Open-ended, connection limits, rate limits, simple denials, lists

HOW CAN YOU <u>OBSERVE</u>?



Namespace: bookinfo | Display | Edge Labels | Graph Type: Service | Fetching: Last min | Every 15 sec

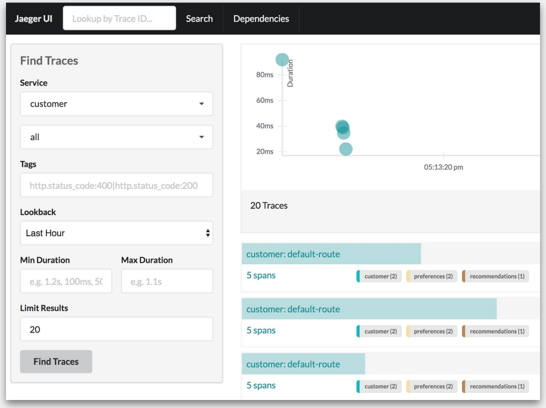
Namespace: bookinfo
5 services
6 links

HTTP Traffic (requests per second):

Total	%Success	%Error
2.73	84.67	15.33

🟩 OK 🟦 3xx 🟧 4xx 🟥 5xx

HTTP - Total Request Traffic min / max:
RPS: 1.80 / 2.33 , %Error 0.00 / 0.00



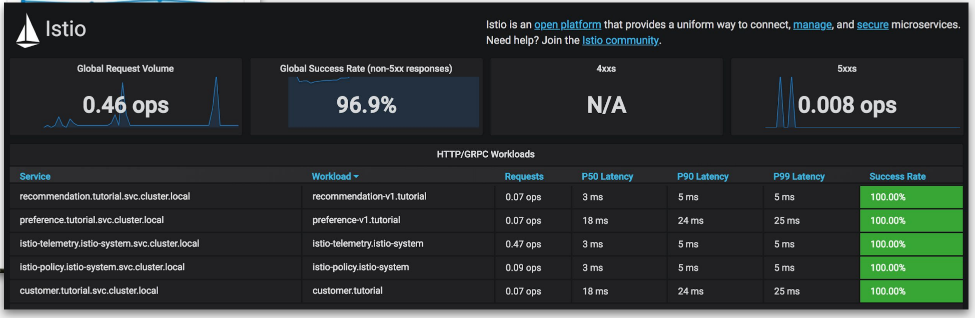
Service: customer
all
Tags: http.status_code:400|http.status_code:200
Lookback: Last Hour
Min Duration: e.g. 1.2s, 100ms, 5(| Max Duration: e.g. 1.1s
Limit Results: 20
Find Traces

20 Traces

customer: default-route
5 spans | customer (2) | preferences (2) | recommendations (1)

customer: default-route
5 spans | customer (2) | preferences (2) | recommendations (1)

customer: default-route
5 spans | customer (2) | preferences (2) | recommendations (1)



Istio is an open platform that provides a uniform way to connect, manage, and secure microservices.
Need help? Join the Istio community.

Global Request Volume	Global Success Rate (non-5xx responses)	4xxs	5xxs
0.46 ops	96.9%	N/A	0.008 ops

HTTP/GRPC Workloads

Service	Workload	Requests	P50 Latency	P90 Latency	P99 Latency	Success Rate
recommendation.tutorial.svc.cluster.local	recommendation-v1.tutorial	0.07 ops	3 ms	5 ms	5 ms	100.00%
preference.tutorial.svc.cluster.local	preference-v1.tutorial	0.07 ops	18 ms	24 ms	25 ms	100.00%
istio-telemetry.istio-system.svc.cluster.local	istio-telemetry.istio-system	0.47 ops	3 ms	5 ms	5 ms	100.00%
istio-policy.istio-system.svc.cluster.local	istio-policy.istio-system	0.09 ops	3 ms	5 ms	5 ms	100.00%
customer.tutorial.svc.cluster.local	customer.tutorial	0.07 ops	18 ms	24 ms	25 ms	100.00%

Understand how your services are operating: Metrics, tracing, network visibility

Red Hat

TRY IT YOURSELF

https://learn.openshift.com/servicemesh/