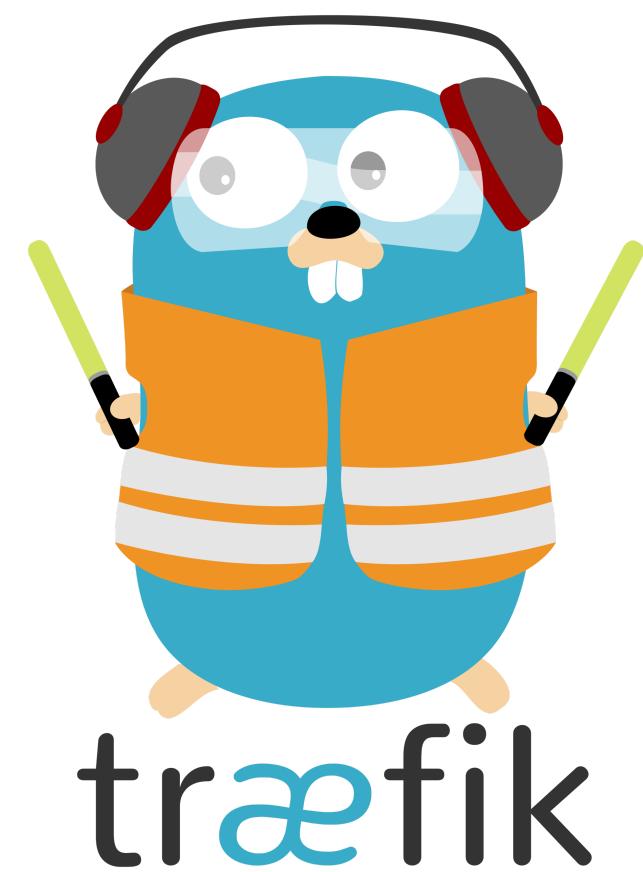


Ease Your Ingress Management And Observability With Traefik And Maesh



Containous & Nuaware

Whoami

Manuel Zapf

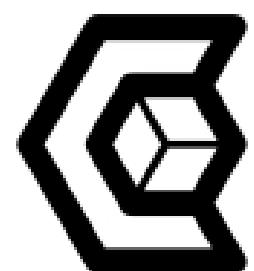
- Head of Product Open Source @ Containous
- Maintainer on Traefik
-  @mZapfDE
-  SantoDE



Containous

<https://containo.us>

- We Believe in Open Source
- We Deliver Traefik, Traefik Enterprise Edition, Maesh
- Commercial Support
- 30 people distributed, 90% tech



CONTAINOUS

Why Traefik



Why, Mr Anderson?

THE EVOLUTION OF
SOFTWARE ARCHITECTURE

1990's

SPAGHETTI-ORIENTED
ARCHITECTURE
(aka Copy & Paste)



2000's

LASAGNA-ORIENTED
ARCHITECTURE
(aka Layered Monolith)



2010's

RAVIOLI-ORIENTED
ARCHITECTURE
(aka Microservices)



WHAT'S NEXT?
PROBABLY PIZZA-ORIENTED ARCHITECTURE

By @benorama

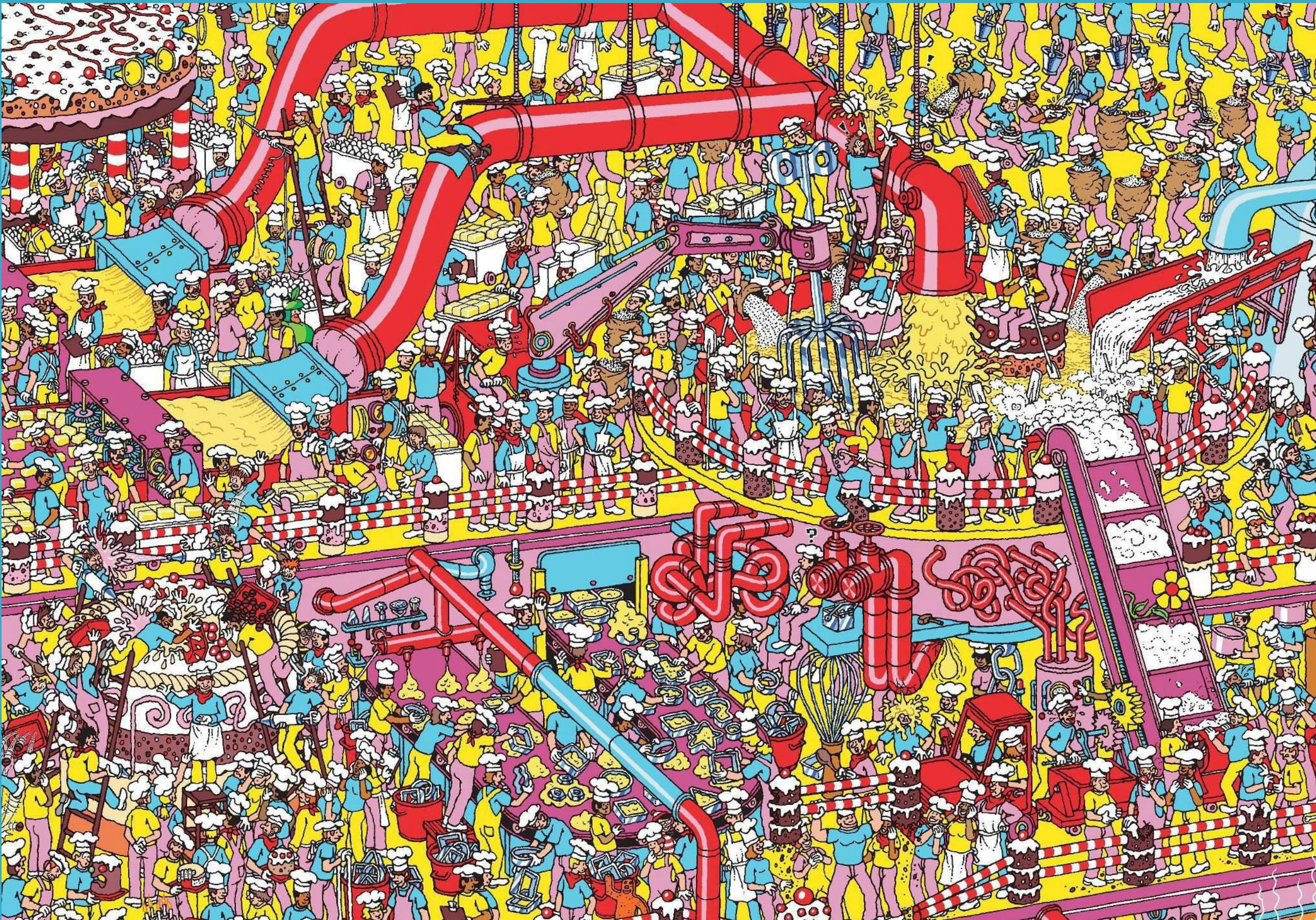
The Promise Of Microservices...



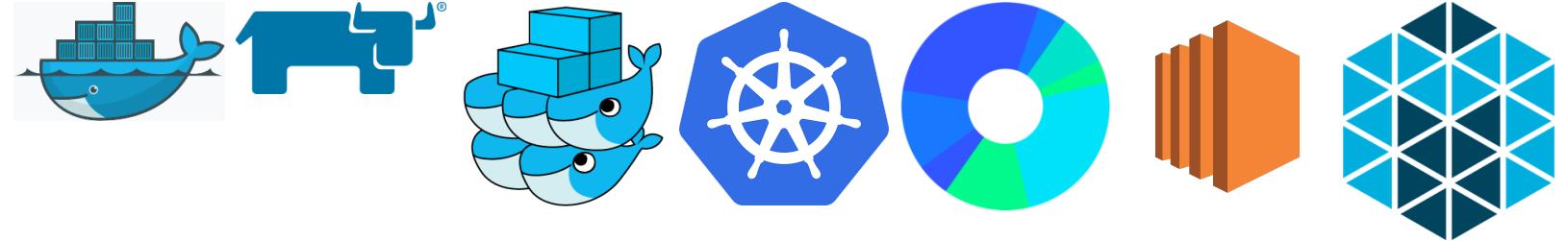
...And What Happens



Where's My Service?

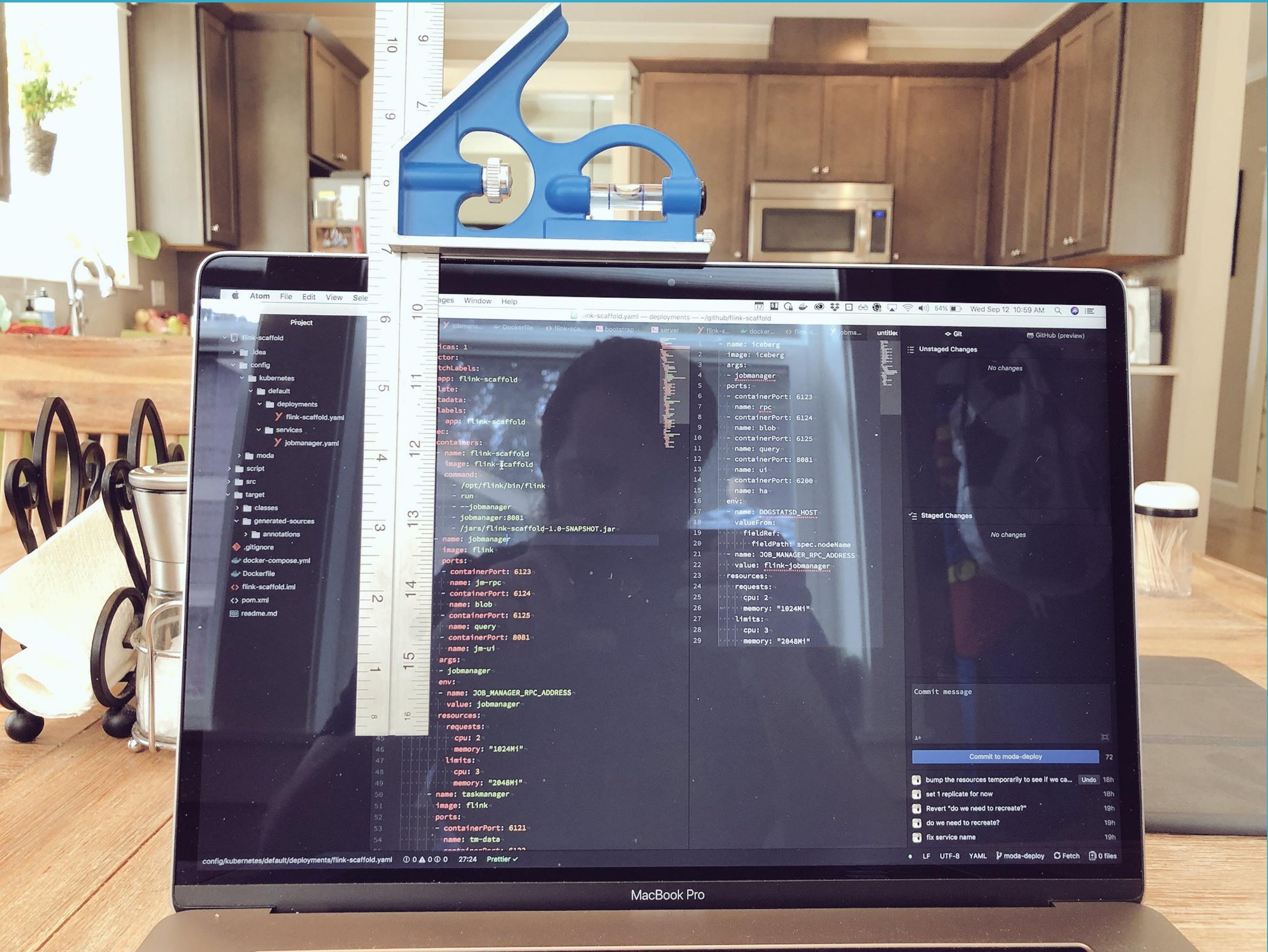


Tools Of The Trade



Common Problems

- Repetitive Configuration
- Pure amount hard to manage
- How to handle rewrites, redirects, tls...?



Source: <https://twitter.com/Caged/status/1039937162769096704>

What If I Told You?



That You Don't Have to Write This Configuration File...?

Here Comes Traefik!



Traefik Project



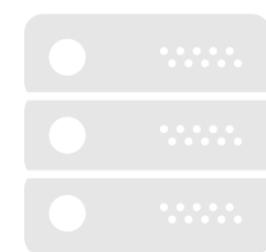
27,000+

stars on GitHub



450+

contributors



100K +
living
instances



1.4 B+

downloads



TOP 15

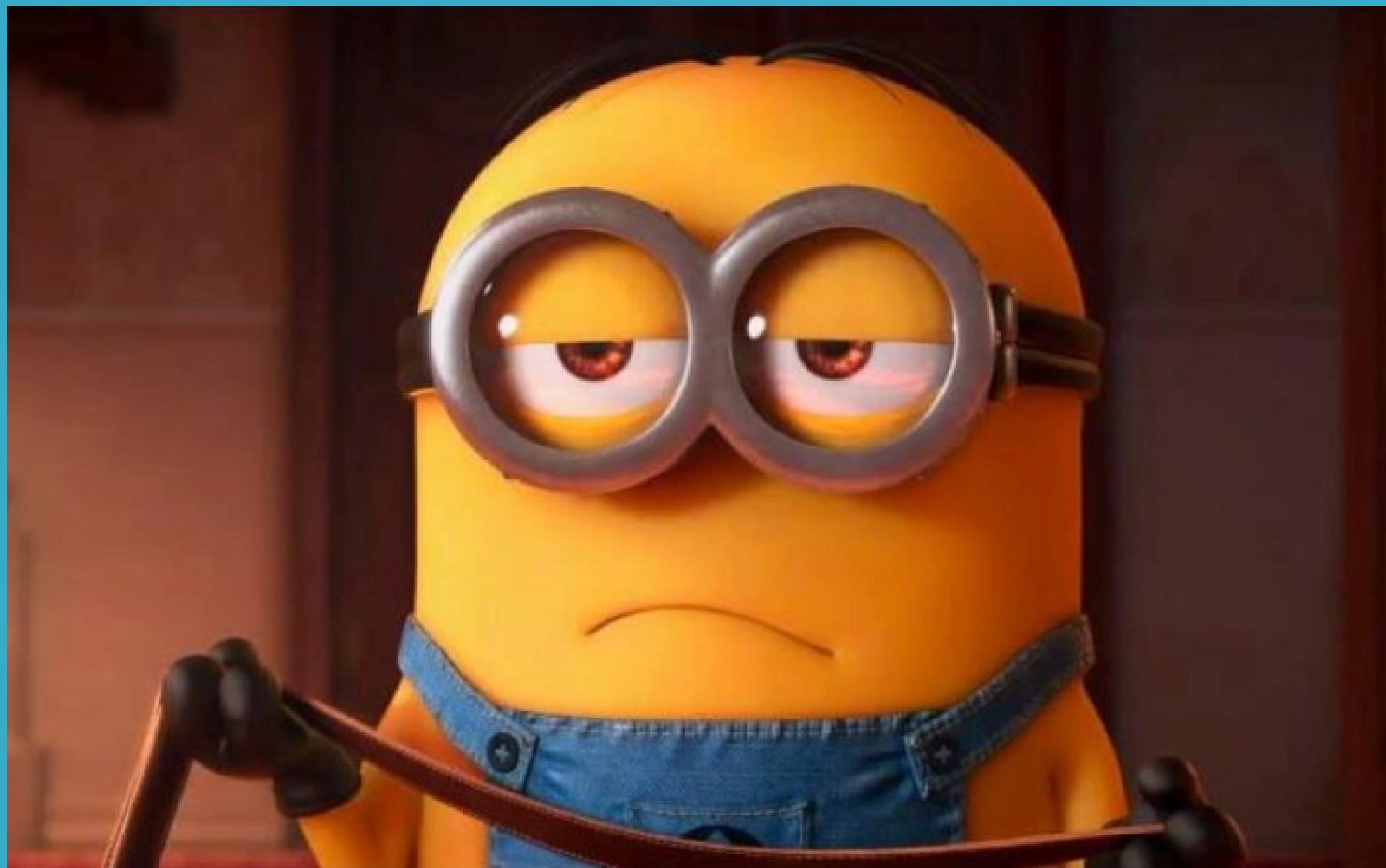
golang project

Traefik 2.0 Quick Overview

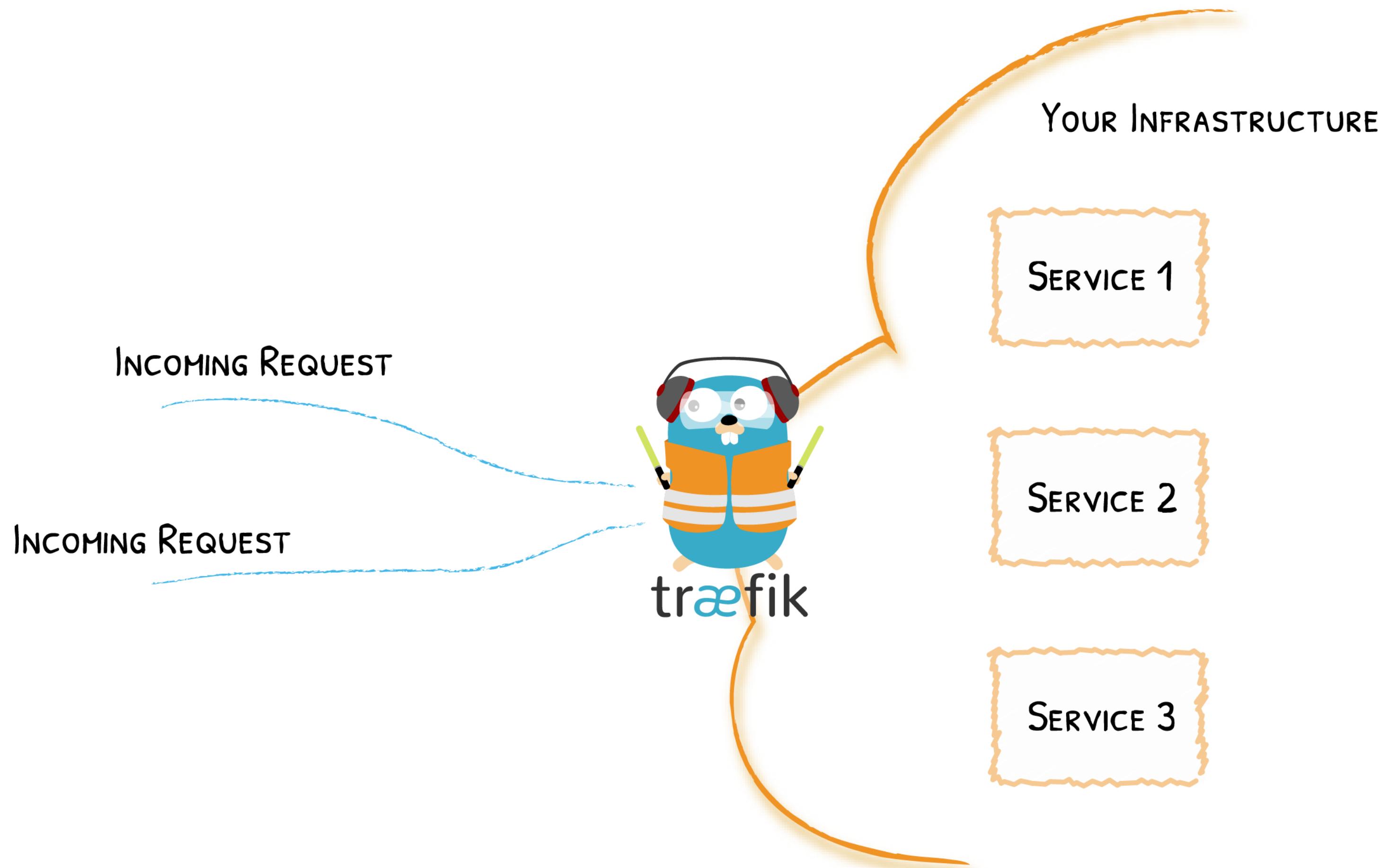
- Revamped Documentation
- Clarified Concepts
- Expressive Routing Rule Syntax
- Middlewares
- TCP Support
- Canary / Mirroring
- And so Much More...

Learn more on the blog post

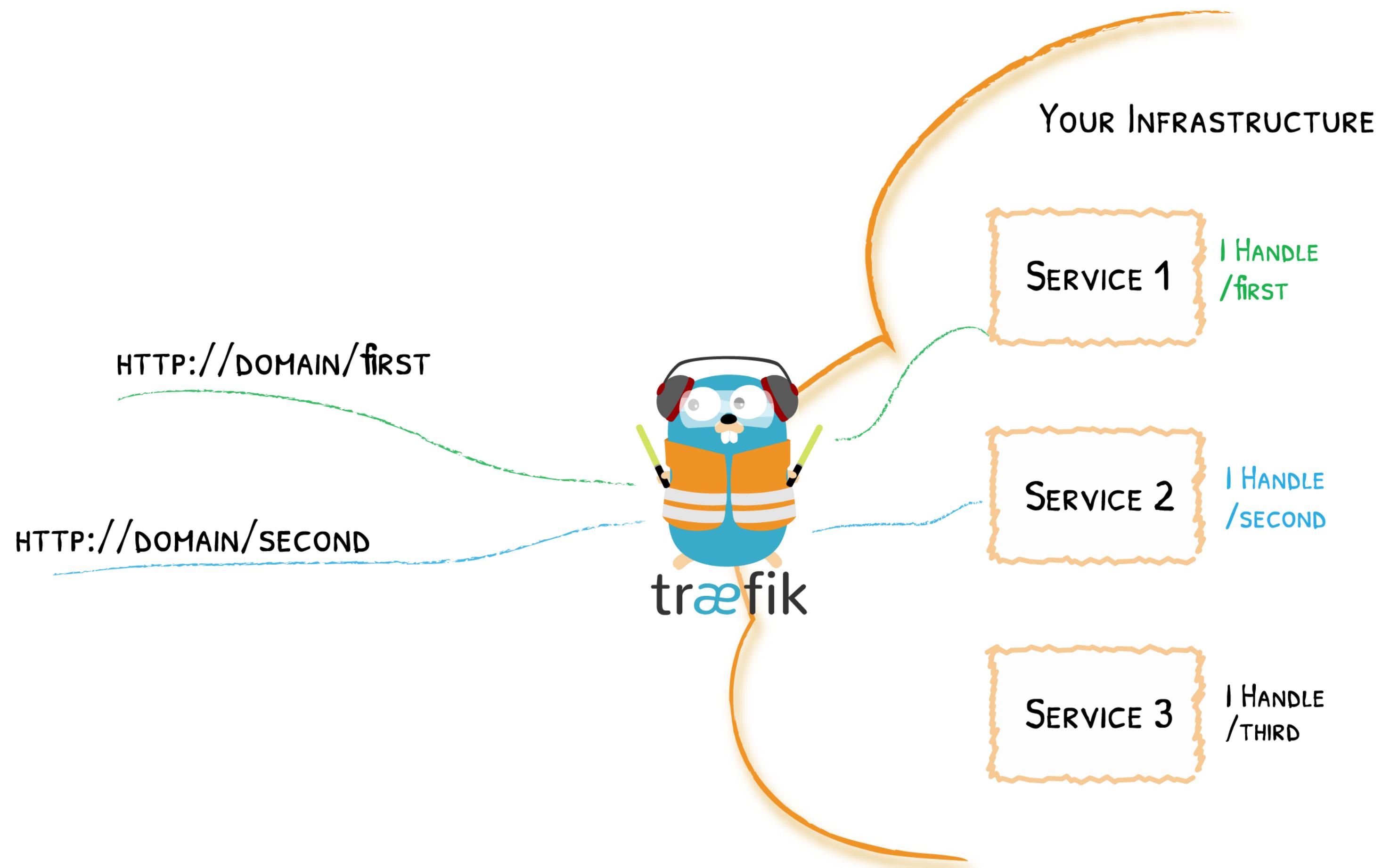
Traefik (V2.0) Core Concepts



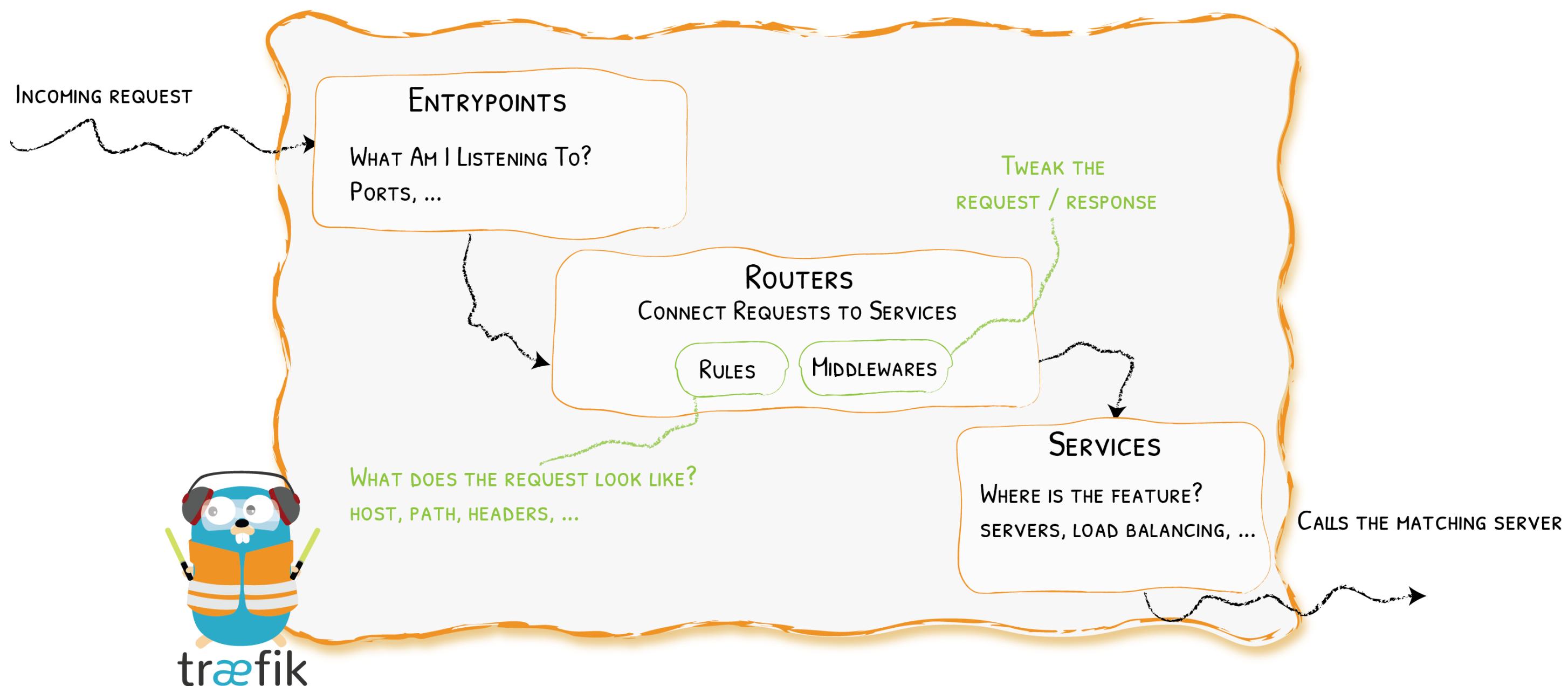
Traefik Is An Edge Router



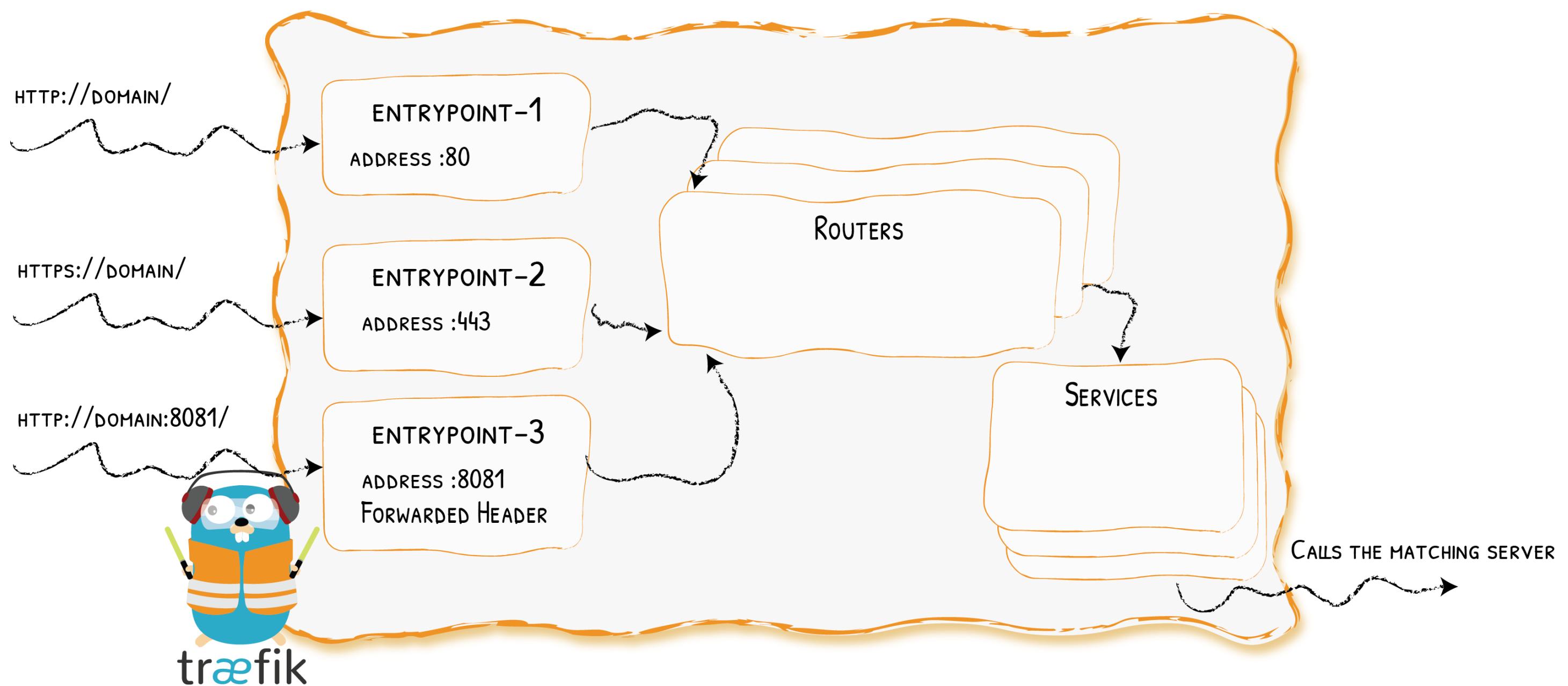
Dynamically Discovers Services



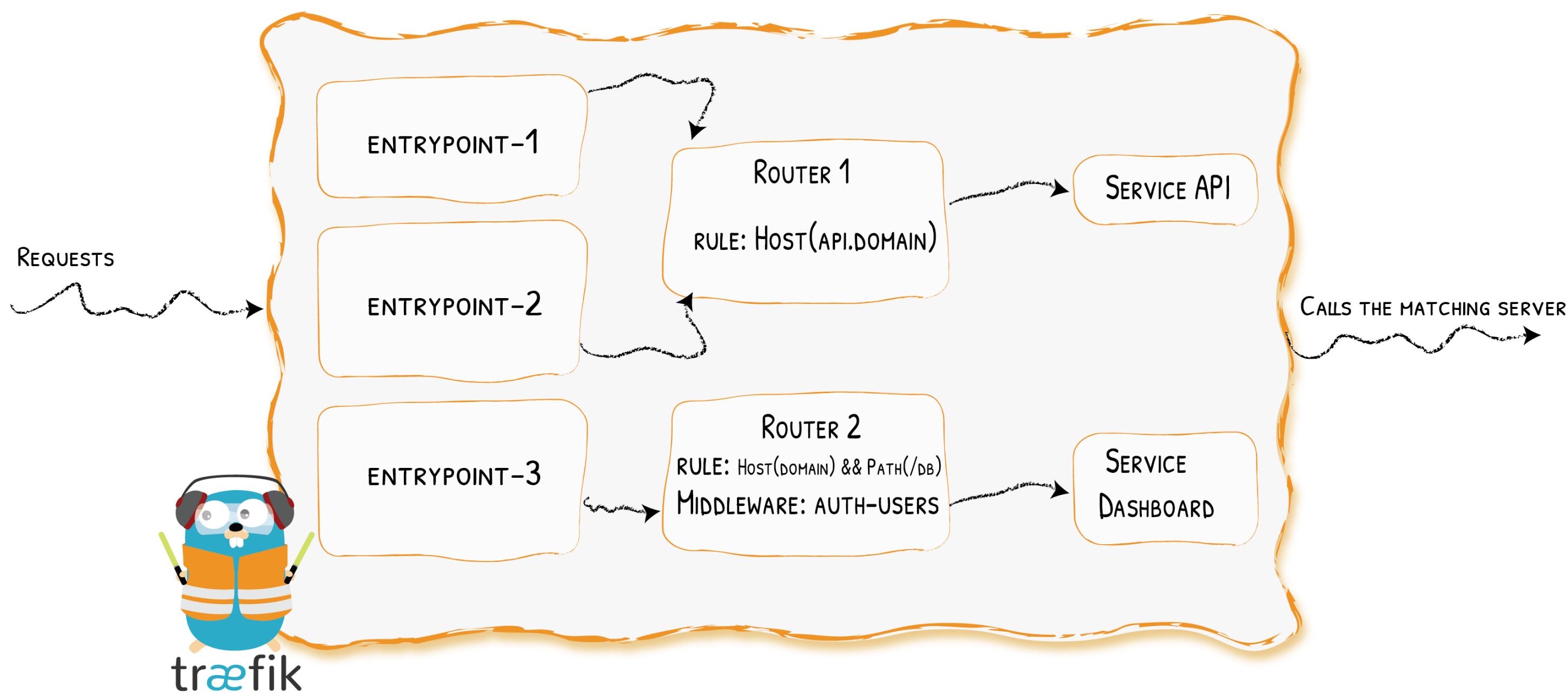
Architecture (V2.0) At A Glance



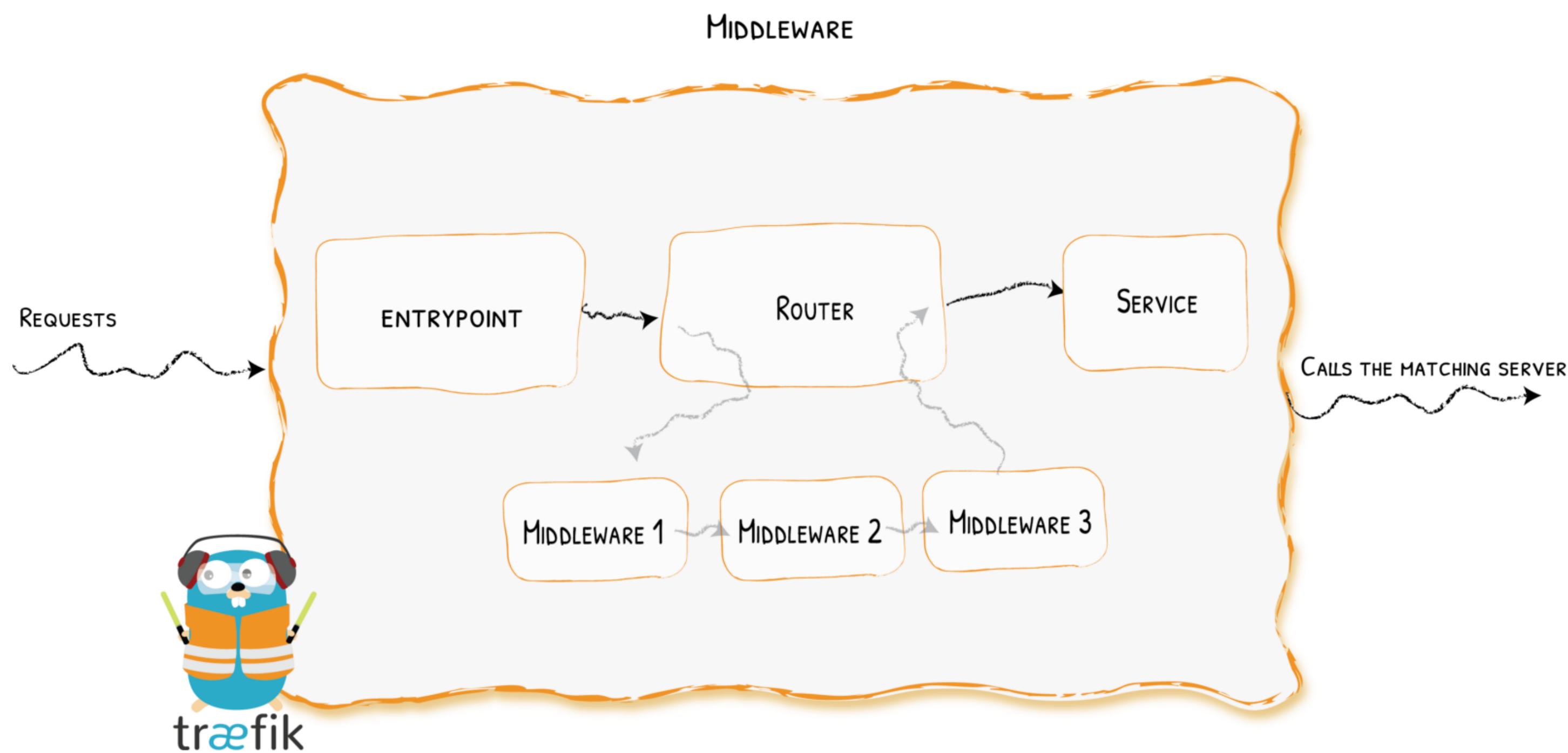
Entrypoints



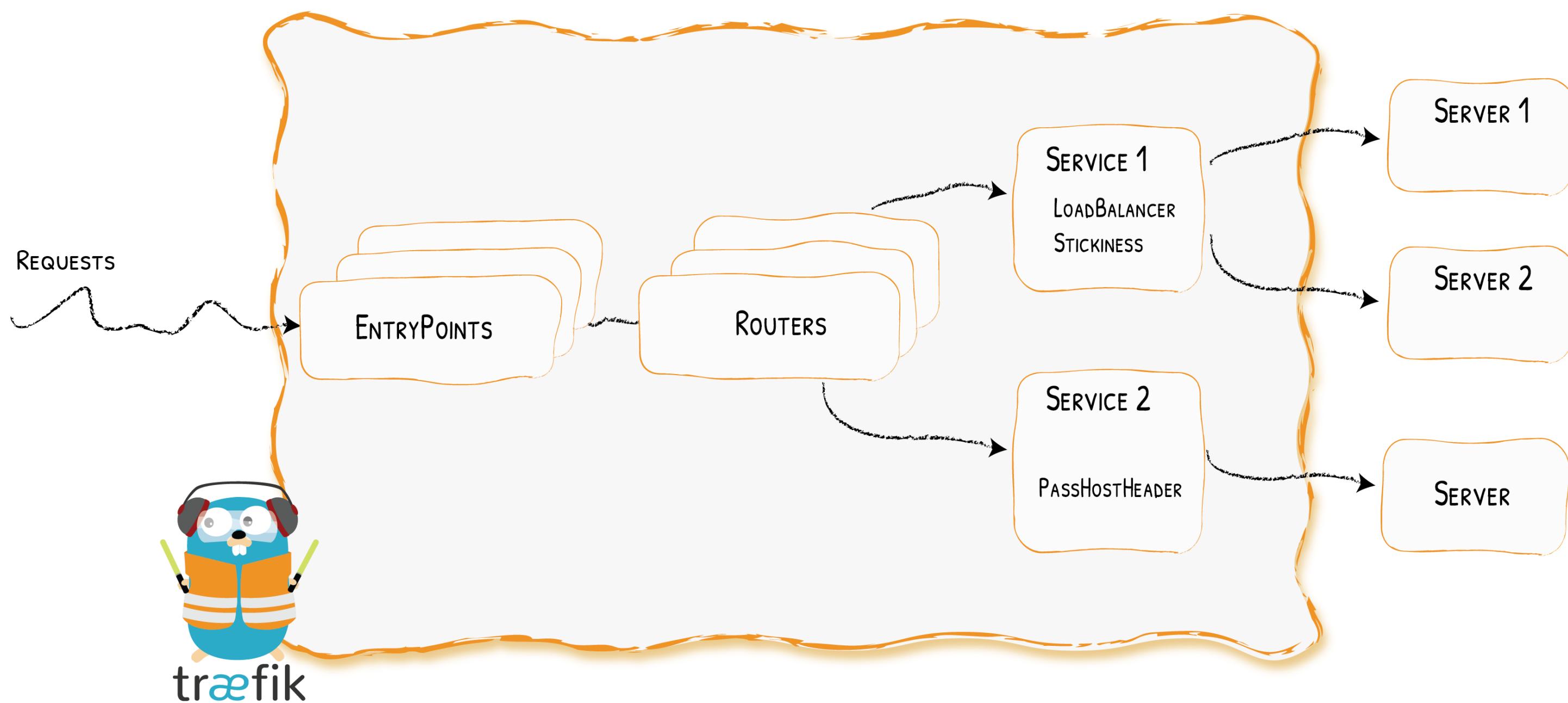
Routers



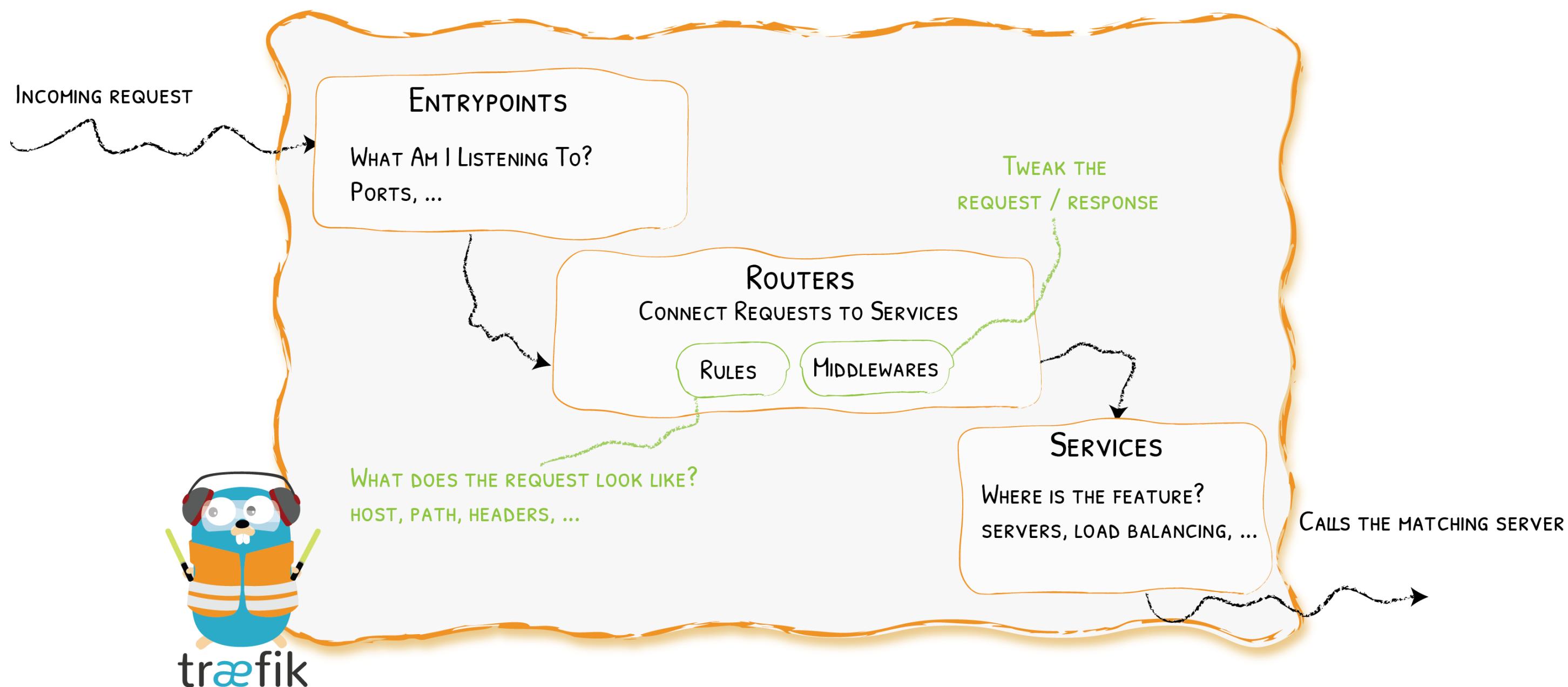
Middlewares



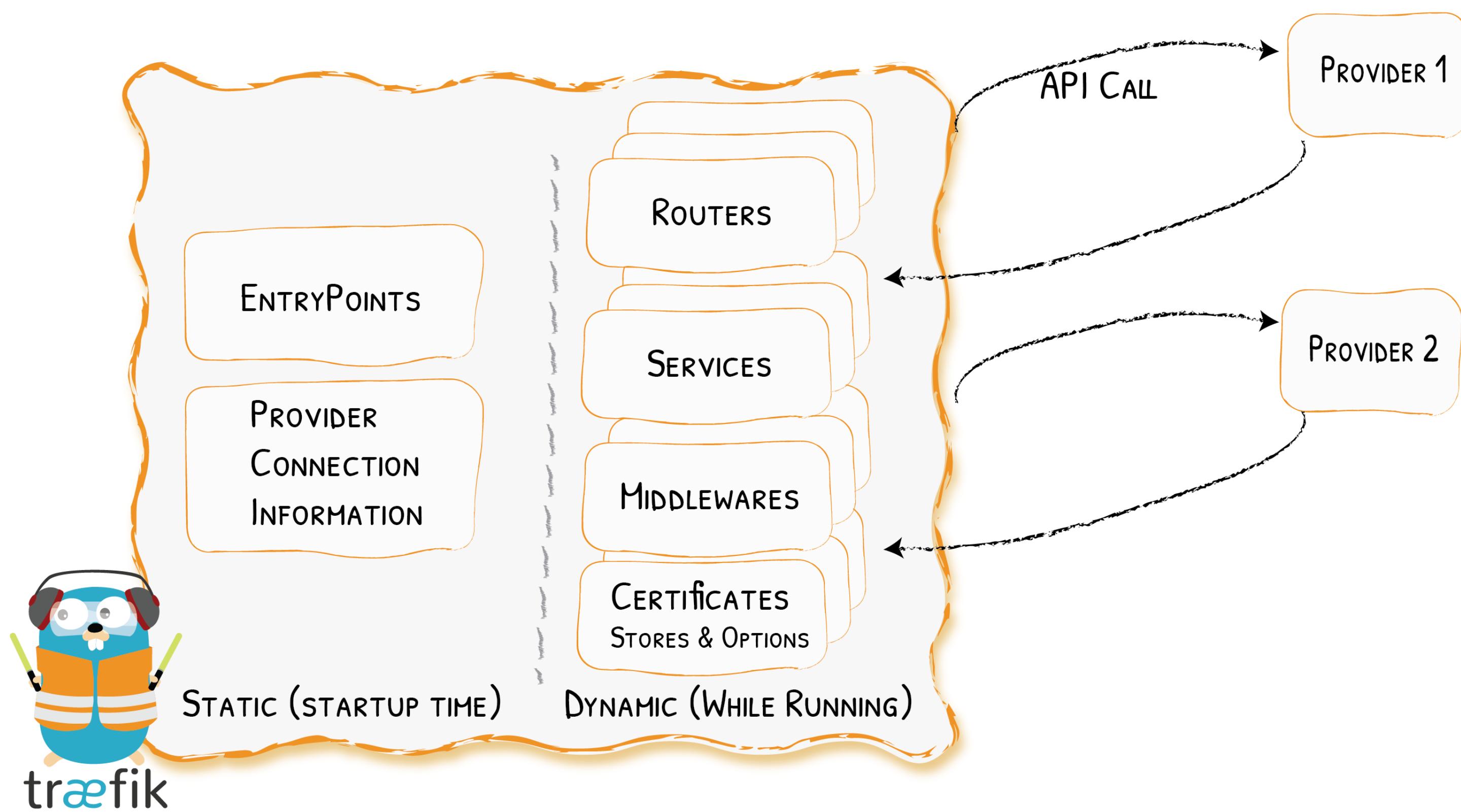
Services



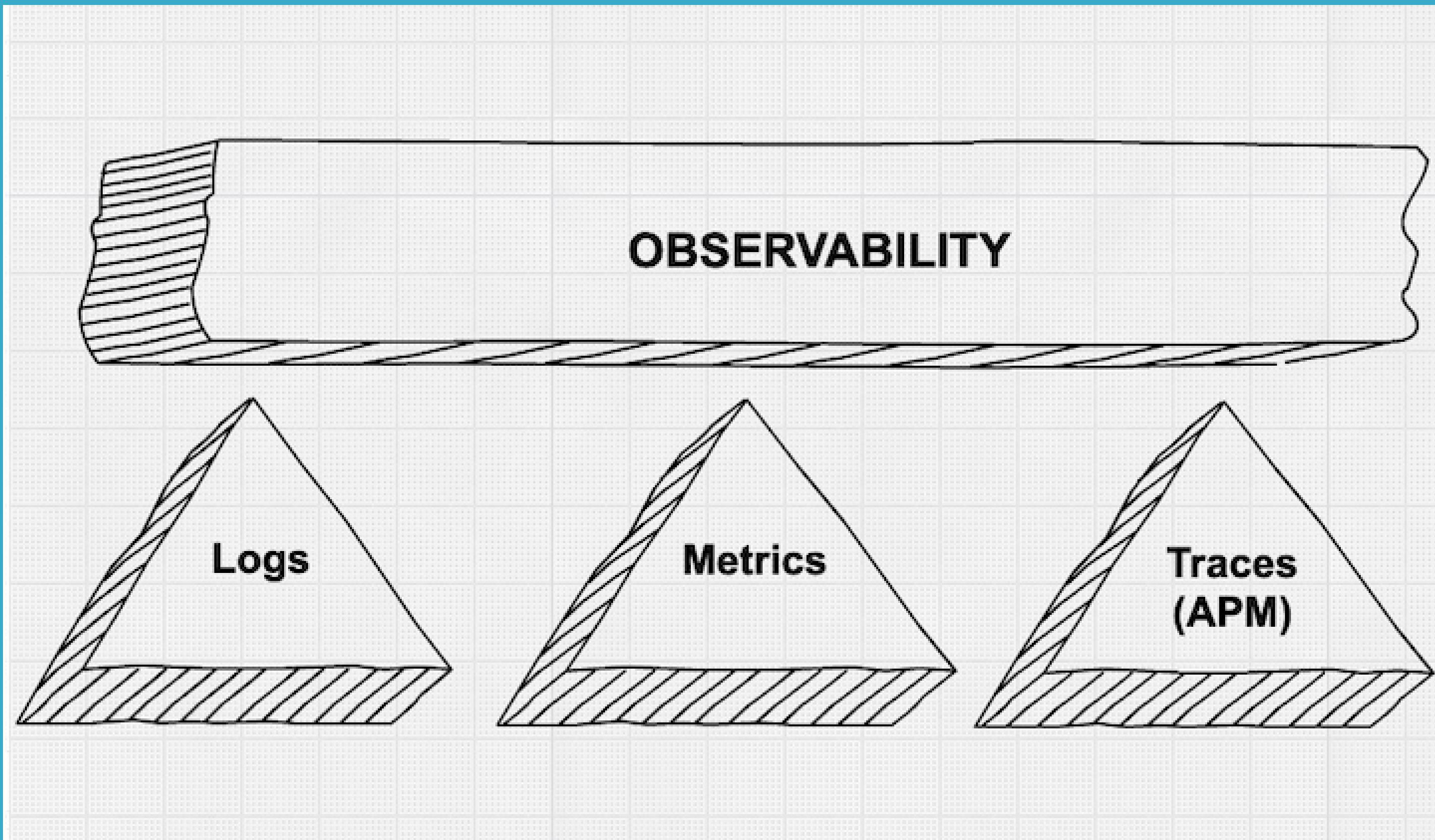
Architecture (Again) At A Glance



Static & Dynamic Configuration



Observability & Tracing With Traefik



Logs

- Difference between access / system logs
- Can you show internal behavior of your application
- Historical View

Metrics

- Ressource metrics (CPU / Memory / IO Performance...)
- Application metrics (Req/s, Open Connections...)
- Short-time period view

Tracing

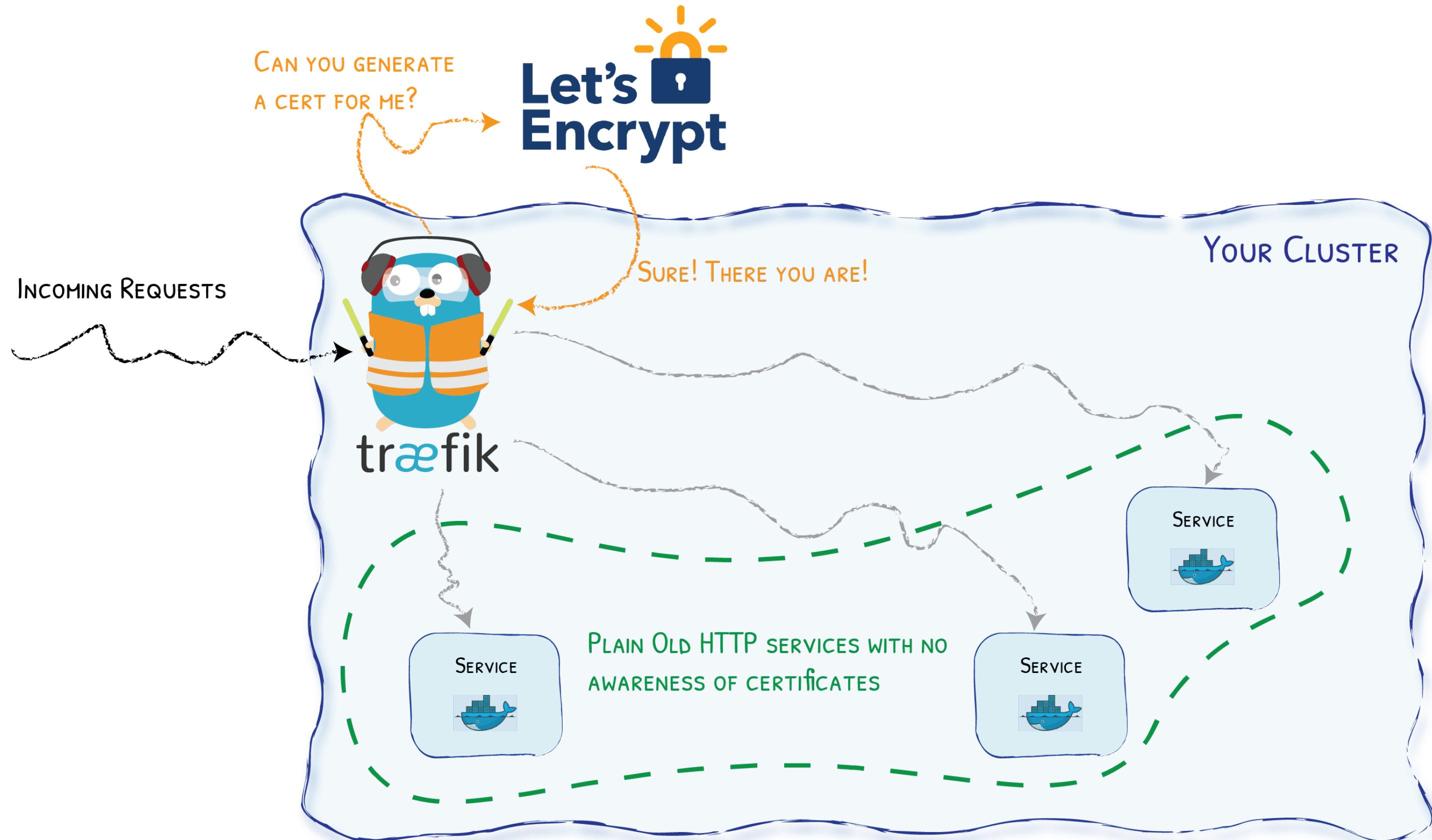
- Deep-Dive into how the app works
- Calltrace

Result

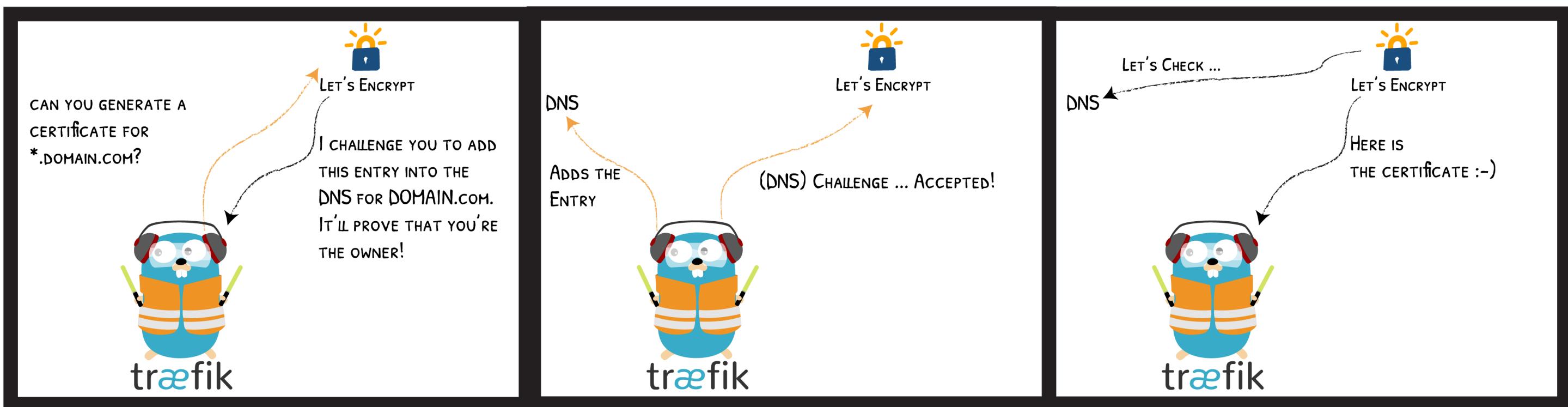
- Combining all 3
- Possibility to fully diagnose problems
- Combination of Logs / Metric might show misbehaving apps
- Tracing for finding root cause

Traefik And Let's Encrypt

HTTPS & Let's Encrypt



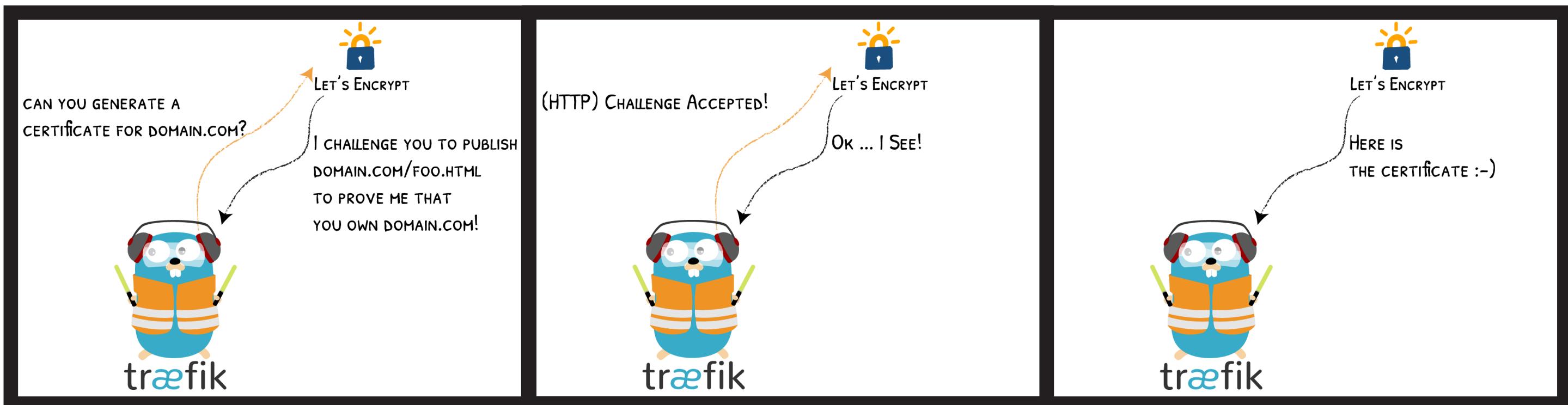
Let's Encrypt DNS Challenge



Let's Encrypt HTTP Challenge



Let's Encrypt TLS Challenge



Traefik With ⚓

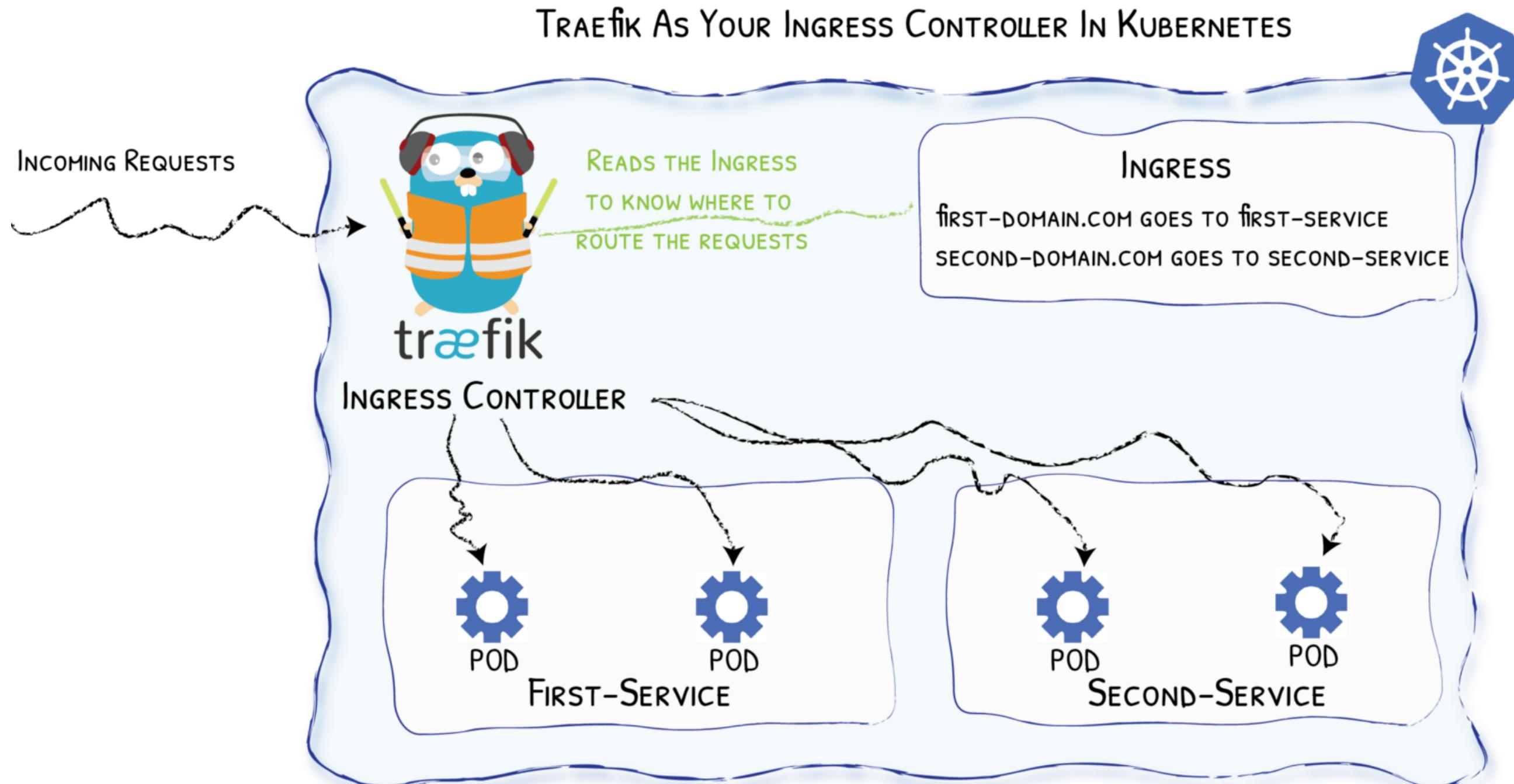


Diagram from <https://medium.com/@geraldcroes>

Ingress Example With ⚙

```
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
  name: corporate-webapp
  annotations:
    kubernetes.io/ingress.class: 'traefik'
spec:
  rules:
  - host: localhost
    http:
      paths:
      - backend:
          serviceName: corporate-webapp
          servicePort: 80
```

✳️ CRD - Custom Resources Definition

```
# File "webapp.yaml"
apiVersion: traefik.containo.us/v1alpha1
kind: IngressRoute
metadata:
  name: simpleingressroute
spec:
  entryPoints:
    - web
  routes:
    - match: Host(`localhost`) && PathPrefix(`/whoami`)
      kind: Rule
      services:
        - name: webapp
          port: 80
```

```
$ kubectl apply -f webapp.yaml
$ kubectl get ingressroute
```

✳️ & TCP (With CRD)

```
apiVersion: traefik.containo.us/v1alpha1
kind: IngressRouteTCP
metadata:
  name: ingressroutetcpmongo.crd
spec:
  entryPoints:
    - mongotcp
  routes:
    - match: HostSNI(`mongo-prod`)
      services:
        - name: mongo-prod
          port: 27017
```

Redirect HTTP To HTTPS

```
apiVersion: traefik.containo.us/v1alpha1
kind: Middleware
metadata:
  name: redirect-http
  namespace: webapp
spec:
  redirectScheme:
    scheme: https
```

Rewrite Path

```
apiVersion: traefik.containo.us/v1alpha1
kind: Middleware
metadata:
  name: redirect-http
  namespace: webapp
spec:
  redirectScheme:
    scheme: https
```

Redirect With Domain Replacement

```
apiVersion: traefik.containo.us/v1alpha1
kind: Middleware
metadata:
  name: test-redirectregex
spec:
  redirectRegex:
    regex: ^http://localhost/(.*)
    replacement: http://mydomain/${1}
```

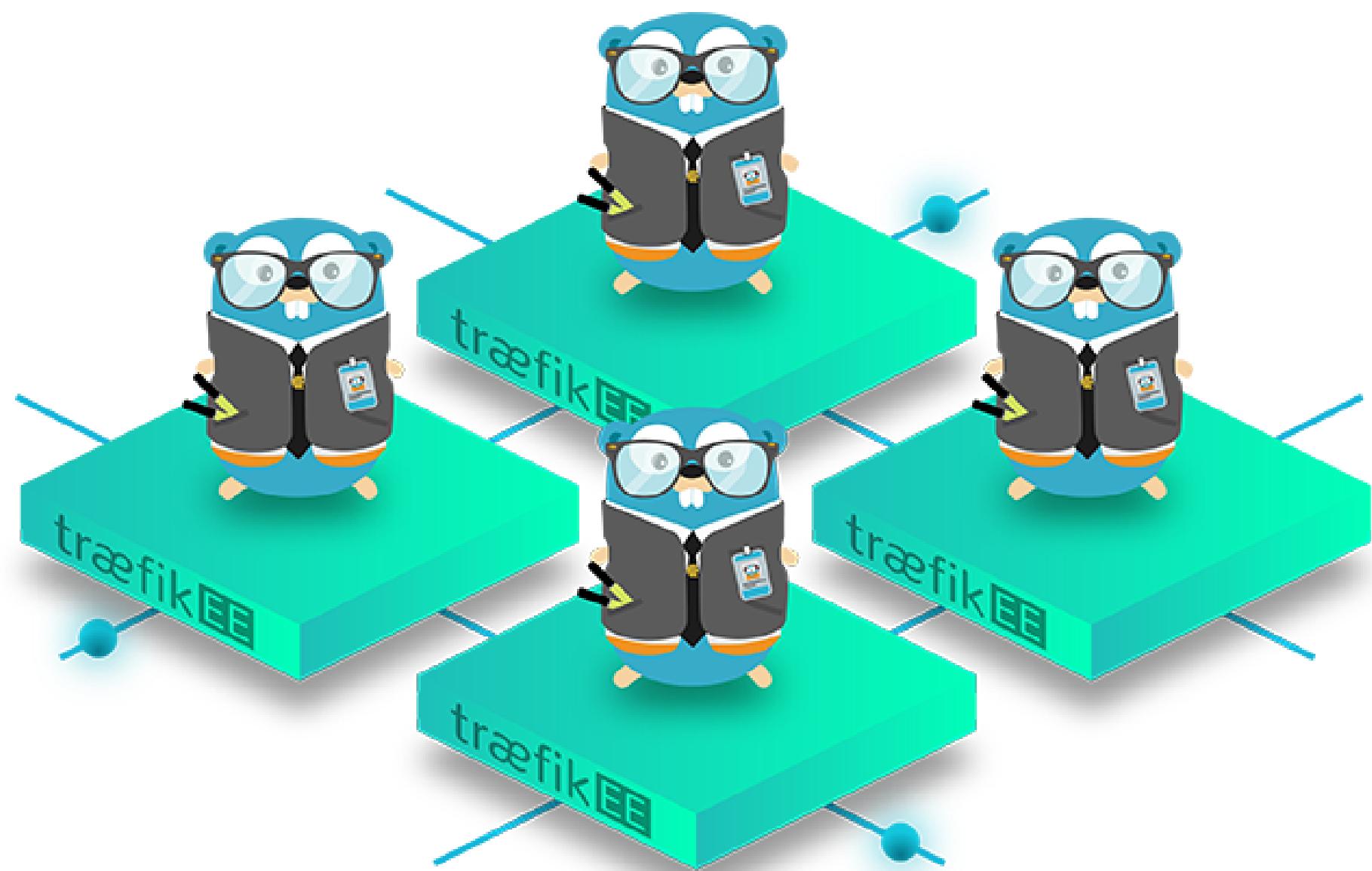
Strip Prefix

```
apiVersion: traefik.containo.us/v1alpha1
kind: Middleware
metadata:
  name: test-stripPrefix
spec:
  stripPrefix:
    prefixes:
      - /foobar
      - /fiibar
```

Canary

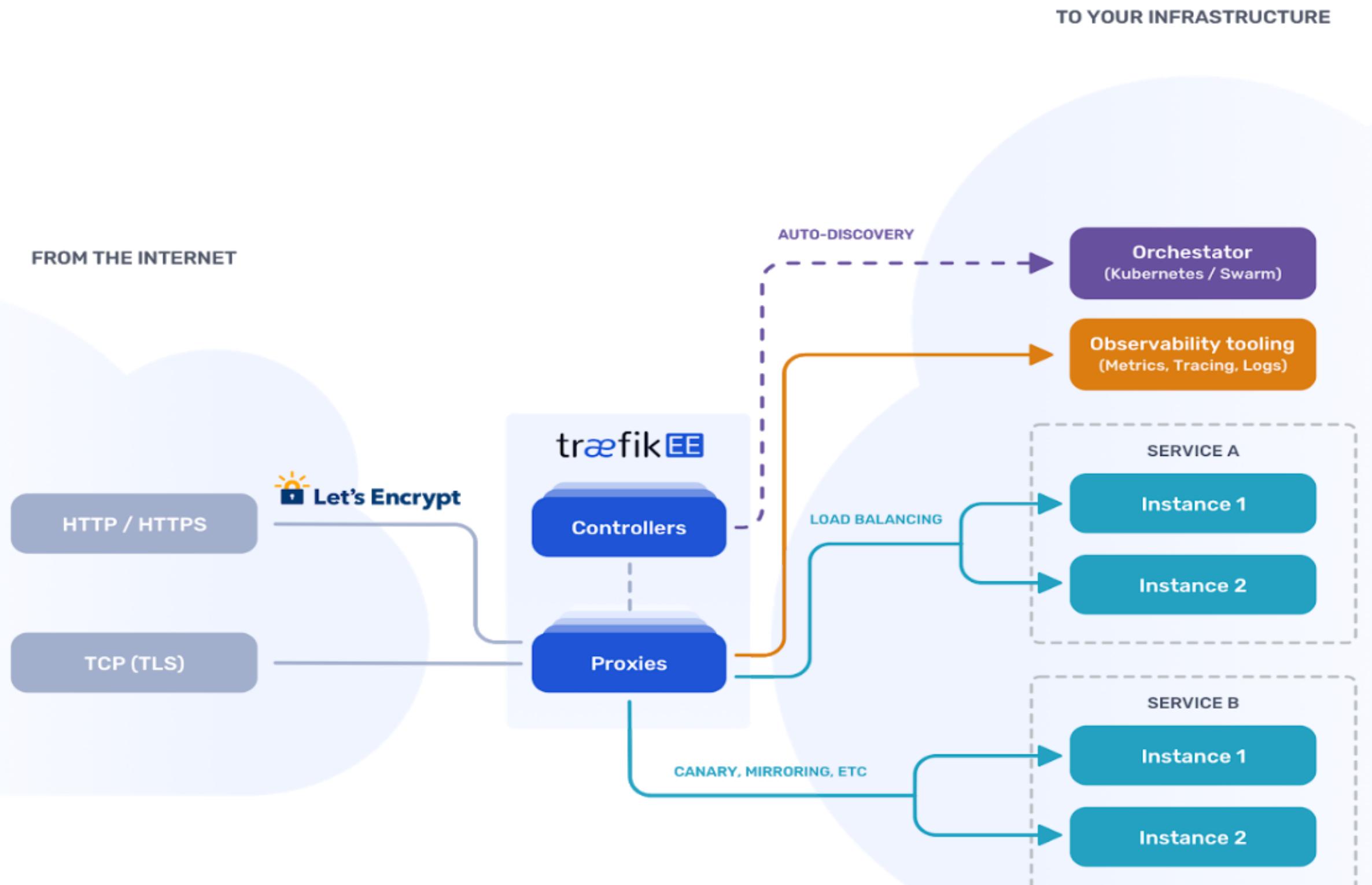
```
apiVersion: traefik.containo.us/v1alpha1
kind: TraefikService
metadata:
  name: webapp-canary
  namespace: webapp
spec:
  weighted:
    services:
      - name: webapp-v1
        weight: 3
        port: 80
        # Optional, as it is the default value
        kind: Service
      - name: webapp-v2
        weight: 1
        port: 80
        kind: Service
```

Traefik Also Comes In Herd



Advantages

- High availability by default
- Cluster consensus build on Raft
- Shared State
- Distributed Middlewares
- Distributed Lets Encrypt



UI

The screenshot shows the TraefikEE v2.0.0-rc1 dashboard. At the top, there are links for Dashboard, HTTP, TCP, License, Documentation, and TraefikEE v2.0.0-rc1. The main interface is divided into several sections:

- Controllers:** Shows three controllers: default-controller-0 (IP: controller-0.traefikee, STATUS: Ready), default-controller-2 (IP: controller-1.traefikee, STATUS: Ready), and default-controller-3 (IP: controller-3.traefikee, STATUS: Ready).
- Proxies:** Shows three proxies: default-proxy-654b56ffdc-4xdvs (IP: 100.96.2.36, STATUS: Disconnected), default-proxy-654b56ffdc-hcpwb (IP: 100.96.3.84, STATUS: Down), and default-proxy-654b56ffdc-zw2w4 (IP: 100.96.3.85, STATUS: Ready).
- Entrypoints:** A grid of ports:

HTTP :8080	HTTPS :8080	TRAEFIK :8080	WEB :8000	WEB-MTLS :8443	WEB-REDIRECT :80
WEB-SECURED :443	WEB-TCP :8100	WEB2 :8002	WEB3 :8003	WEB4 :8004	
- HTTP Metrics:** Routers, Services, and Middlewares status charts with 42 entries each.
- TCP Metrics:** Routers, Services status charts with 42 entries each.



træfikEE

CONTAINOUS

UI



All types of companies

All over the world

All industries



PrestaShop



As Simple As Traefik

- Install it:

```
# Cluster Installation
traefikeectl install \
--licensekey="SuperSecretLicence" \
--dashboard \
--kubernetes # Or --swarm
```

- Configure it:

```
# Routing Configuration, same as Traefik's
traefikeectl deploy \
--defaultentrypoints=http,https \
--entryPoints='Name:http Address::80' \
--entryPoints='Name:https Address::443 TLS' \
--logLevel=INFO \
--kubernetes
```

Pricing

Professional



2 Data Nodes
& 1 Control Node

8-hour SLA (weekdays)

\$5 000 annual
subscription

Enterprise



10 Data Nodes
& 1 Control Node

8-hour SLA (weekdays)

\$10 000 annual
subscription

Mission Critical



10 Data Nodes
& 7 Control Nodes

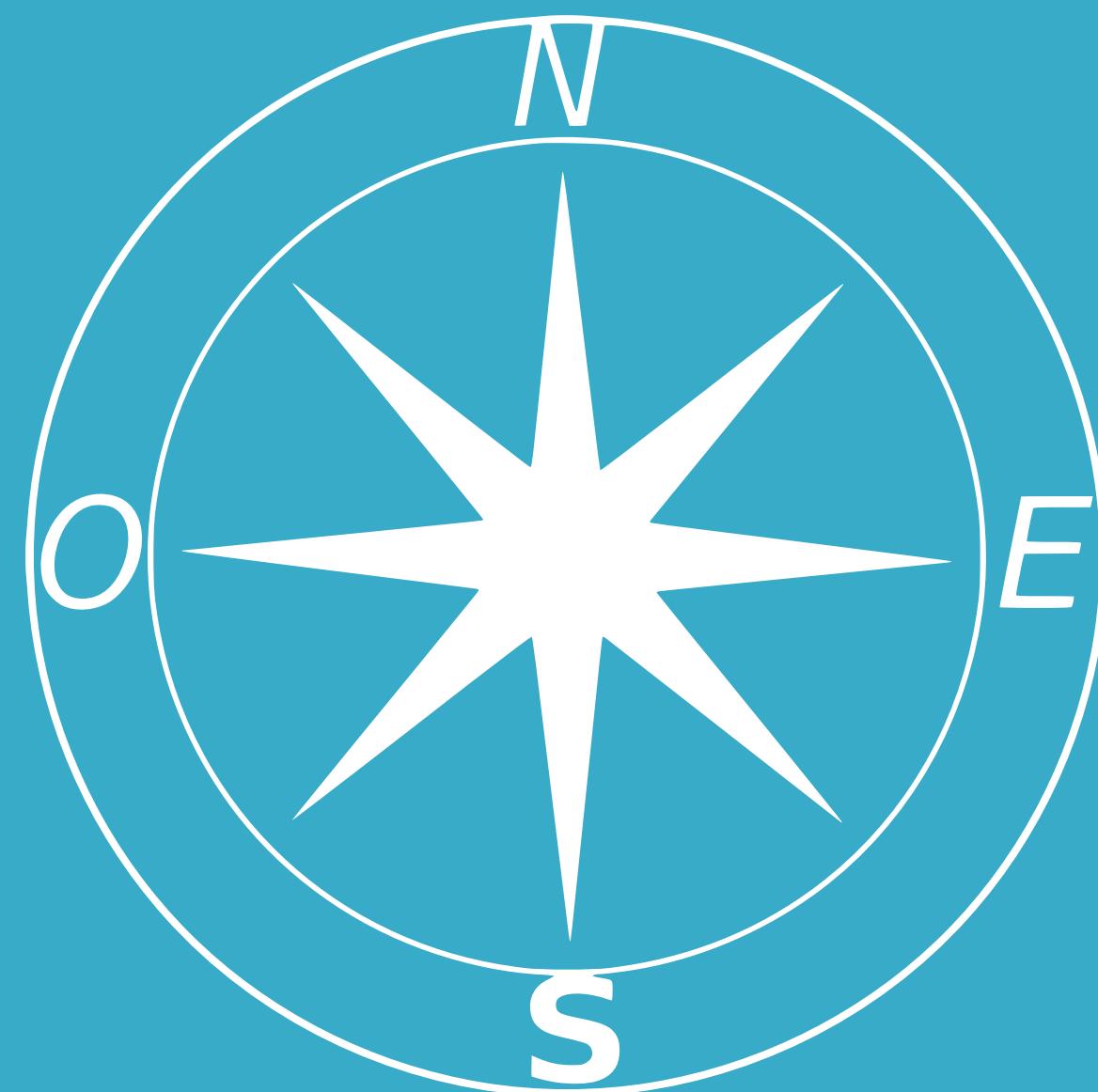
2-hour SLA

\$30 000 annual
subscription

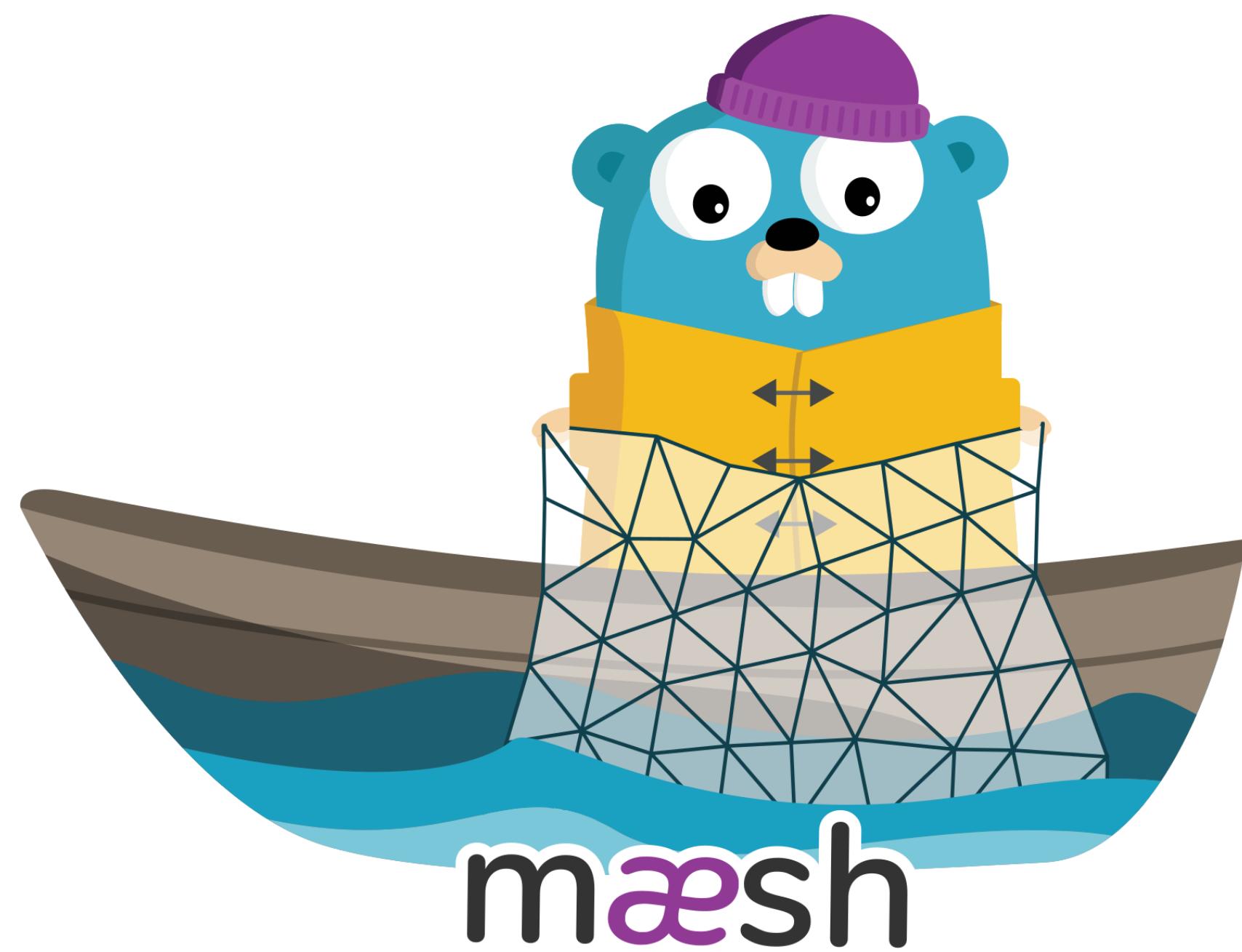
Free Trial

<https://containo.us/traefikee>

East / West Traefik



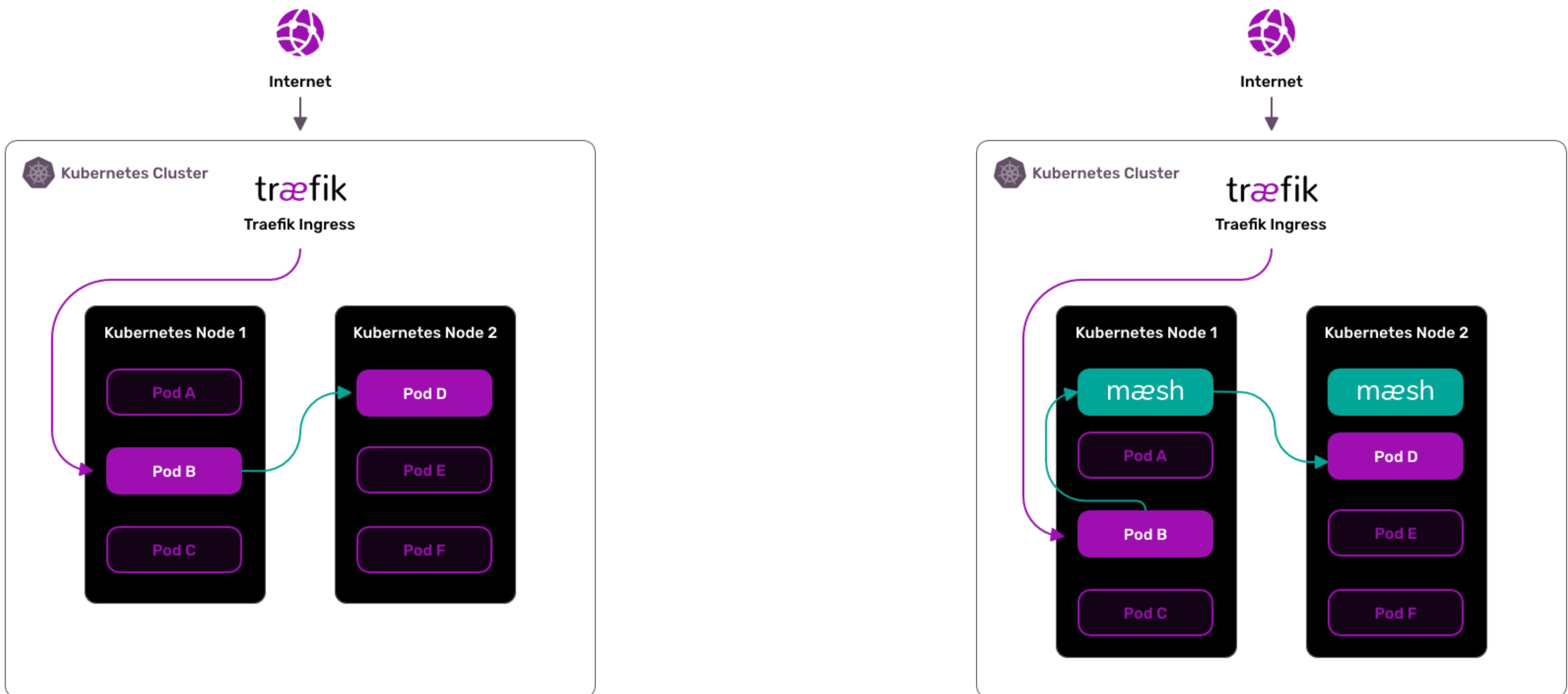
Say Hello To Maesh



What Is Maesh?

Maesh is a lightweight, easy to configure, and non-invasive service mesh that allows visibility and management of the traffic flows inside any Kubernetes cluster.

Maesh Architecture



More On Maesh

- Built on top of Traefik,
- SMI (Service Mesh Interface specification) compliant,
- Opt-in by default.

Maesh Website

Show Me The Code!

- Install Maesh (Helm Chart):

```
helm repo add maesh https://containous.github.io/maesh/charts  
helm repo update  
helm install --name=maesh --namespace=maesh maesh/maesh --values=./maesh/values.yaml
```

- Deploy Applications:

```
kubectl apply -f apps/0-namespace.yaml  
kubectl apply -f apps/1-svc-accounts.yaml  
kubectl apply -f apps/2-apps-client.yaml  
kubectl apply -f apps/3-apps-servers.yaml  
kubectl apply -f apps/4-ingressroutes.yaml
```

- Deploy SMI Objects to allow traffic in the mesh:

```
kubectl apply -f apps/5-smi-http-route-groups.yaml  
kubectl apply -f apps/6-smi-traffic-targets.yaml
```

A Closer Look To SMI Objects

```
apiVersion: specs.smi-spec.io/v1alpha1
kind: HTTPRouteGroup
metadata:
  name: app-routes
  namespace: apps
matches:
- name: all
  pathRegex: "/"
  methods: [ "*" ]
---
apiVersion: access.smi-spec.io/v1alpha1
kind: TrafficTarget
metadata:
  name: client-apps
  namespace: apps
destination:
  kind: ServiceAccount
  name: apps-server
  namespace: apps
specs:
- kind: HTTPRouteGroup
  name: app-routes
  matches:
  - all
sources:
- kind: ServiceAccount
  name: apps-client
  namespace: apps
```

That's All Folks!



We Have
Stickers!

traefik

We Are Hiring!

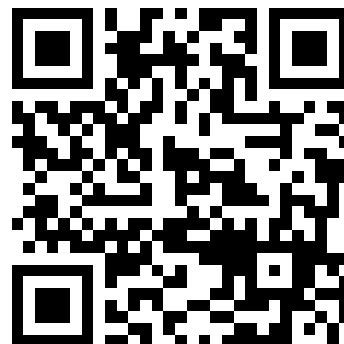


CONTAINOUS

```
docker run -it containous/jobs
```

Thank You!

-  @mZapfDE
-  SantoDE



- Slides (HTML): <https://containous.github.io/slides/containous-nuaware-london-2020>
- Slides (PDF): <https://containous.github.io/slides/containous-nuaware-london-2020/slides.pdf>
- Source on : <https://github.com/containous/slides/tree/containous-nuaware-london-2020>