15.07.2020

CONTINUOUS LIFECYCLE ONLINE

Service Mesh

A good deal for Microservices?

INNOQ

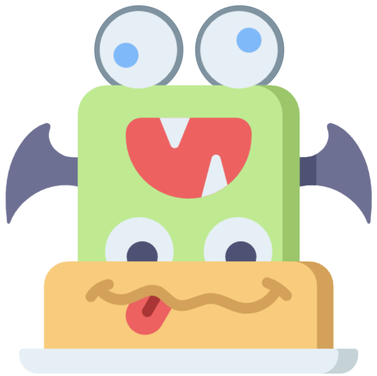
Hanna Prinz

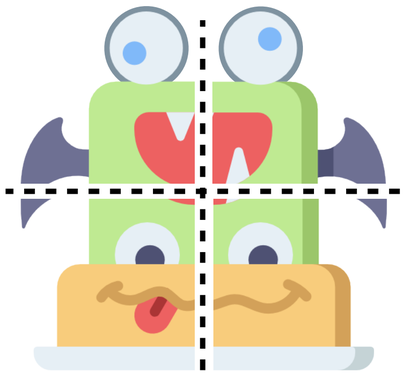
~

"Fix your Microservices by throwing a Mesh at it!"

~

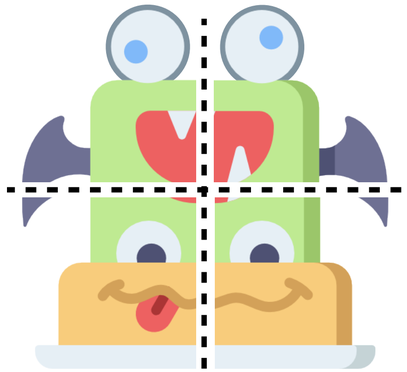
How did we get here?



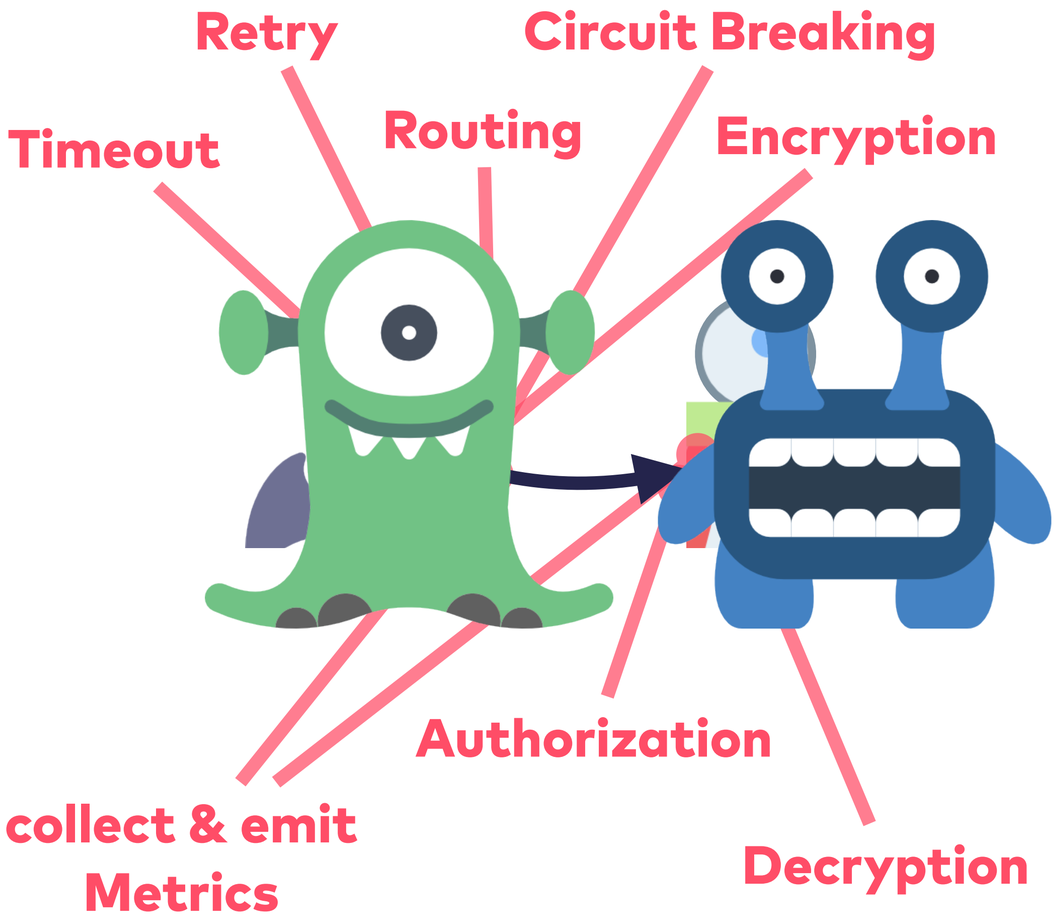


Monolith

Microservices

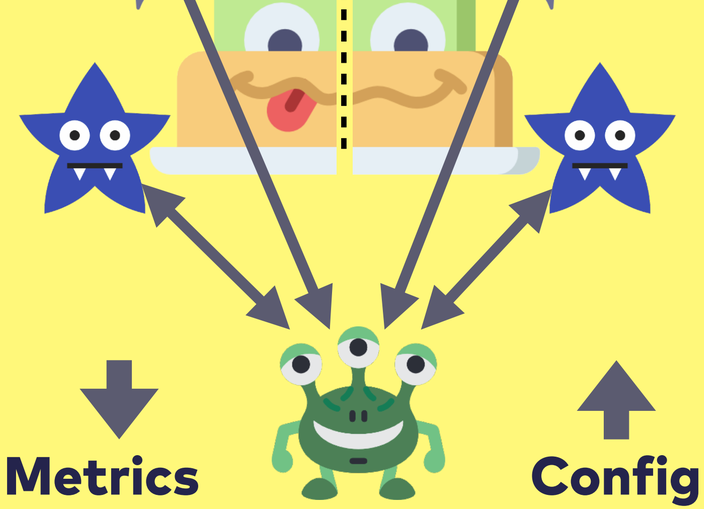


Microservices


Retry
Circuit Breaking
Timeout
Routing
Encryption
Authorization
collect & emit
Metrics
Decryption

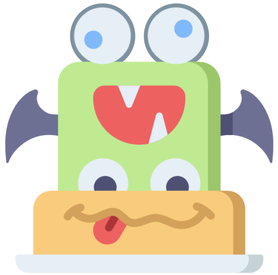
Service Mesh

Retry
Timeout
Circuit Breaker
Routing
Encrypt
Decrpyt
Authorization
Metrics
...

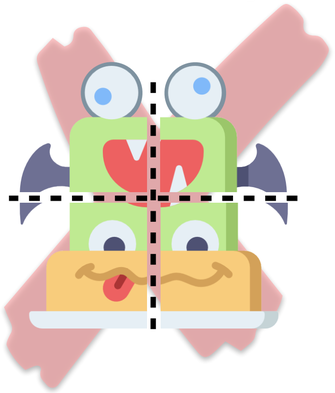


Service Mesh Evolution

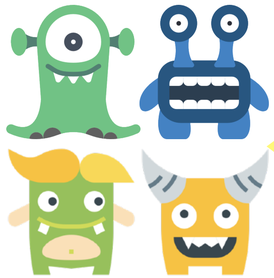
Monolith



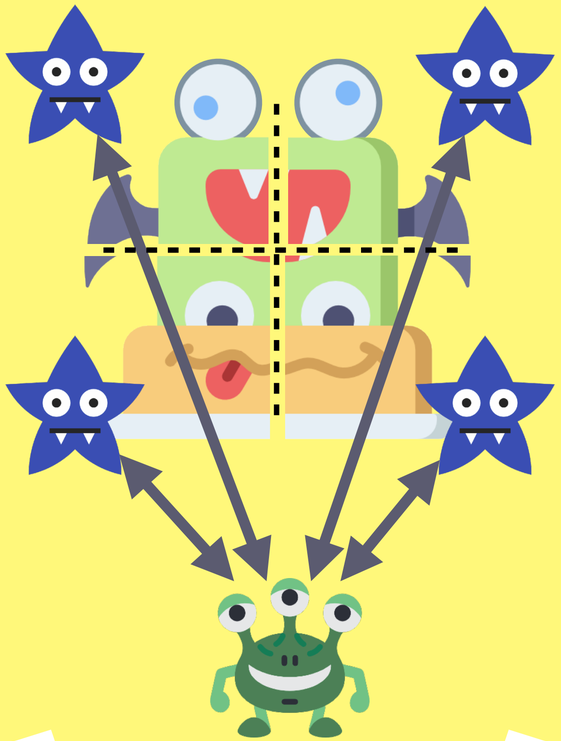
Microservices In Theory



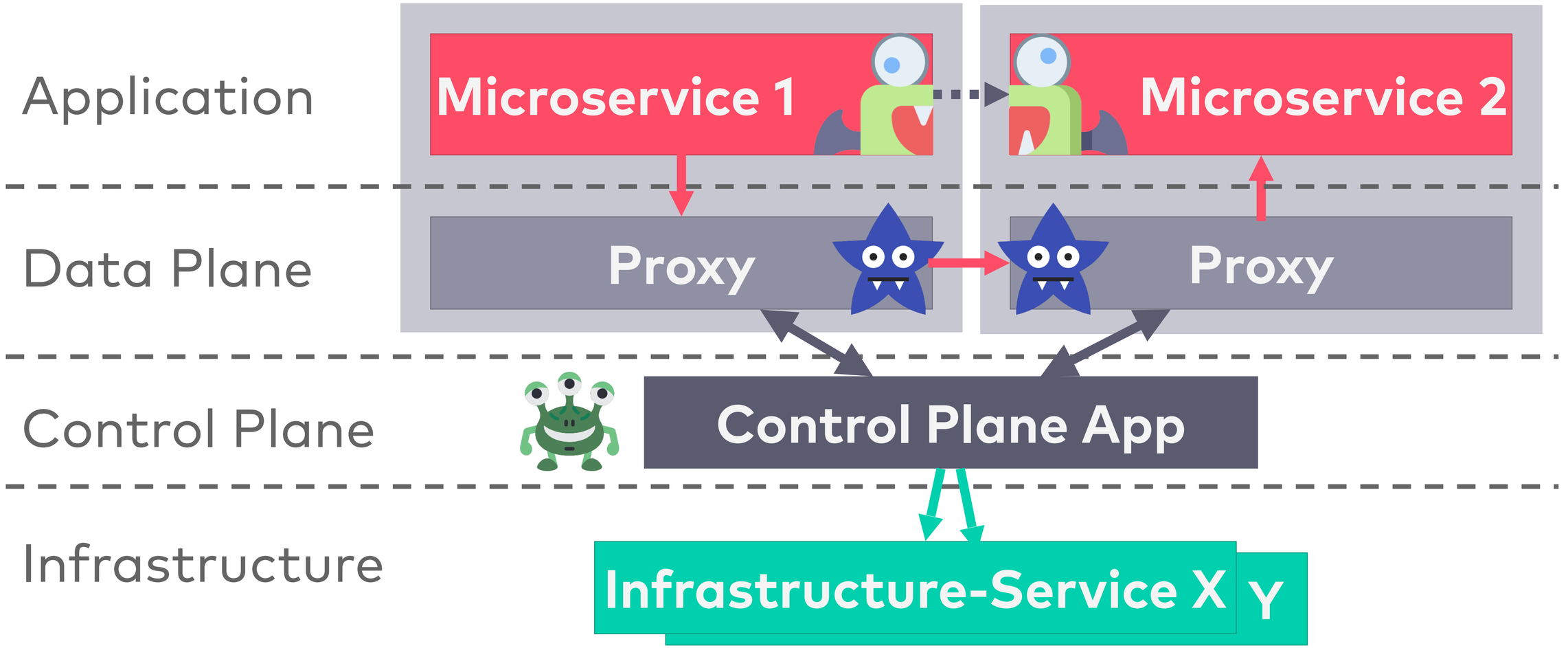
Microservices in Practice



Microservices with Service Mesh



Service Mesh Architecture



Hurray, Technology!

Service Mesh Features

Observability

Routing

Resilience

Security

Monitoring

A Service Mesh can automatically deliver all 4 "Golden Signals":

Latency

Traffic Volume

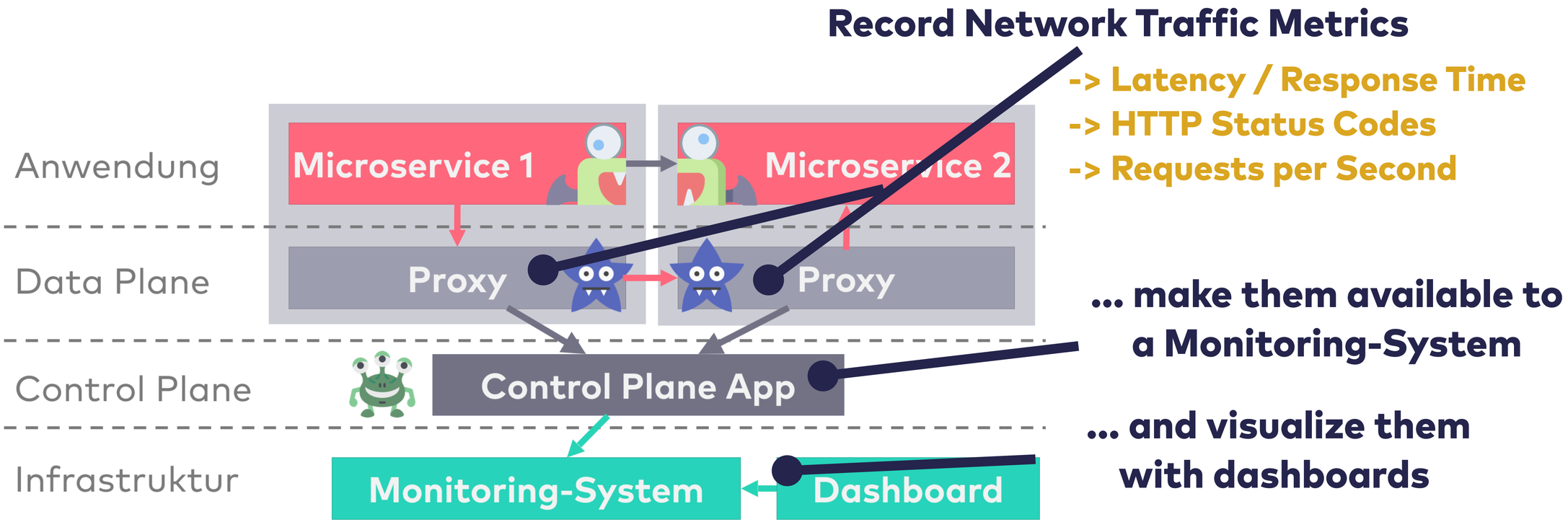
Errors (Status Codes)

Satuation

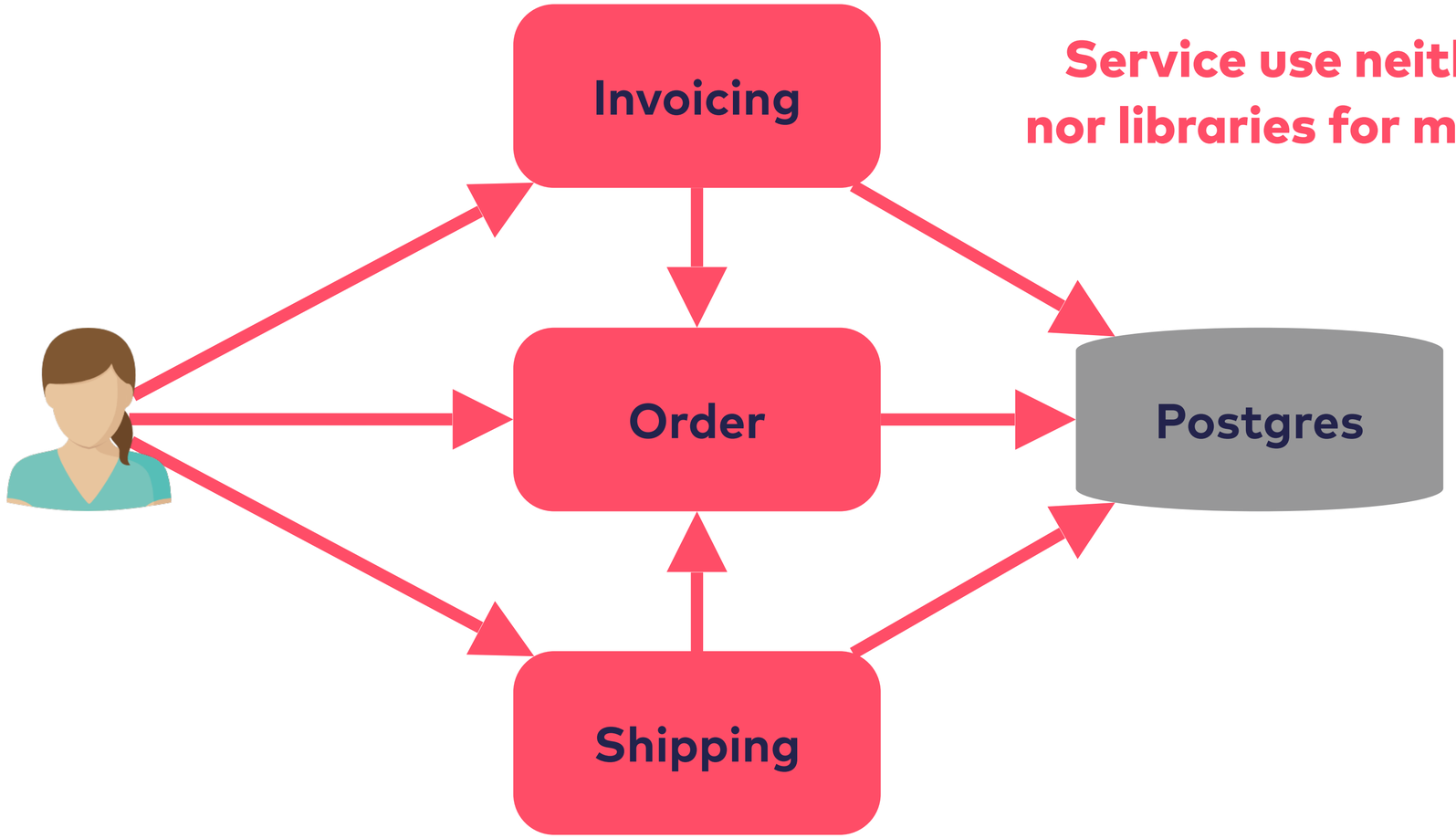
... but it cannot look into the Microservices' Business Logic

https://landing.google.com/sre/sre-book/chapters/monitoring-distributed-systems/#xref_monitoring_golden-signals

Monitoring mit Service Mesh

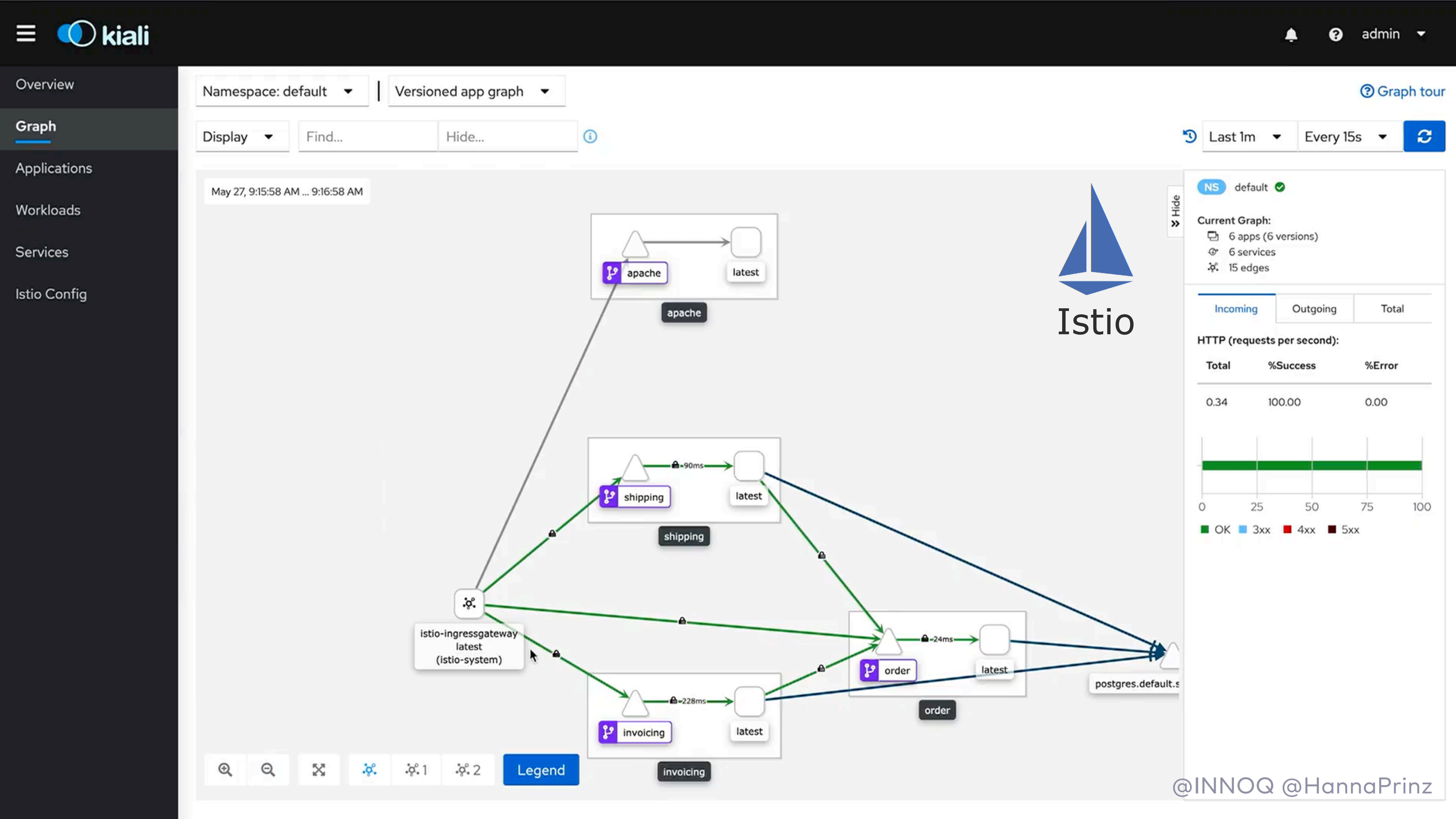


Demo Application



Service use neither code nor libraries for monitoring!

https://github.com/ewolff/microservice-istio


kiali
admin
Overview
Graph
Applications
Workloads
Services
Istio Config
Namespace: default
Versioned app graph
Graph tour
Display
Find...
Hide...
Last 1m
Every 15s
May 27, 9:15:58 AM ... 9:16:58 AM
Istio
apache
latest
apache
shipping
90ms
latest
shipping
istio-ingressgateway
latest
(istio-system)
order
24ms
latest
order
invoicing
228ms
latest
invoicing
postgres.default.s
Legend
NS
default
Current Graph:
6 apps (6 versions)
6 services
15 edges
Hide
Incoming
Outgoing
Total
HTTP (requests per second):
Total
%Success
%Error
0.34
100.00
0.00
0
25
50
75
100
OK
3xx
4xx
5xx
@INNOQ @HannaPrinz

Service Mesh Features

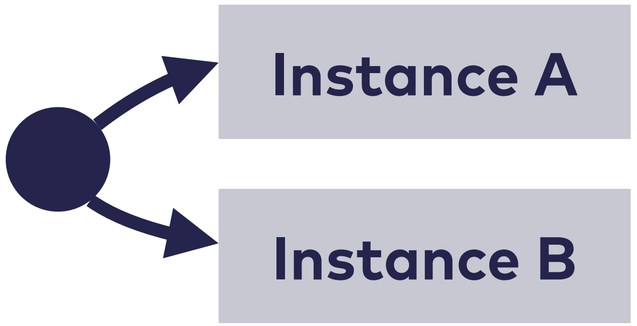
Observability

Routing

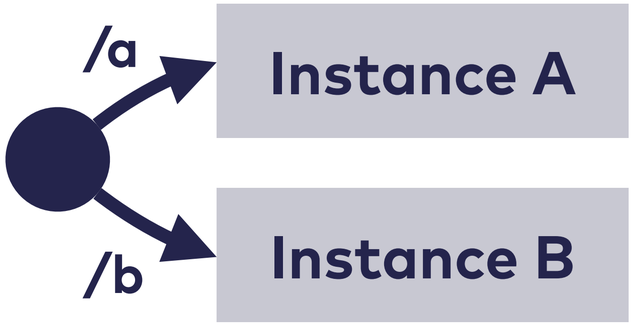
Resilience

Security

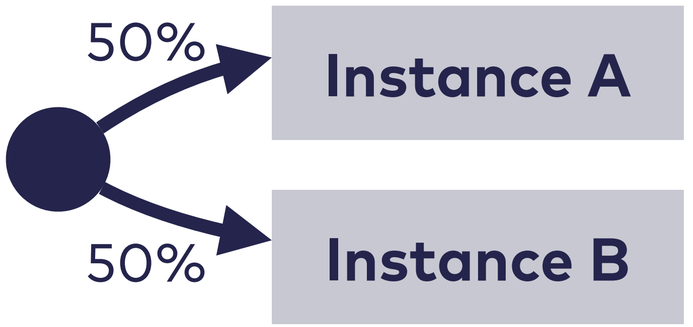
Routing



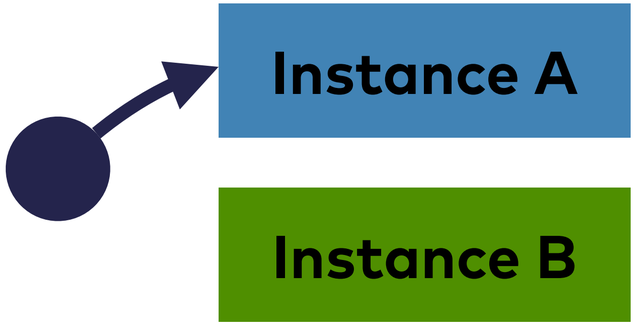
Load Balancing



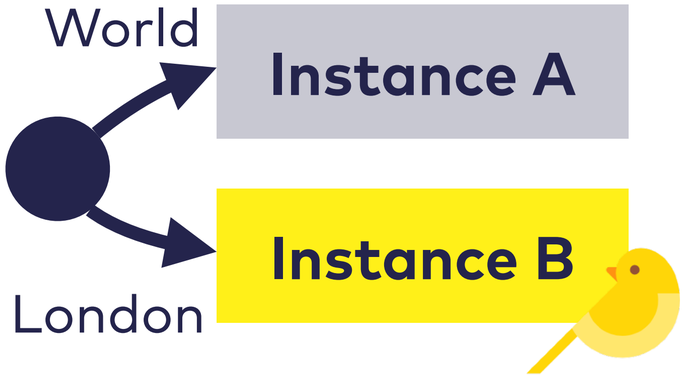
Path-based Routing



A/B-Testing



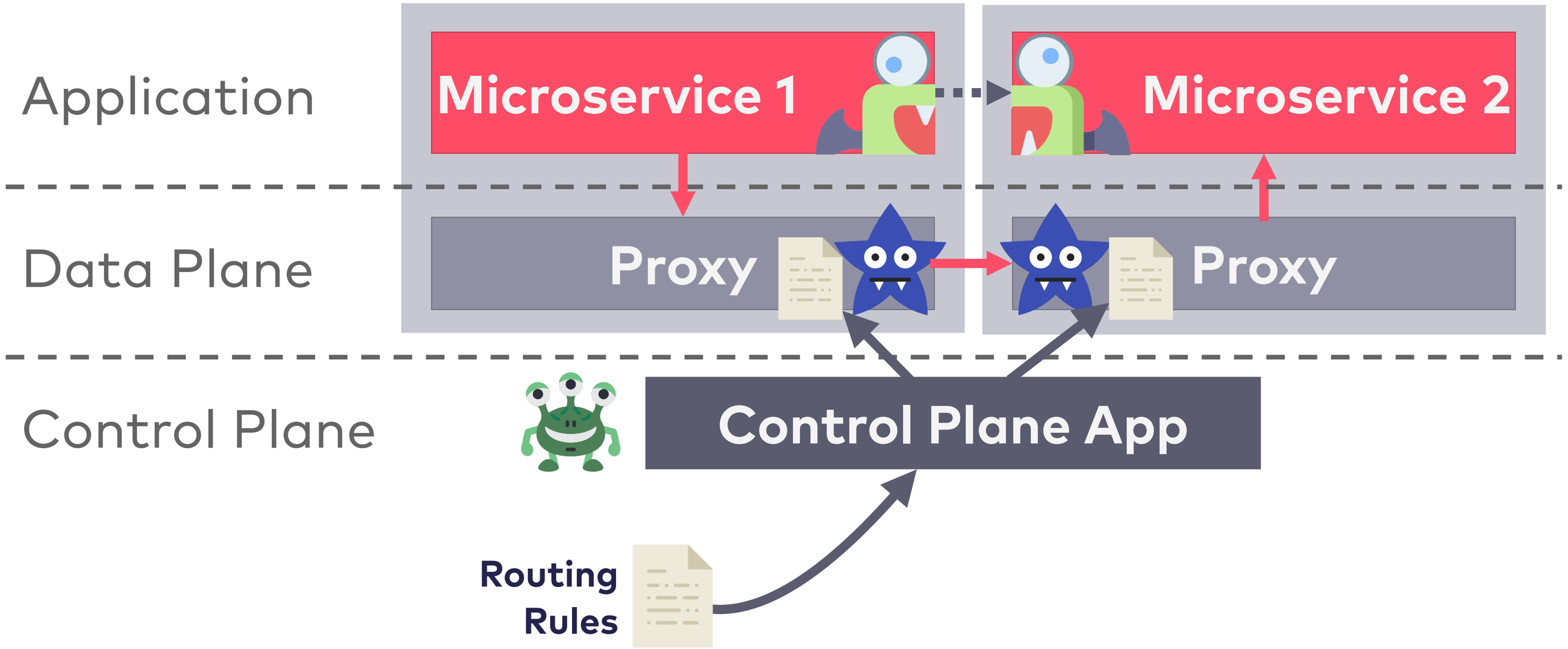
Blue/Green Deployment



Canary Releasing

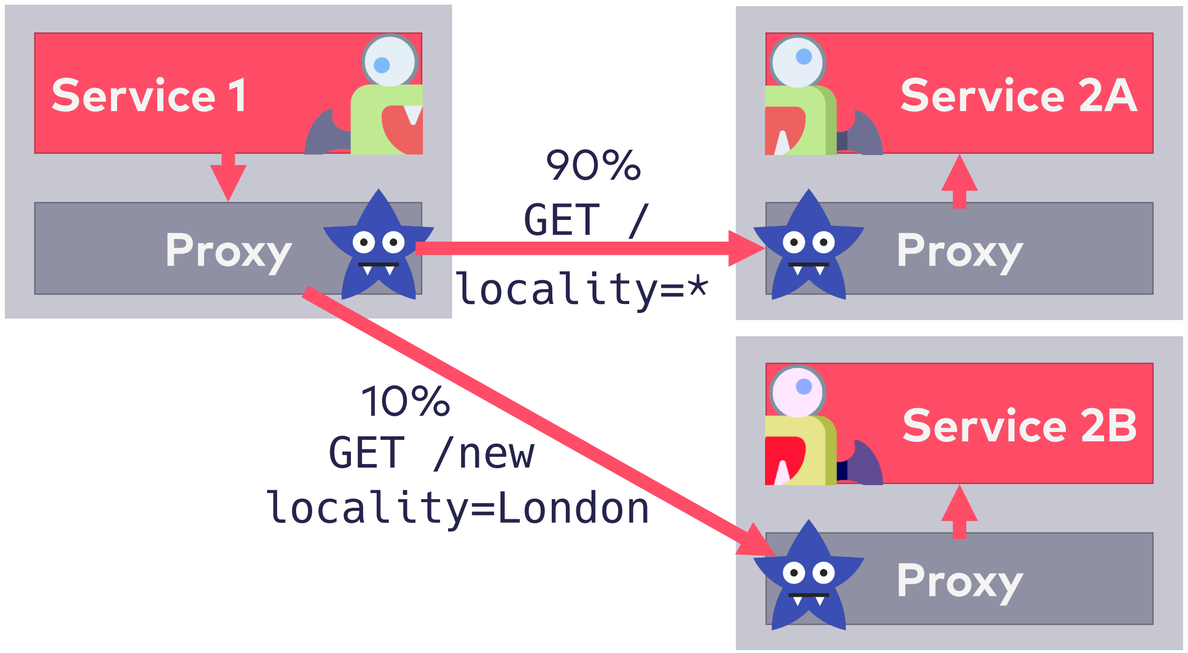
Typically implemented in the
Edge Router / API Gateway
e.g. NGINX, Envoy,
Ambassador, Traefik

Routing with a Service Mesh

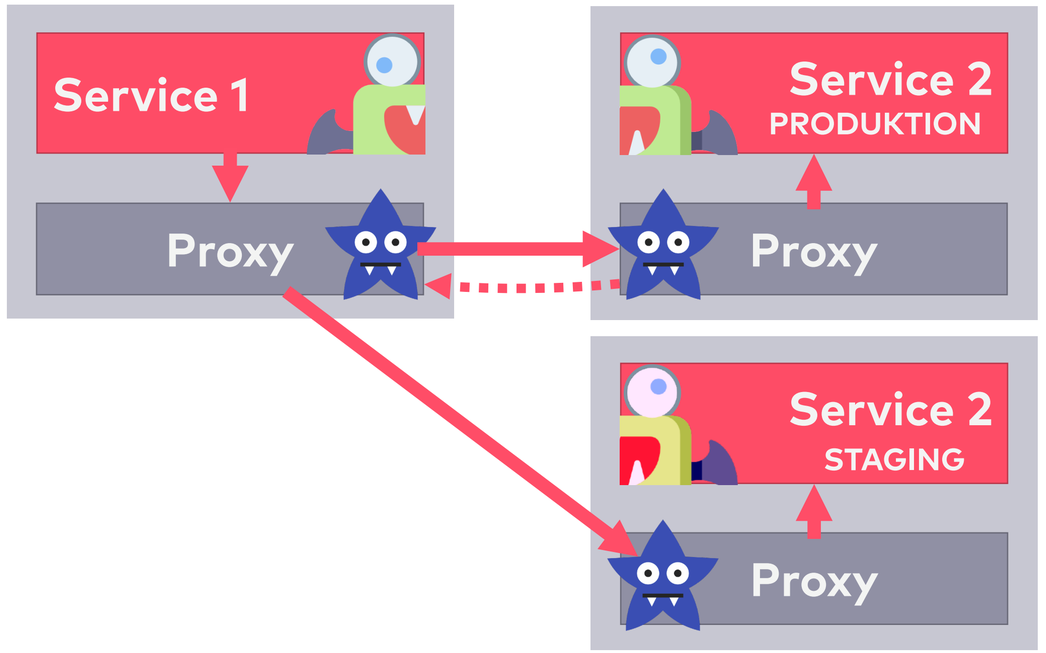


Routing mit Service Mesh

Complex Routing Rules
for A/B Testing and Canary Releasing



Traffic Mirroring



Service Mesh Features

Observability

Routing

Resilience

Security

Resilience

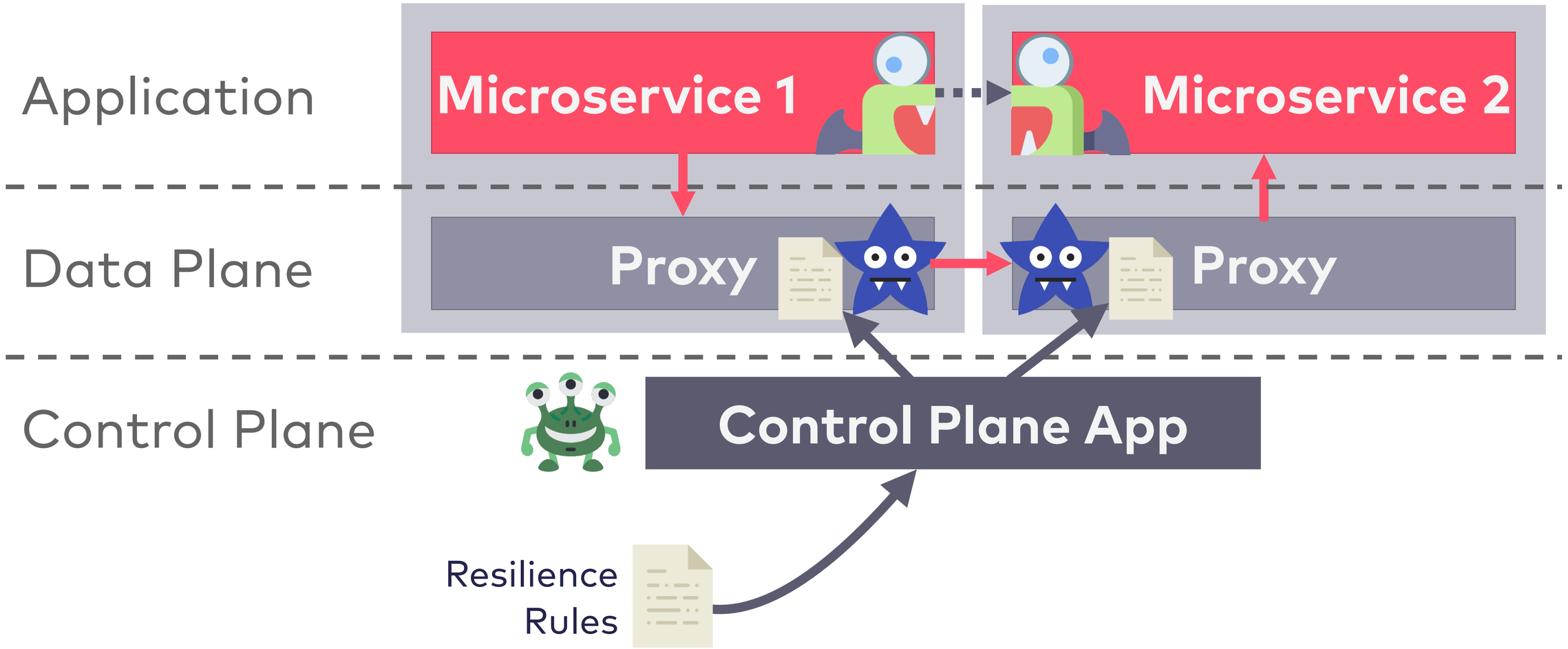
What if a service is not available as expected?

500

Goal:
Overall system continues to function
... with restrictions where necessary

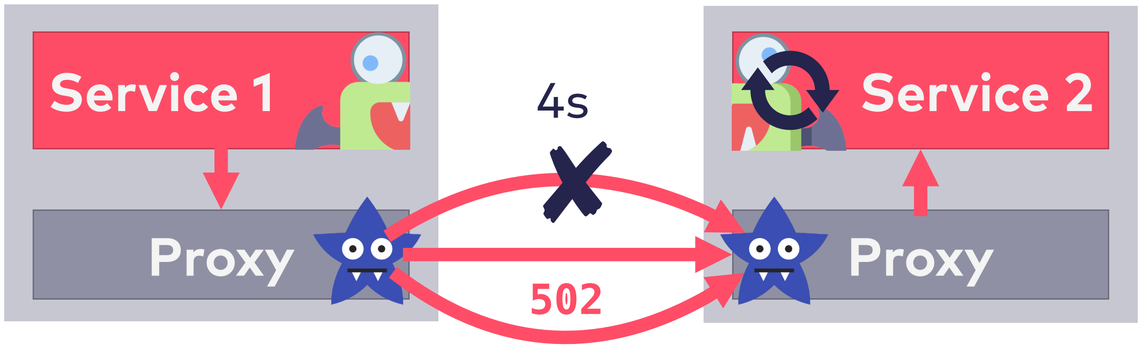
Methods:
Retry, Timeout, Circuit Breaking

Resilience with a Service Mesh

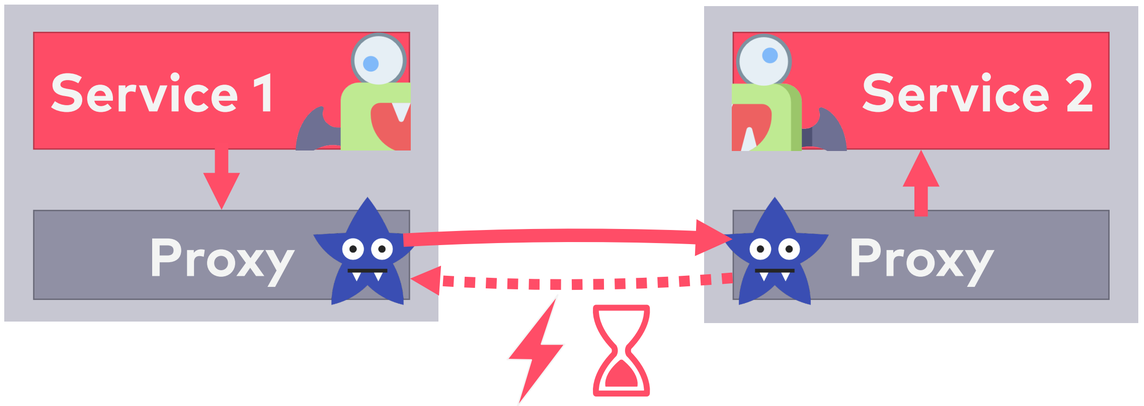


Resilience with a Service Mesh

Timeout
Retry



Fault Injection
Delay Injection



Service Mesh Features

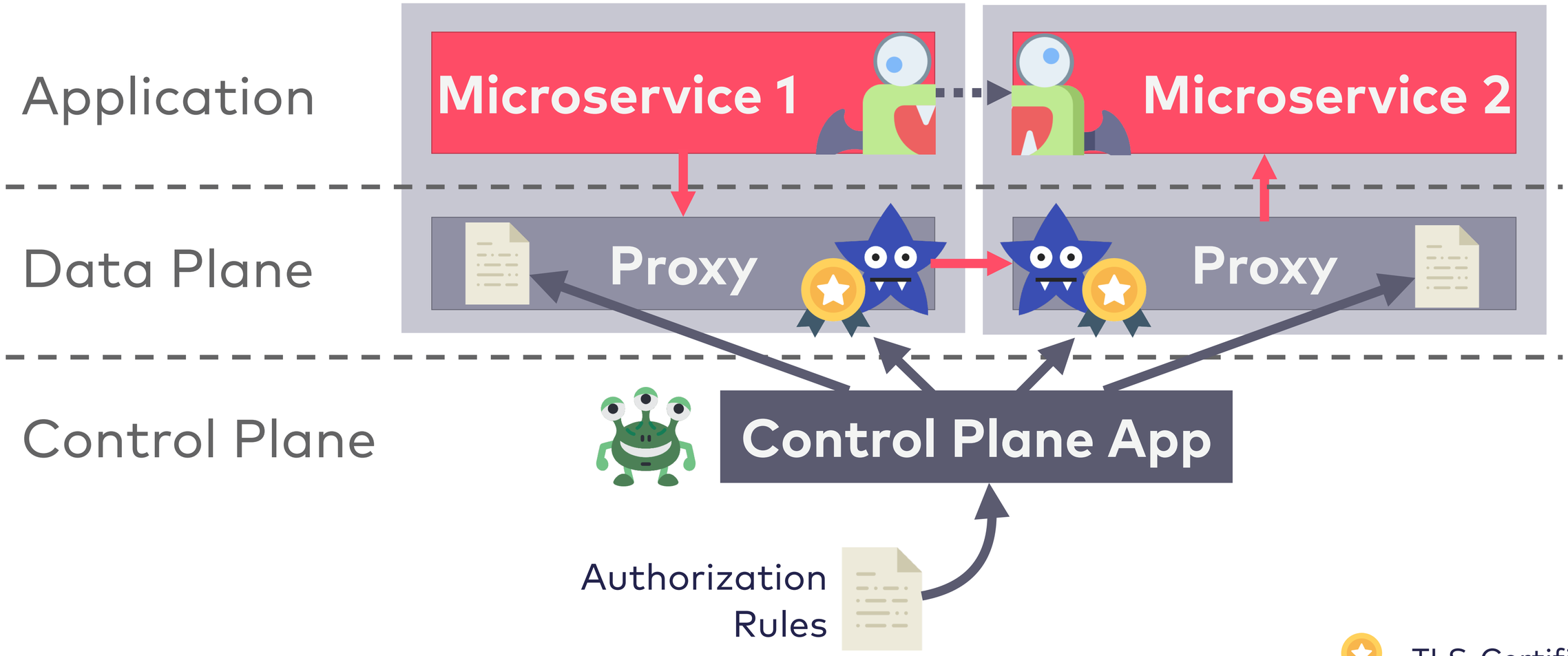
Observability

Routing

Resilience

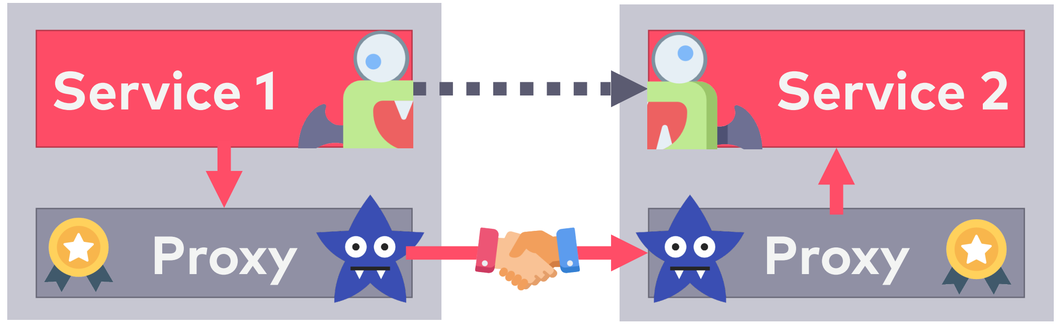
Security

Security with a Service Mesh

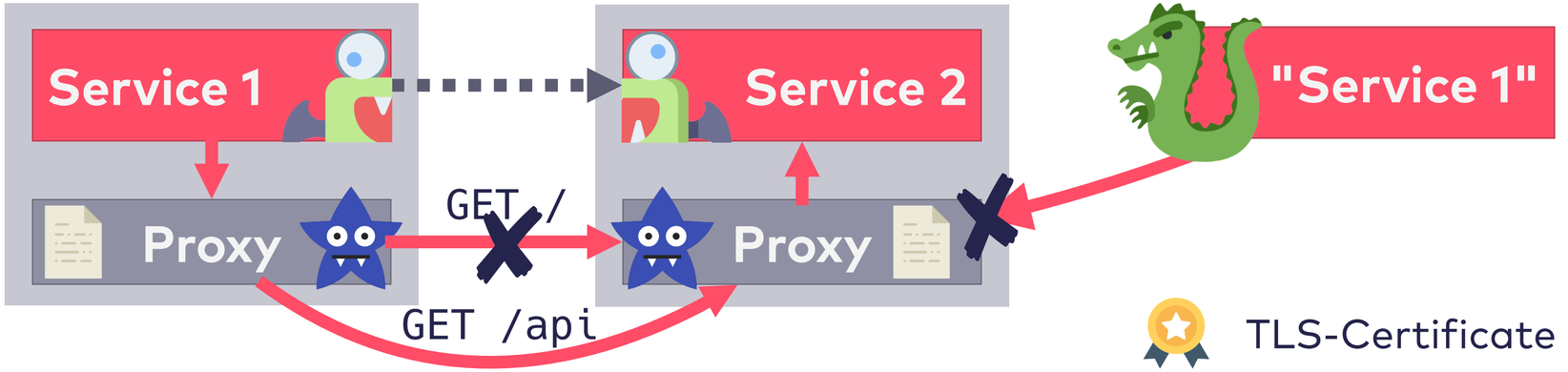


Security with a Service Mesh

Authentication with mTLS



Authorization



TLS-Certificate

Authorization Rule

Service Mesh Features

Network metrics and access logs

Emit tracing data to backend

Observability

Business metrics or logs

Passing on tracing headers

Alerting

Complex routing rules

Canary Releasing & A/B-Testing

Routing

Automatic Canary Releasing

Timeouts & Retries

Circuit Breaking

Resilience

Use cache or standard responses in Circuit Breaker

Authentication with mTLS

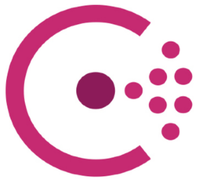
Authorization

Security

Service Mesh Market

Service Mesh Implementations

LINKERD



HashiCorp Consul

AWS App Mesh

mæsh

Istio

servicemesh.es

Kuma

servicemesh.es

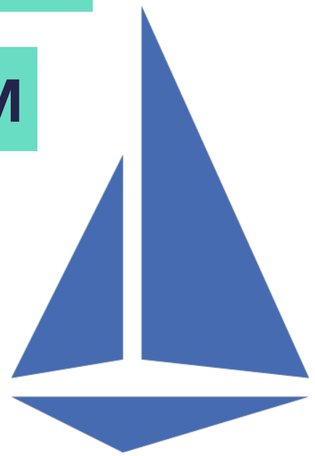
	Istio	Linkerd 2	AWS App Mesh	Consul Connect	Maesh	Kuma
Current version	1.4	2.6		1.6	1.0	0.3
License	Apache License 2.0	Apache License 2.0	Closed Source	Mozilla License	Apache License 2.0	Apache License 2.0
Developed by	Google, IBM, Lyft	Buoyant	AWS	HashiCorp	containous	Kong
Service Proxy	Envoy	linkerd-proxy	Envoy	defaults to Envoy, exchangeable	Traefik	Envoy
Ingress Controller	Envoy / Own Concept	any		any	any	any
Governance	see Istio Community	see Linkerd Governance and CNCF Charter	AWS	see Contributing to Consul	see Contributing notice	see Contributing notice
Tutorial	Istio Tasks	Linkerd Tasks	AWS App Mesh Getting Started	Hashicorp Learn platform	Maesh Example	Kuma Kubernetes Quickstart
Platform	Kubernetes, Consul & Nomad	Kubernetes	ECS, Fargate, EKS, EC2	Consul, Nomad, Kubernetes	Kubernetes	Kubernetes, VMs (Universal)
Automatic Sidecar Injection	yes	yes	yes	yes	yes (per Node)	yes
Used in production	yes	yes				
Advantages	Istio can be adapted and extended like no other Mesh. Its many features are available for Kubernetes and other	Linkerd 2 is designed to be non-invasive and is optimized for performance and usability. Therefore, it requires little	AWS App Mesh is integrated into the AWS landscape and	Consul Connect can be used in any Consul environment and therefore does not require a scheduler. The	Maesh focuses on a selection of features to achieve good	Kuma supports both Kubernetes and plain VMs and allows you to

Istio VS Linkerd 2

*2017

By **Google & IBM**

optimized for **feature-richness** and **configurability**



optimized for Kubernetes, but not exclusive

*2017

by **Buoyant**

optimized for **usability** and **performance**

Kubernetes only

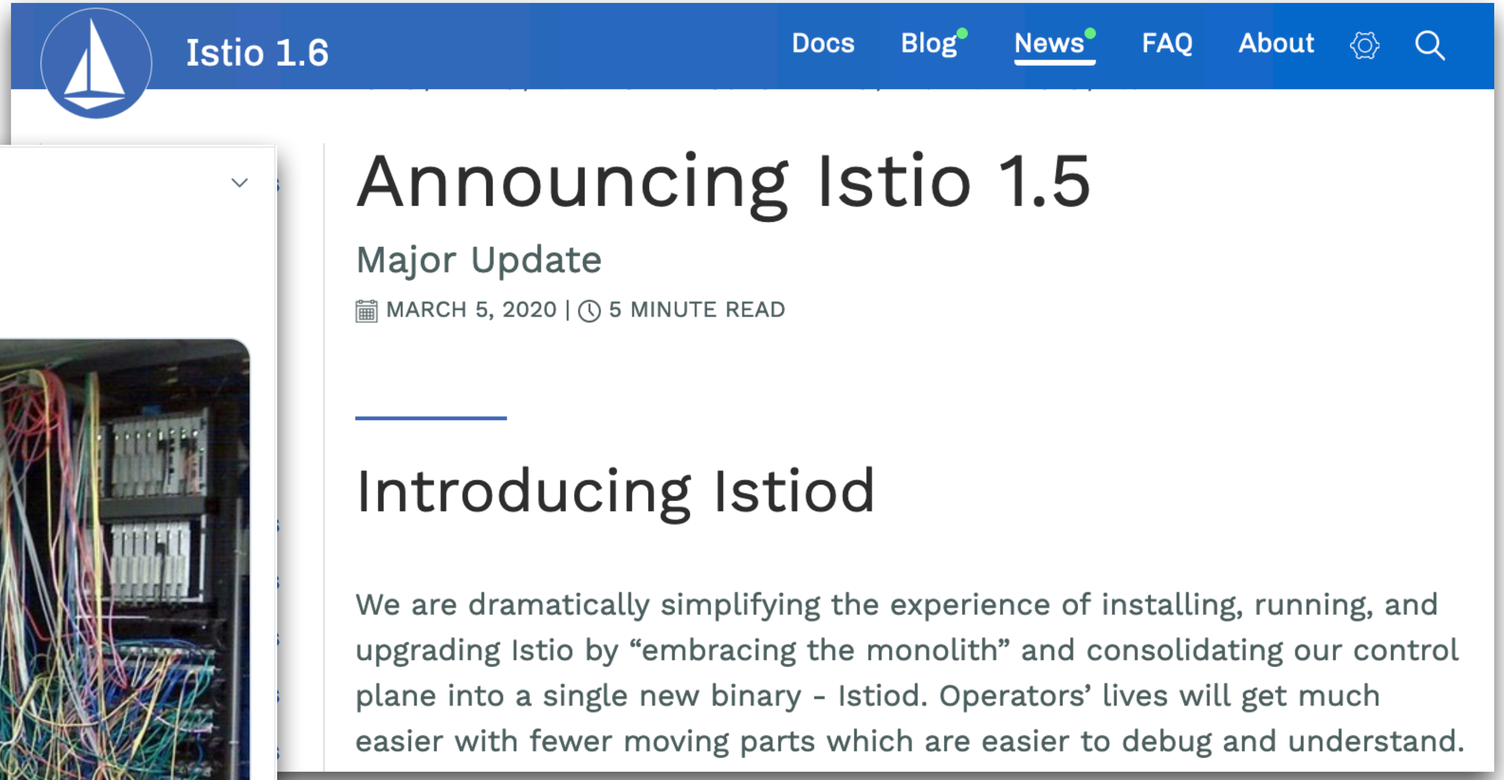
Features

		Istio	LINKERD
 Observability	Network metrics and access logs	✓	✓
	Emit tracing data to backend	✓	✓
 Routing	Complex routing rules	✓	✓
	Canary Releasing & A/B-Testing	✓	✓
 Resilience	Timeouts & Retries	✓	✓
	Circuit Breaking	✓	✗
 Security	Authentication with mTLS	✓	✓
	Authorization	✓	✗

Nice Table.

Usability


Dave Strebel
@dave_strebel
“Finally got Istio into production”
2:21 AM · Sep 18, 2019 from Apple Valley, MN · Tweetbot for iOS
322 Retweets 1.2K Likes

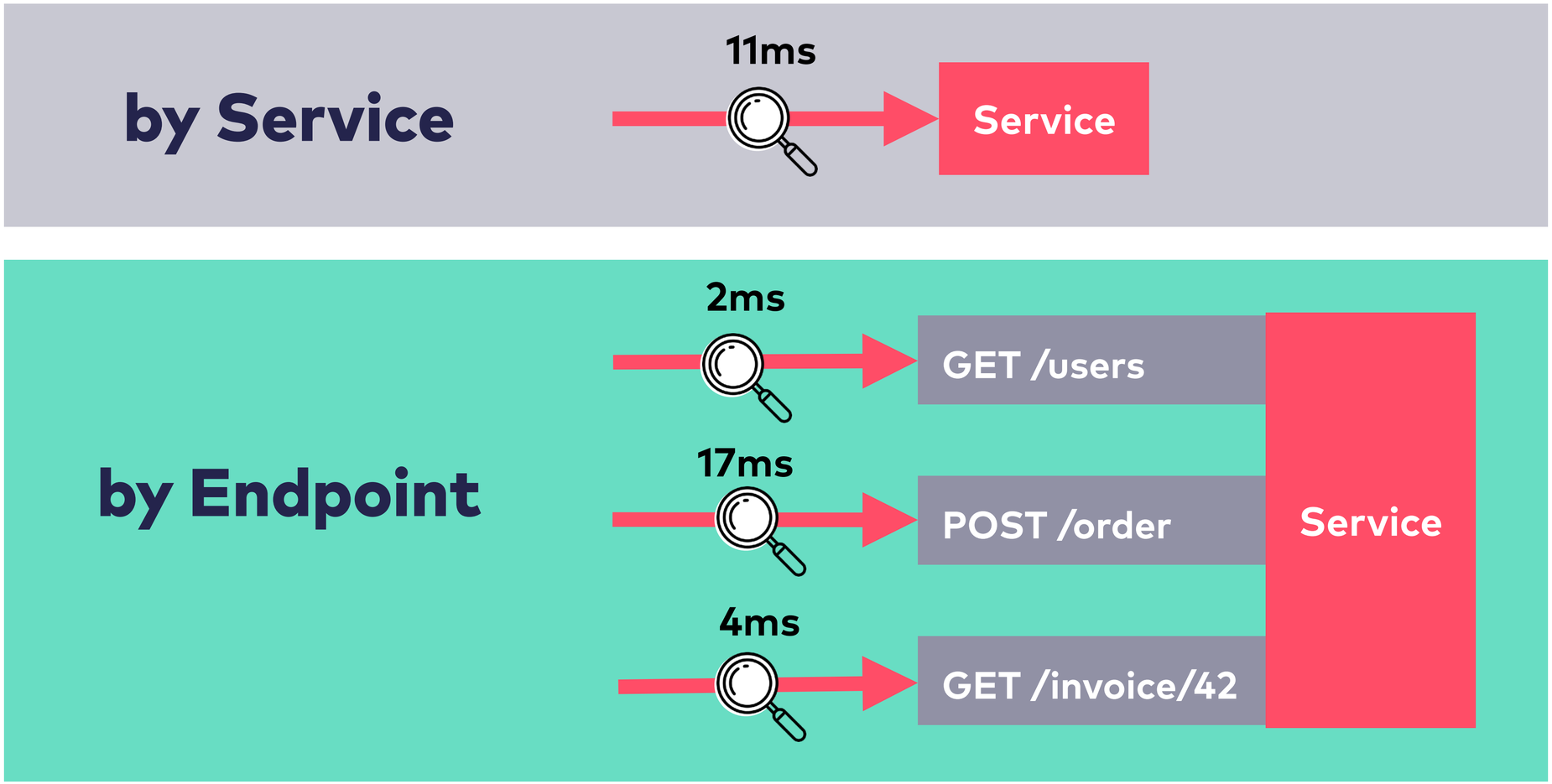

Istio 1.6
Docs
Blog
News
FAQ
About
Announcing Istio 1.5
Major Update
MARCH 5, 2020 | 5 MINUTE READ
Introducing Istiod
We are dramatically simplifying the experience of installing, running, and upgrading Istio by “embracing the monolith” and consolidating our control plane into a single new binary - Istiod. Operators’ lives will get much easier with fewer moving parts which are easier to debug and understand.


Service Mesh
Magic is based
on a lot of
YAML

Istio VS Linkerd 2

Comparison of Configuration

Monitoring Precision



LINKERD

Namespace > default

CLUSTER

- Namespaces
- Control Plane

DEFAULT

WORKLOADS

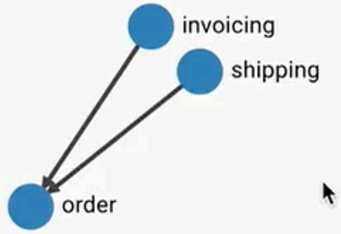
- Cron Jobs
- Daemon Sets
- Deployments
- Jobs
- Pods
- Replica Sets
- Replication Controllers
- Stateful Sets

CONFIGURATION

- Traffic Splits

TOOLS

- Tap
- Top



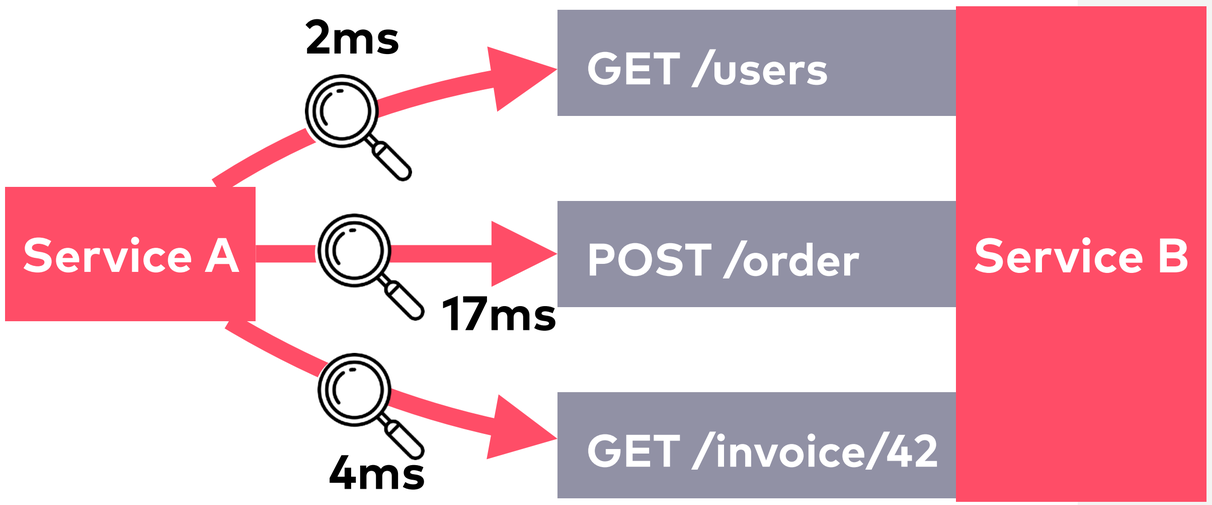
Deployments

Deployment	Meshed	Success Rate	RPS	P50 Latency	P95 Latency	P99 Latency	Grafana
apache	1/1	100.00%	0.42	1 ms	1 ms	1 ms	
invoicing	1/1	100.00%	0.83	6 ms	16 ms	19 ms	
order	1/1	100.00%	1.75	17 ms	29 ms	30 ms	
postgres	1/1	---	---	---	---	---	
shipping	1/1	100.00%	0.83	10 ms	19 ms	20 ms	

Pods

Pod	Meshed	Success Rate	RPS	P50 Latency	P95 Latency	P99 Latency	Grafana

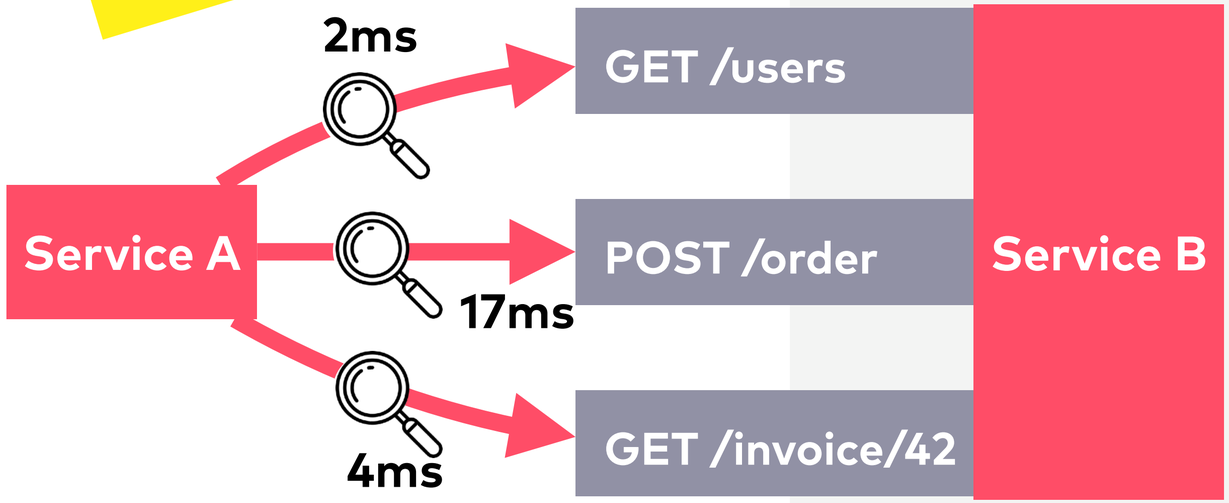
Monitoring Precision by Endpoint with Linkerd 2




```yaml
apiVersion: linkerd.io/v1alpha1
kind: ServiceProfile
metadata:
  name: service-b.default.svc.cluster.local
  namespace: default
spec:
  routes:
  - name: GET /users
    condition:
      method: GET
      pathRegex: /users
  - name: POST /order
    condition:
      method: POST
      pathRegex: /order
  - name: GET /invoice/{id}
    condition:
      method: GET
      pathRegex: /invoice/[^/]*
```

Monitoring Precision by Endpoint with Istio

Experimental in Version 1.6




```yaml
apiVersion: networking.istio.io/v1alpha3
kind: EnvoyFilter
metadata:
  name: istio-attributegen-filter
spec:
  workloadSelector:
    labels:
      app: reviews
  configPatches:
  - applyTo: HTTP_FILTER
    match:
      context: SIDECAR_INBOUND
      proxy:
        proxyVersion: '1\.6.*'
      listener:
        filterChain:
          filter:
            name: "envoy.http_connection_manager"
            subFilter:
              name: "istio.stats"
    patch:
      operation: INSERT_BEFORE
      value:
        name: istio.attributegen
        typed_config:
          "@type": type.googleapis.com/udpa.type.v1.TypedStruct
          type_url: type.googleapis.com/envoy.extensions.filters.http.wasm.v3.Wasm
          value:
            config:
              configuration: |
                {
                  "attributes": [
                    {
                      "output_attribute": "istio_operationId",
                      "match": [
                        {
                          "value": "GET /users",
                          "condition": "request.url_path == '/users' && request.method == 'GET'"
                        },
                        {
                          "value": "POST /order",
                          "condition": "request.url_path == '/order' && request.method == 'POST'"
                        },
                        {
                          "value": "GET /invoice/{id}",
                          "condition": "request.url_path.matches('^/invoice/[[:alnum:]]*$')
                             && request.method == 'GET'"
                        }
                      ]
                    }
                  ]
                }
              vm_config:
                runtime: envoy.wasm.runtime.null
                code:
                  local: { inline_string: "envoy.wasm.attributegen" }
```

Performance & Ressourcen

- **Latency** - highly dependent on traffic
  - **Istio**: additional ca. **3ms** Latency - per call between services!
  - **Linkerd 2**: no current numbers, similar to Istio in earlier versions
- **Resources**
  - Additional containers for Control Plane & each sidecar
  - → Increased CPU & memory consumption

But: Depending on the concrete project
→ make your own benchmark!

TL;DR

Service Mesh

+

Solves many essential problems of microservices

... without changing the code!

−

Another complex piece of technology

Increased latency and resource consumption

Decision support

Service Mesh Indicators

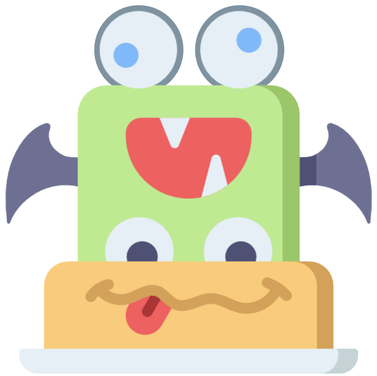
- Many **microservices**, many synchronous calls
- Many **unsolved problems** in monitoring, routing, resilience and/or security
- Most services run in **Kubernetes**

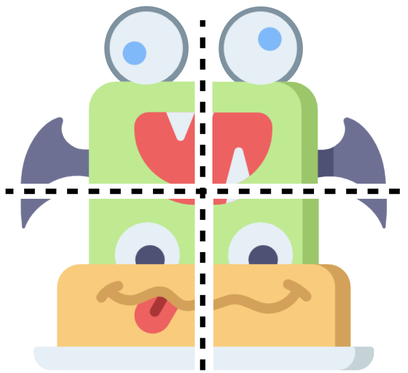
Selection criteria

- Which features are really **missing**?
- Existing **infrastructure** - Kubernetes, Consul, AWS, ...
- Temporal and **cognitive capacity** in the team
- Activity of the **Community**

Objective: As much complexity as necessary, but as little as possible

Complexity? Uhm...





Monolith

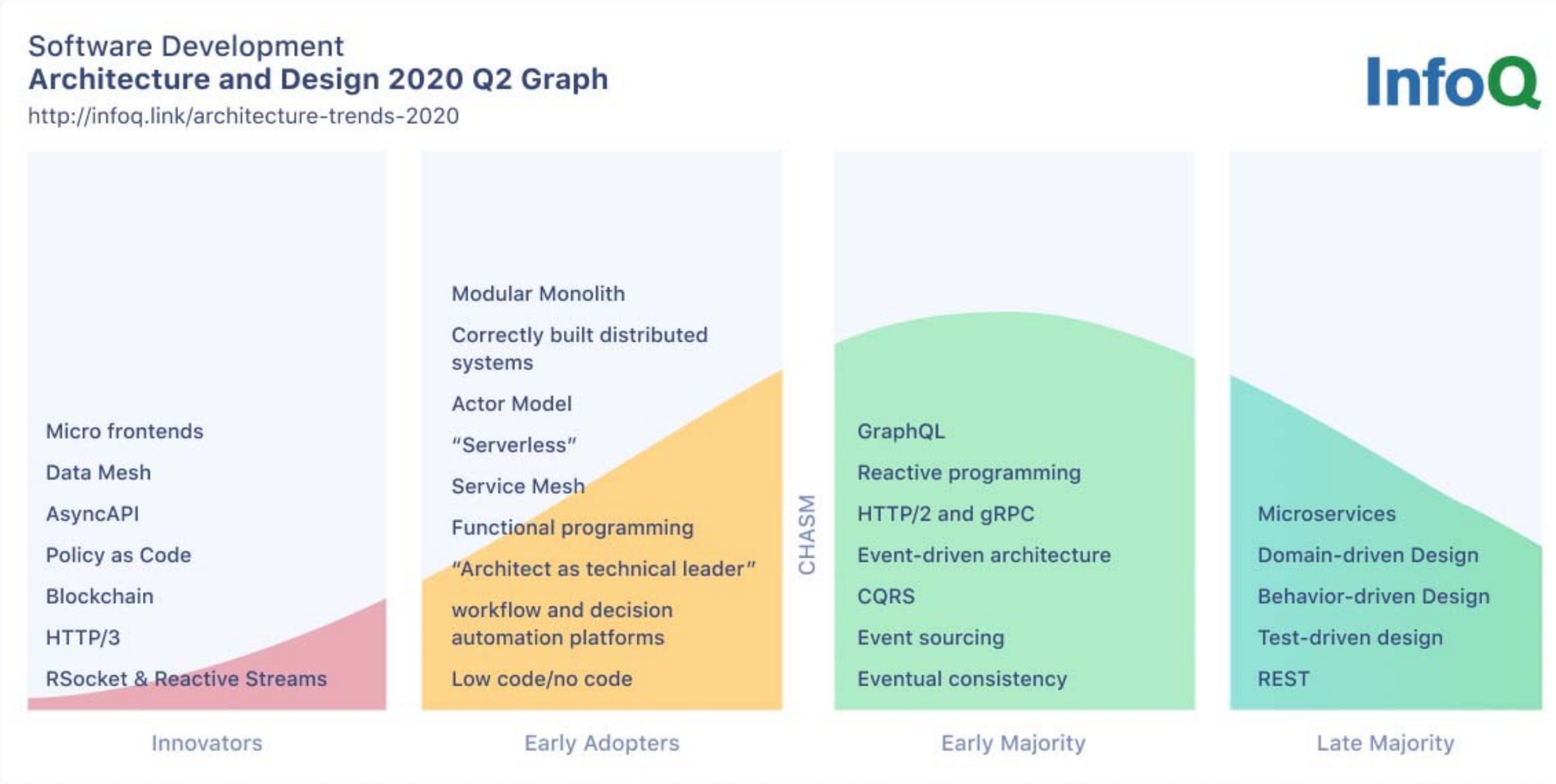
Microservices

"don't distribute your objects."

Martin Fowler

https://martinfowler.com/articles/distributed-objects-microservices.html

Alternatives?



https://www.infoq.com/articles/architecture-trends-2020/

Try not to need a Service Mesh

More Service Mesh

- **Service Mesh Comparison at servicemesh.es**
  https://servicemesh.es/
- **Blog Post: Happy without a Service Mesh**
  https://innoq.com/en/blog/happy-without-a-service-mesh/
- **Example-Application on GitHub**
  https://github.com/ewolff/microservice-istio
- **Linkerd Tutorial**
  https://linkerd.io/2/tasks/
- **Istio Tutorial**
  https://istio.io/docs/setup/getting-started/

Thank you!
Questions?

Hanna Prinz
hanna.prinz@innoq.com
@HannaPrinz

INNOQ
www.innoq.com

Service Mesh Primer - 2nd Edition
Free at leanpub.com/service-mesh-primer

innoQ Deutschland GmbH

Krischerstr. 100
40789 Monheim am Rhein
Germany
+49 2173 3366-0

Ohlauer Str. 43
10999 Berlin
Germany
+49 2173 3366-0

Ludwigstr. 180E
63067 Offenbach
Germany
+49 2173 3366-0

Kreuzstr. 16
80331 München
Germany
+49 2173 3366-0

Hermannstrasse 13
20095 Hamburg
Germany
+49 2173 3366-0

innoQ Schweiz GmbH

Gewerbestr. 11
CH-6330 Cham
Switzerland
+41 41 743 0116