Traefik - The Cloud Native Edge Router

Traefik In Action!

How To Use These Slides?

- **Browse the slides:** Use the arrows
  - Change chapter: Left/Right arrows
  - Next or previous slide: Top and bottom arrows
- **Overview of the slides:** keyboard's shortcut "o"
- **Speaker mode (and notes):** keyboard's shortcut "s"

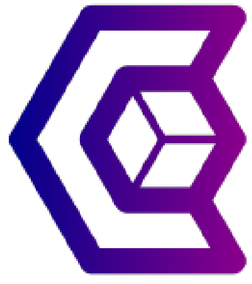
Whoami

- Manuel Zapf:
  - Træfik's Solution Architect 🔨 @ Containous
- 🐦 @mZapfDE
- SantoDE

Containous

https://containo.us

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



Why Traefik?

Why, Mr Anderson?

Evolution Of Software Design

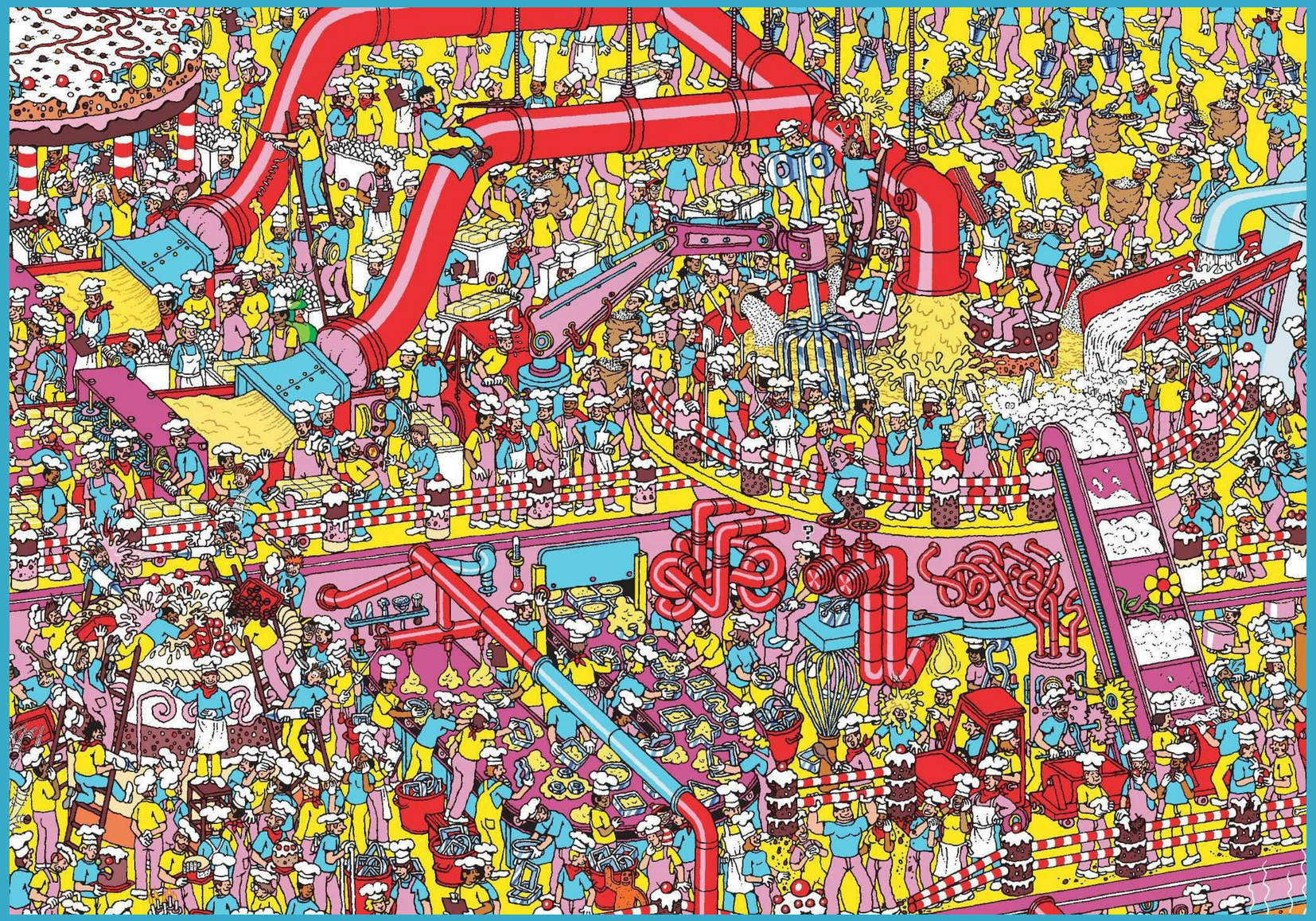
1990s and earlier	2000s	2010s
Coupling		
Pre-SOA (monolithic)	**Traditional SOA**	**Microservices**
Tight coupling	Looser coupling	Decoupled

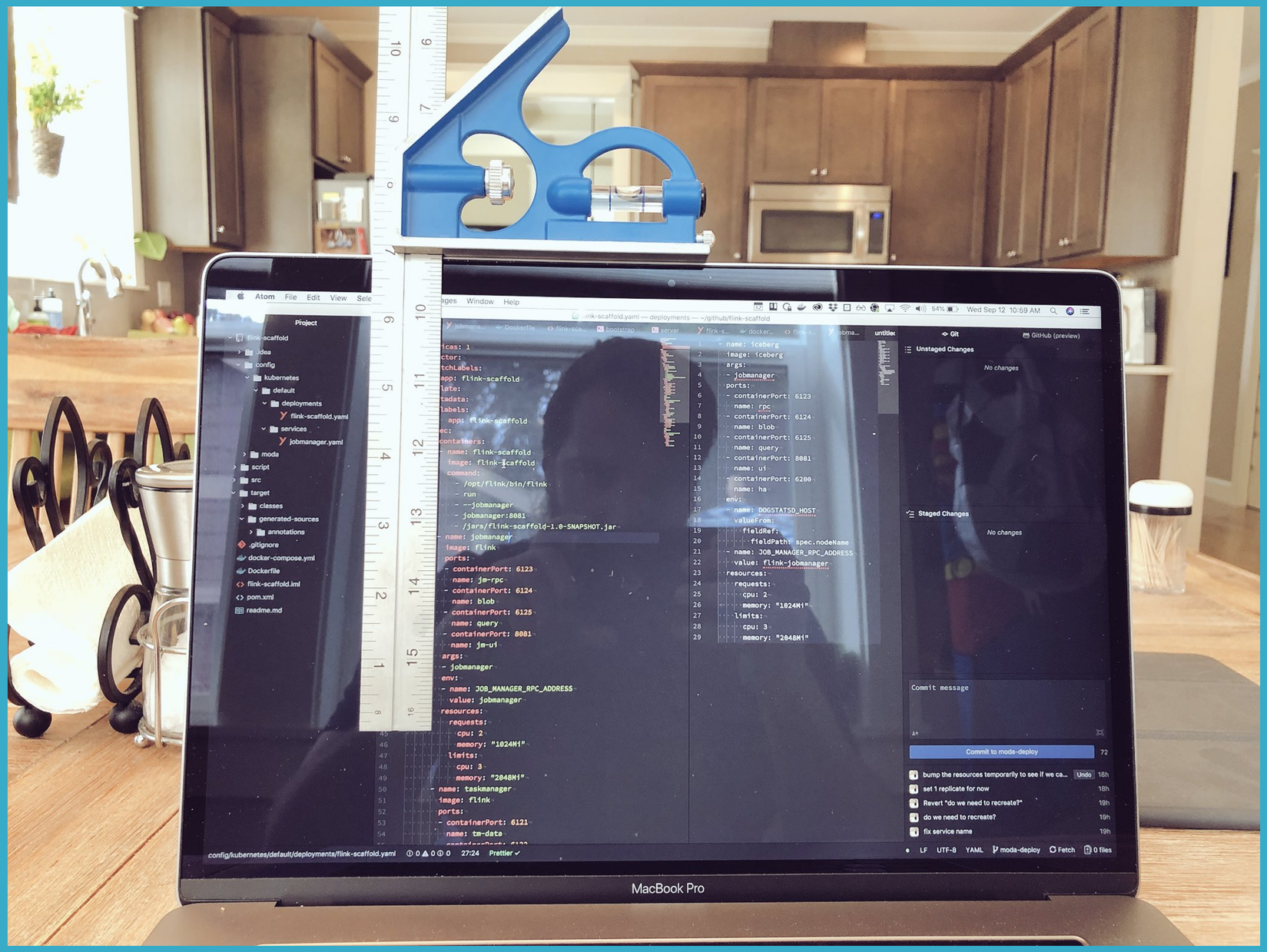
The Premise Of Microservices...

...And What Happens

Tools Of The Trade

Where's My Service?



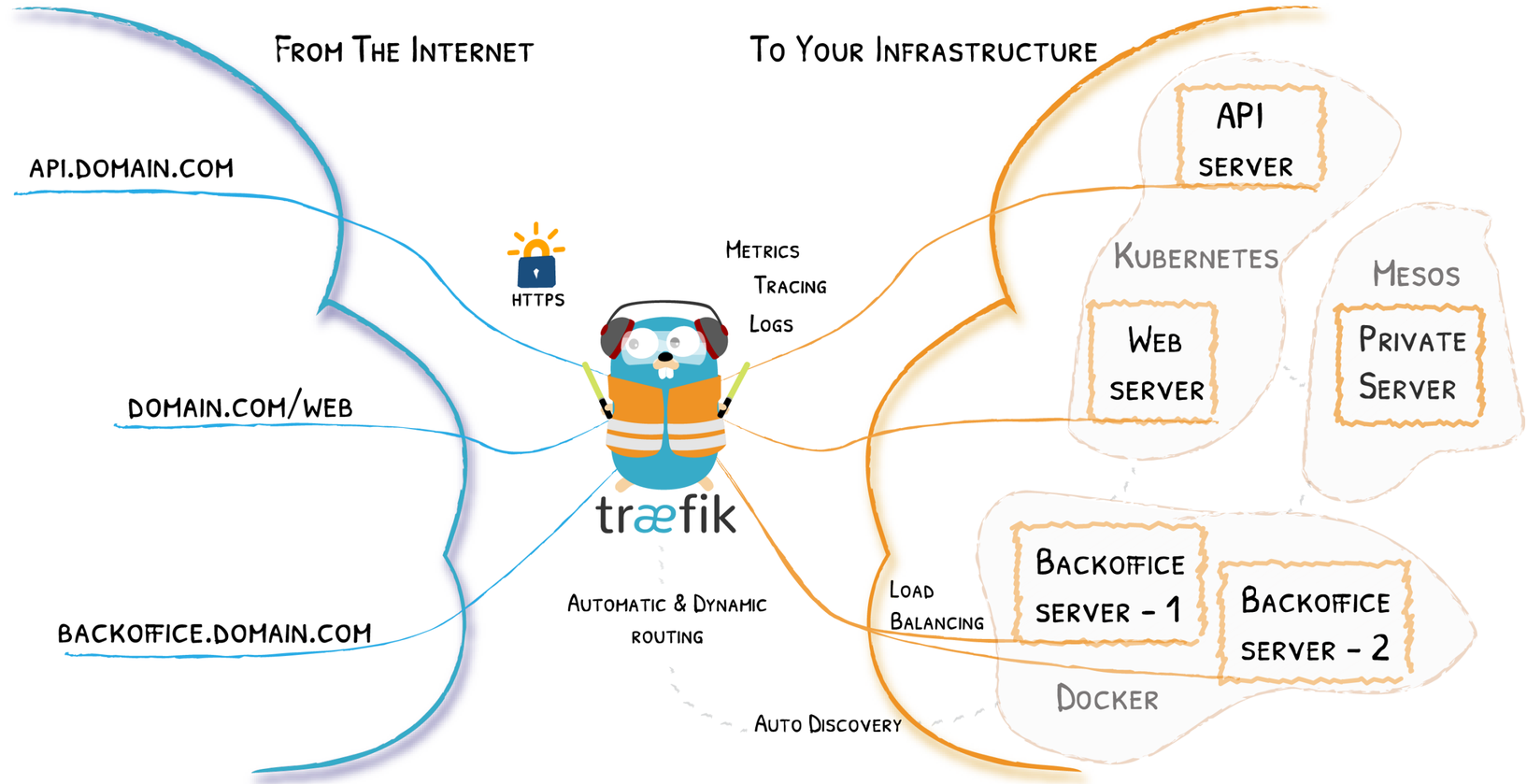


Source: https://twitter.com/Caged/status/1039937162769096704

What If I Told You?

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

Here Comes Traefik!



Traefik Project

- https://github.com/containous/traefik
- MIT License
- Written in Go
- 24,000+ ☆ 900M+ ⬇ 350+ 👷
- Created in 2015
- Current stable branch: `v1.7`

BACK TO TRAEFIK PART 2.0

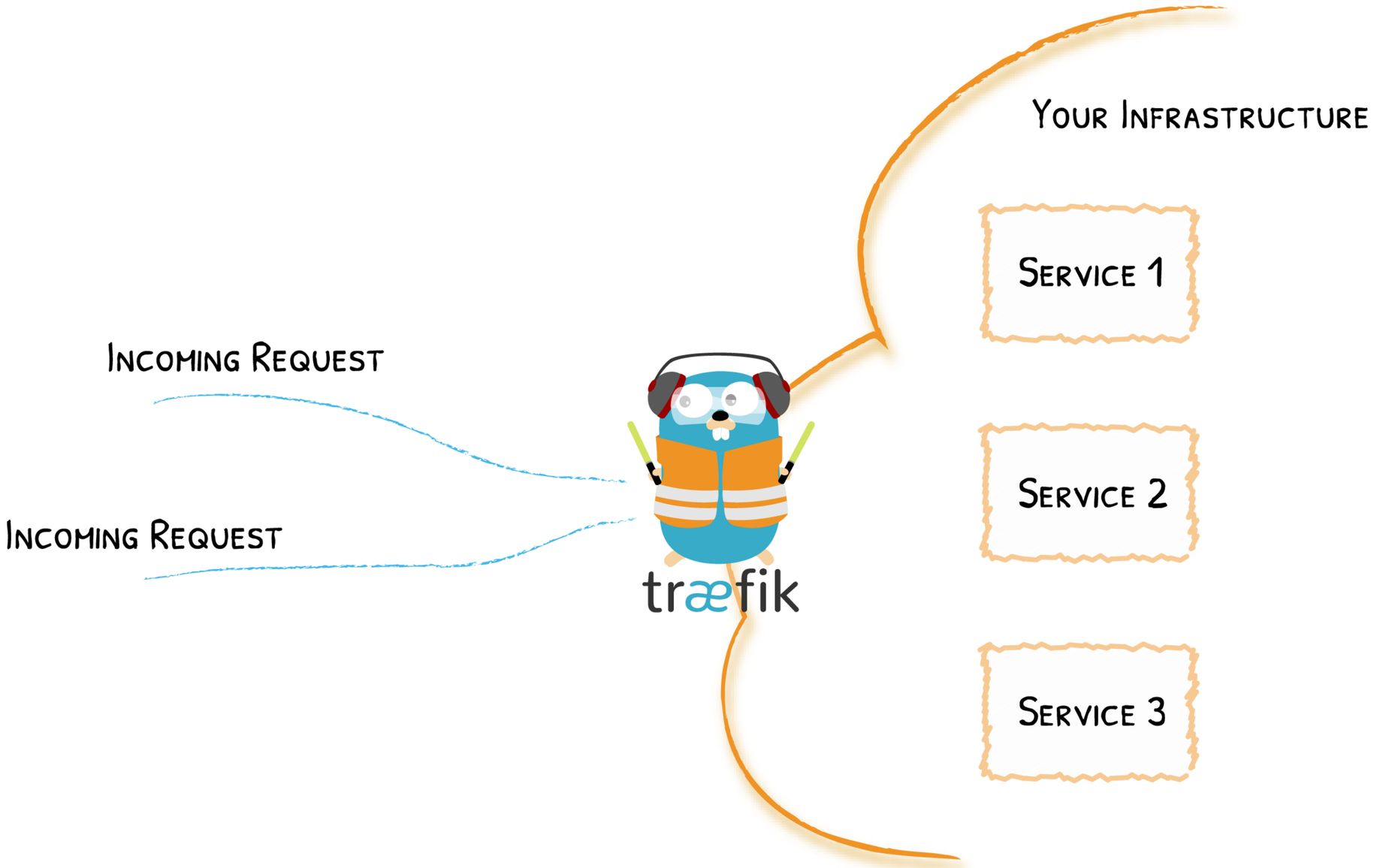
Traefik 2.0 Quick Overview

- Revamped Documentation
- Clarified Concepts
- Expressive Routing Rule Syntax
- Middlewares
- TCP Support
- TLS stores & options
- And so Much More...

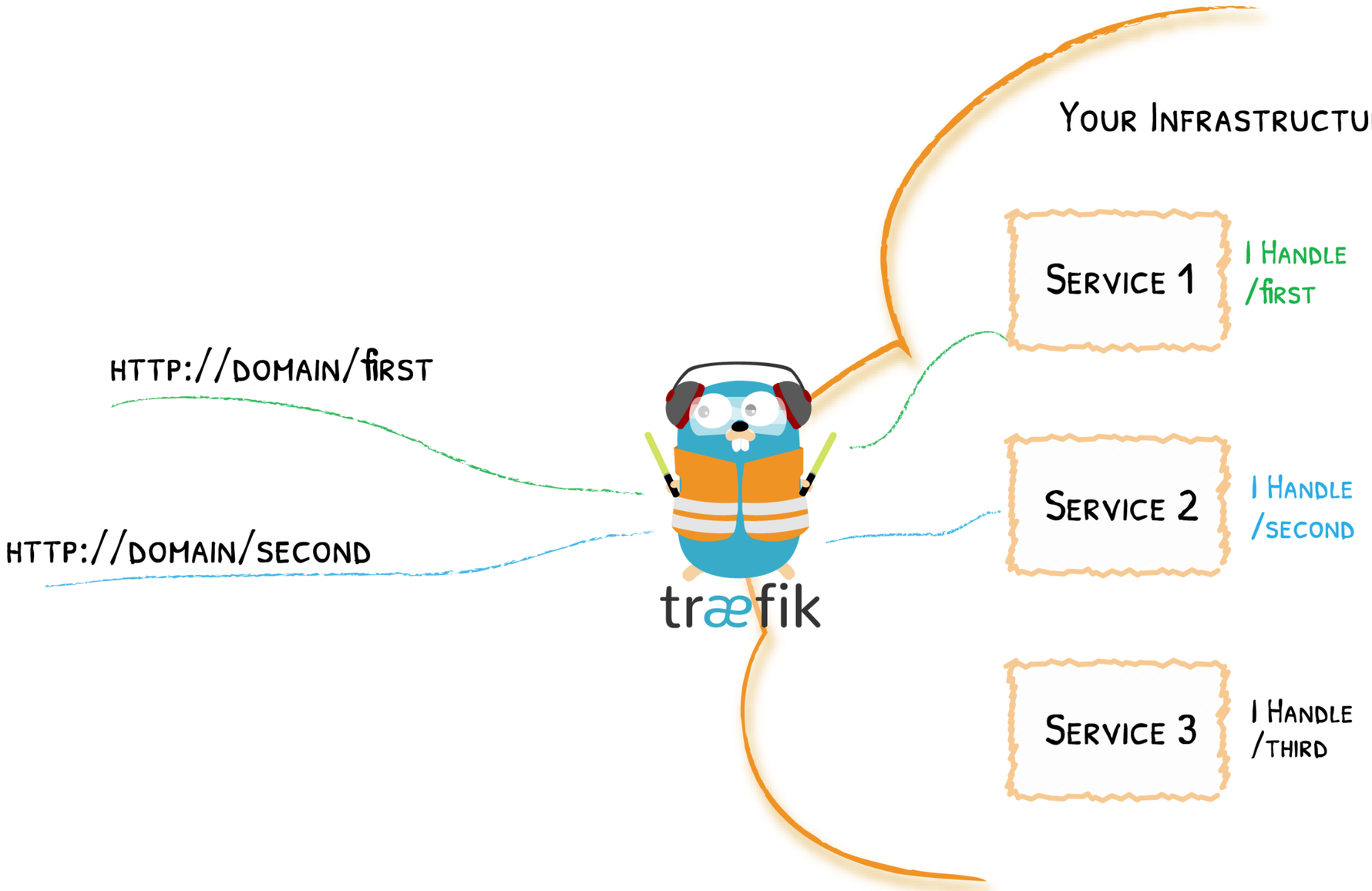
Learn more on the blog post

Traefik (V2.0) Core Concepts

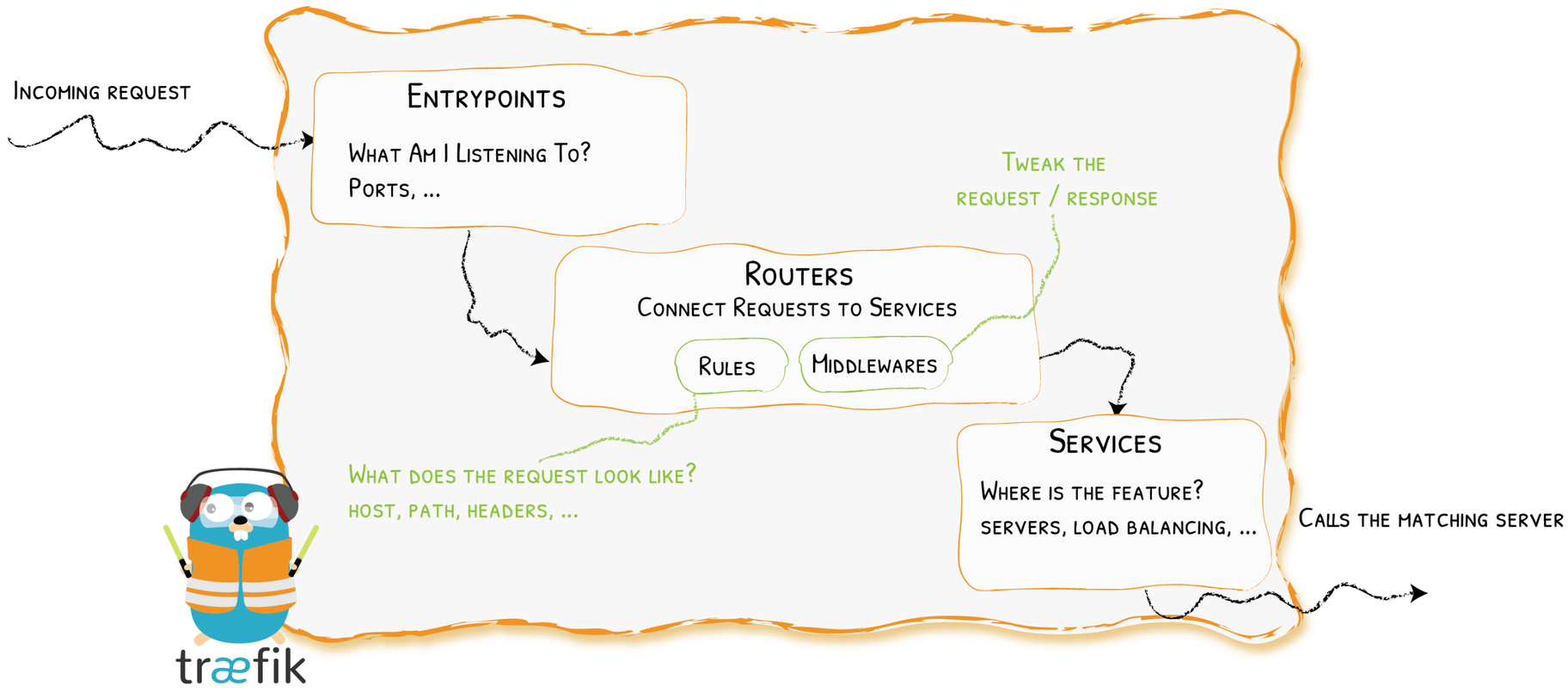
Traefik Is An Edge Router



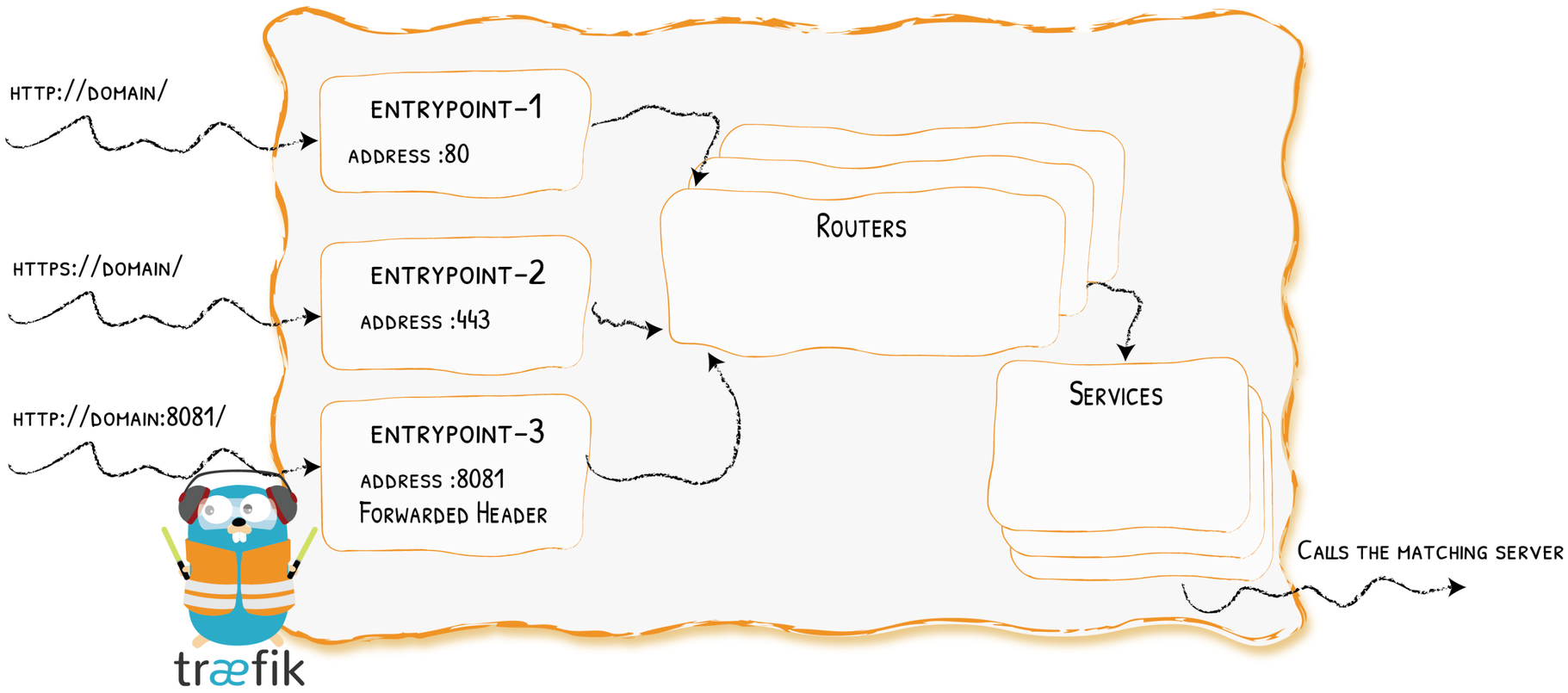
Traefik Dynamically Discovers Services



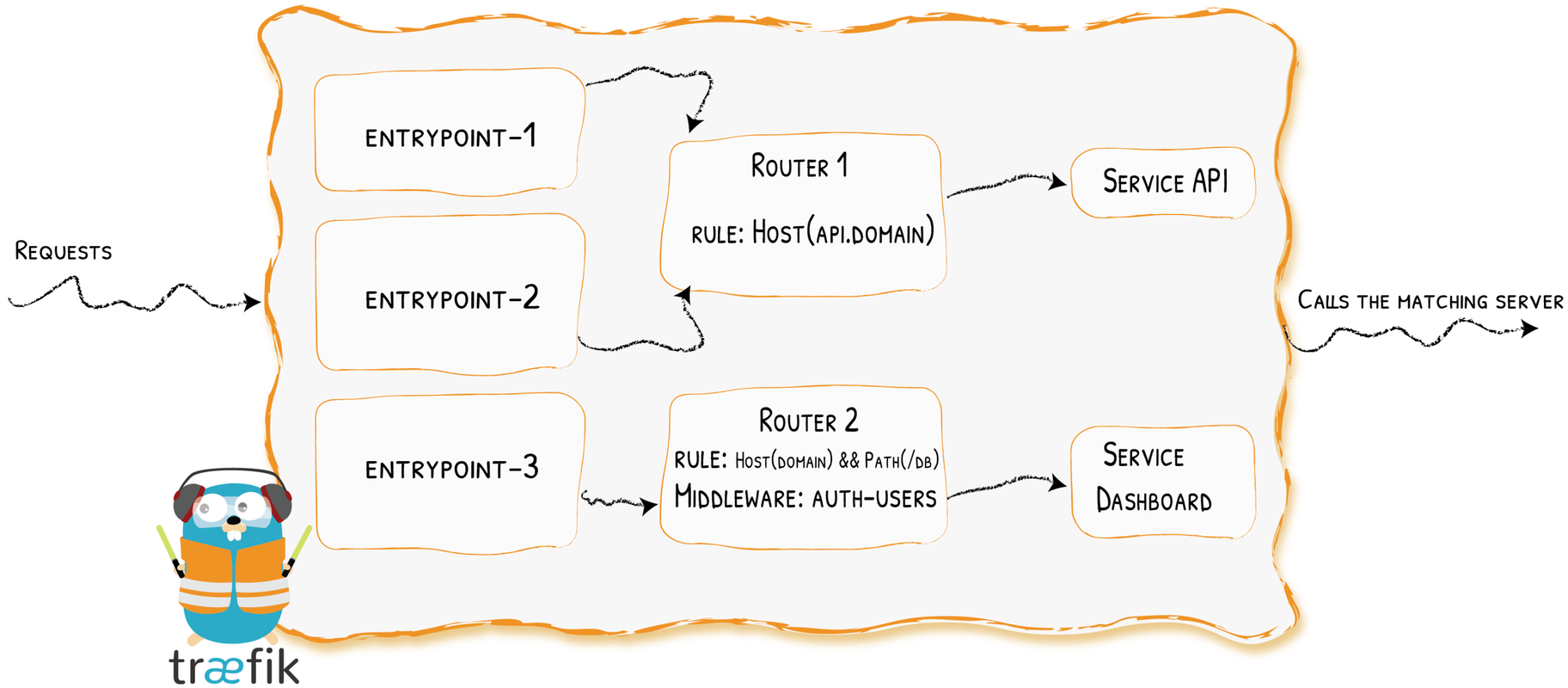
Architecture (V2.0) At A Glance



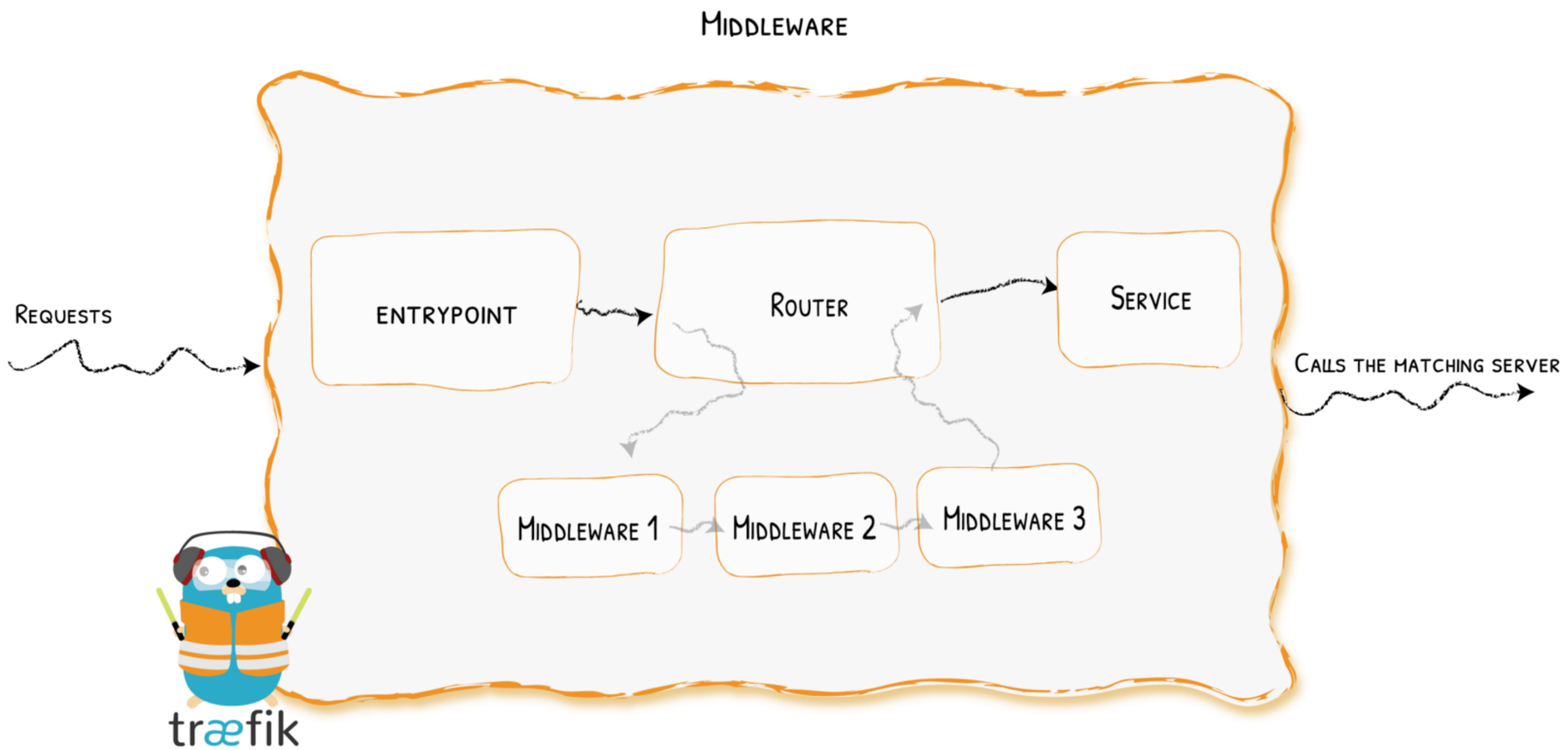
Entrypoints



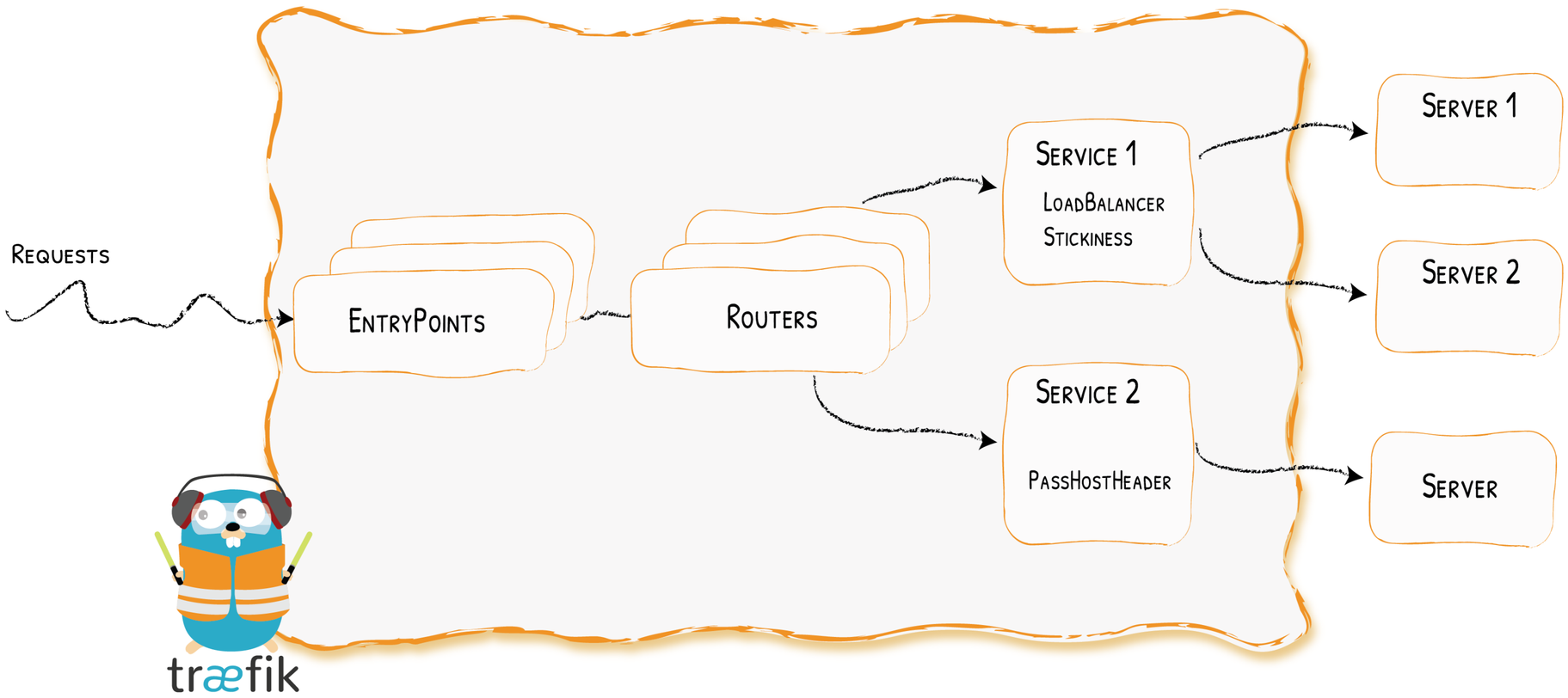
Routers


ENTRYPOINT-1
ENTRYPOINT-2
ENTRYPOINT-3
ROUTER 1
RULE: HOST(API.DOMAIN)
ROUTER 2
RULE: HOST(DOMAIN) && PATH(/DB)
MIDDLEWARE: AUTH-USERS
SERVICE API
SERVICE DASHBOARD
REQUESTS
CALLS THE MATCHING SERVER
træfik

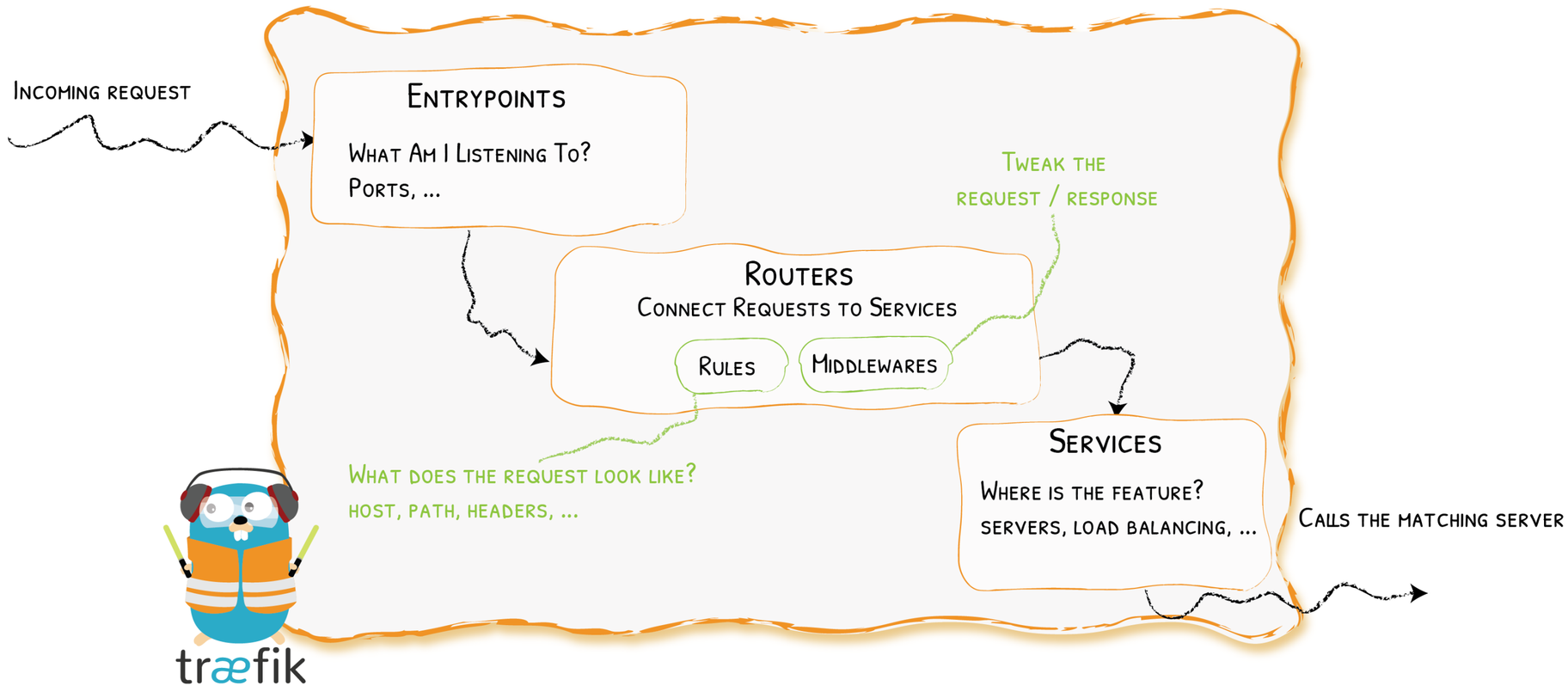
Middlewares



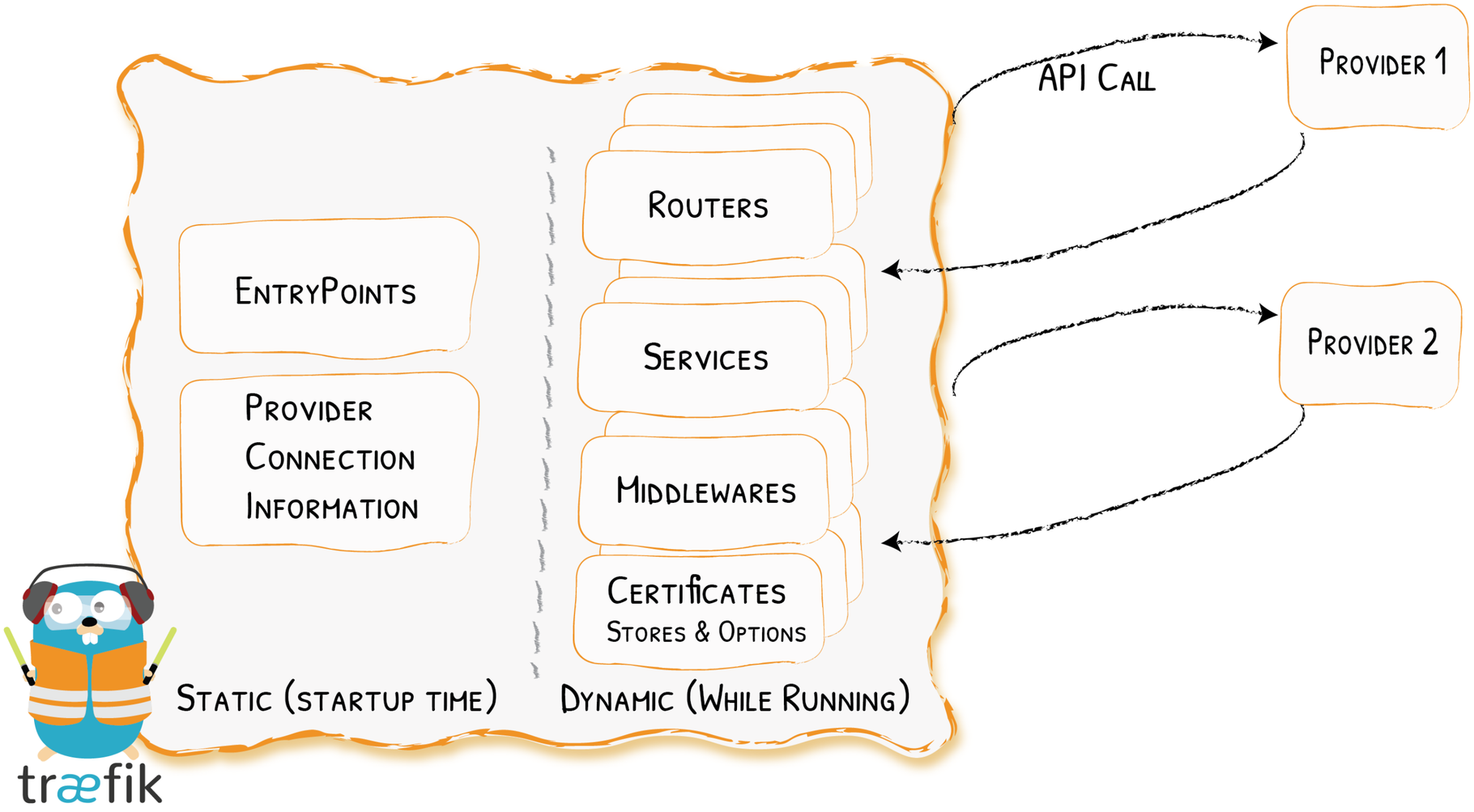
Services


Requests
EntryPoints
Routers
Service 1
LoadBalancer
Stickiness
Service 2
PassHostHeader
Server 1
Server 2
Server
træfik

Architecture (Again) At A Glance

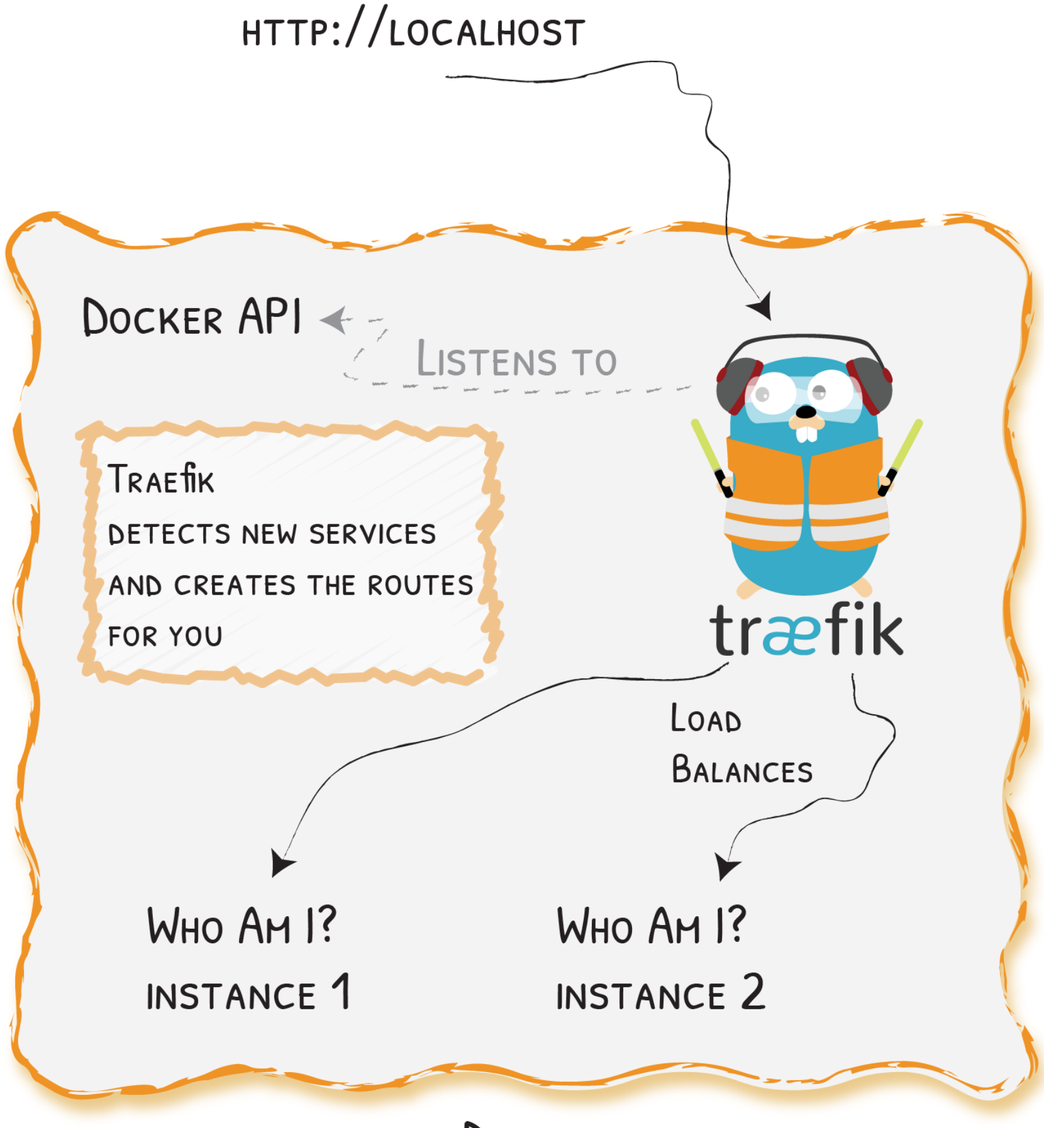


Static & Dynamic Configuration



Show Me The Configuration!

Simple Example With



TRAEFIK IS THE ONLY ACCESSIBLE CONTAINER FROM THE OUTSIDE.

OTHER SERVICES ARE DEPLOYED ON DEMAND AND AVAILABLE THROUGHT TRAEFIK.

With 

- With Docker Compose:

```yaml
version: '3'

services:
  reverse-proxy:
    image: traefik:v2.0
    command: --providers.docker
    ports:
      - "80:80"
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock

  webapp:
    image: containous/whoami
    labels:
      - "traefik.http.routers.webapp.rule=Host(`localhost`)"
```

With : Context

```yaml
# https://mycompany.org/jenkins -> http://jenkins:8080/jenkins
jenkins:
    image: jenkins/jenkins:lts
    environment:
      - JENKINS_OPTS=--prefix=/jenkins
    labels:
      - "traefik.http.services.jenkins.LoadBalancer.server.Port=8080" # Because 50000 is also exposed
      - "traefik.http.routers.jenkins.rule=Host(`mycompany.org`) && PathPrefix(`/jenkins`)"
      - "traefik.http.routers.jenkins.service=jenkins"
```

With : Rewrites

```
# https://mycompany.org/gitserver -> http://gitserver:3000/
gitserver:
    image: gitea/gitea
    labels:
      - "traefik.http.routers.gitserver.rule=Host(`mycompany.org`) && PathPrefix(`/gitserver`)"
      - "traefik.http.middlewares.gitserver-stripprefix.stripprefix.prefixes=/gitserver"
      - "traefik.http.routers.gitserver.middlewares=gitserver-stripprefix"
```

Traefik With ⎈

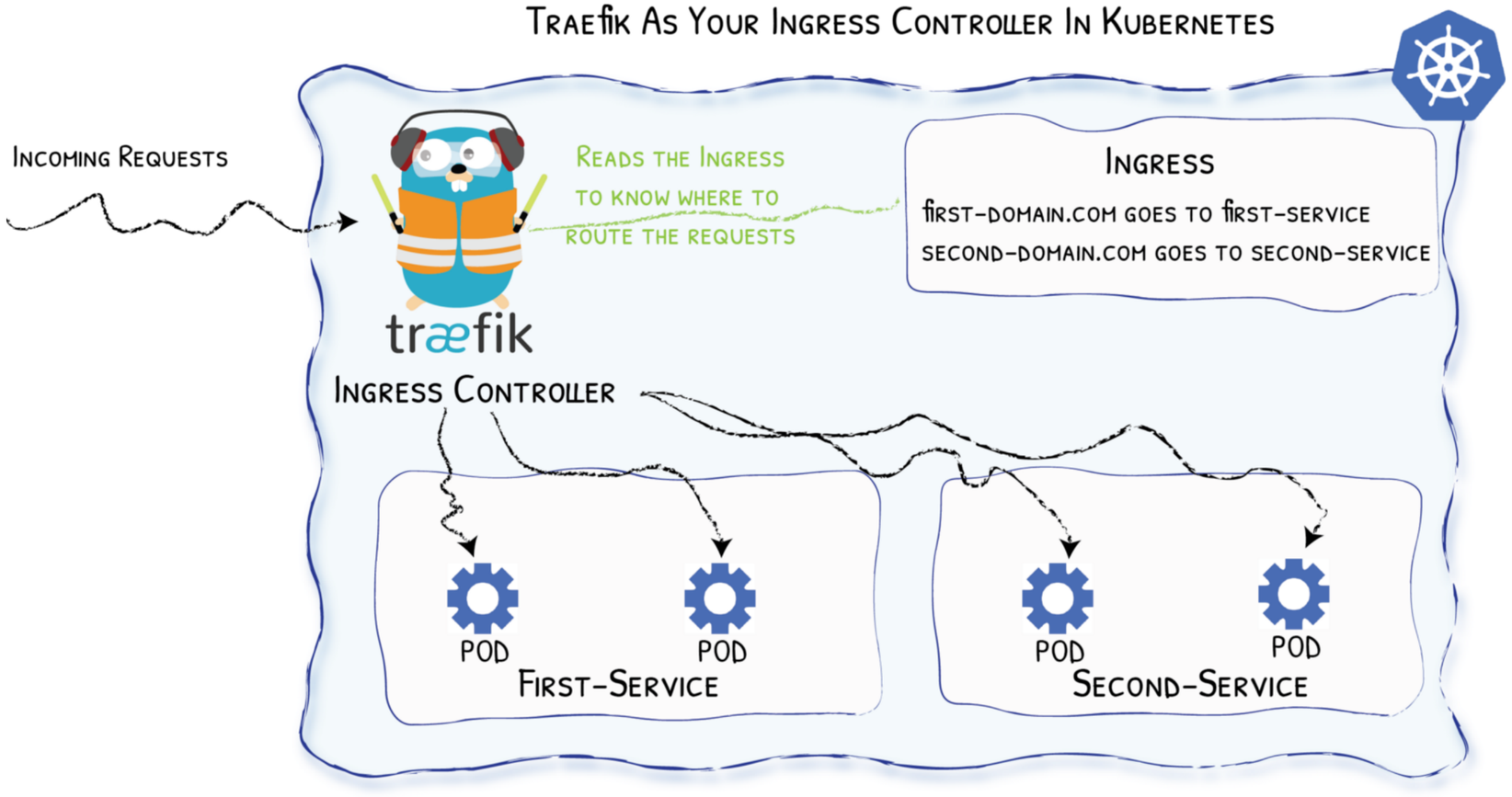


Diagram from https://medium.com/@geraldcroes

Example Code With ⎈

```yaml
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  annotations:
    kubernetes.io/ingress.class: 'traefik'
spec:
  rules:
  - host: localhost
    http:
      paths:
      - path: "/whoami"
        backend:
          serviceName: webapp
          servicePort: 80
```

⎈ CRD - Custom Resources Definition

```yaml
# 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)

```yaml
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
```

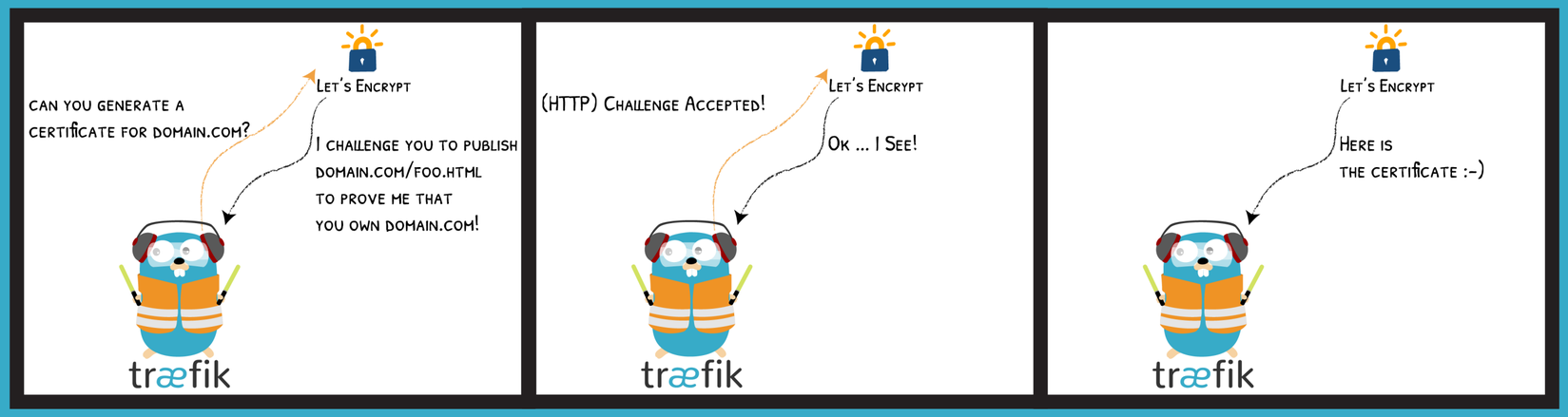
traefik

HTTP

&

TCP

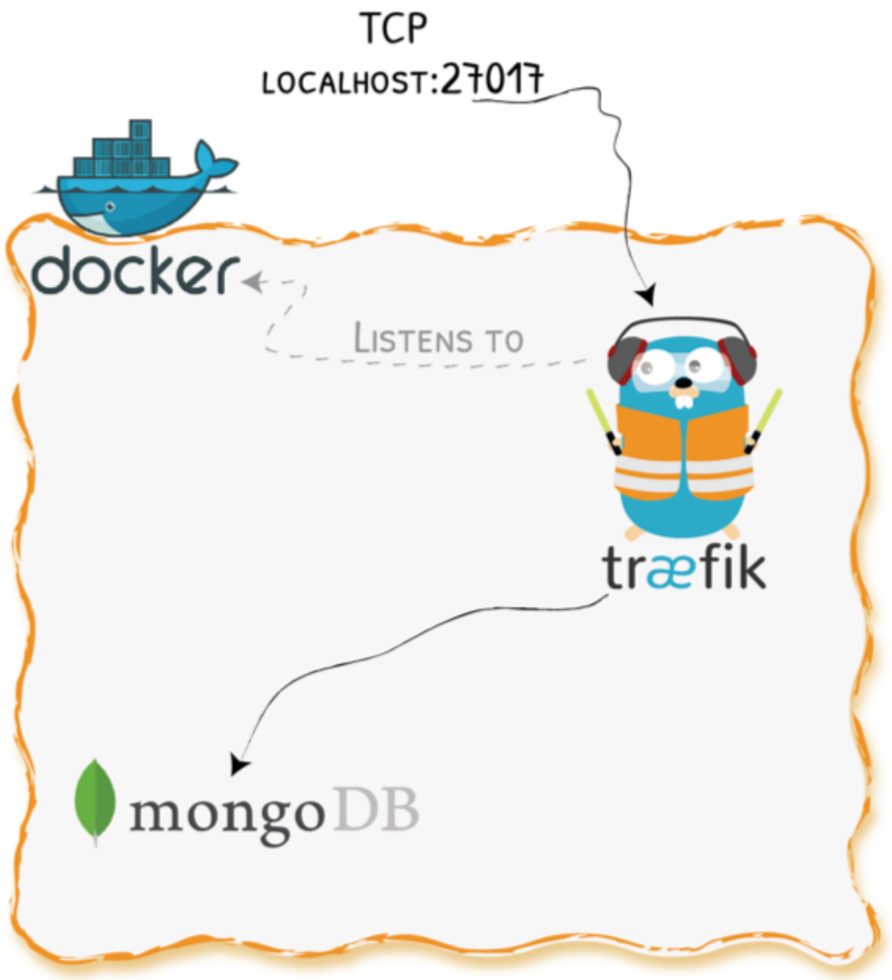
HTTPS For Everyone With Let's Encrypt



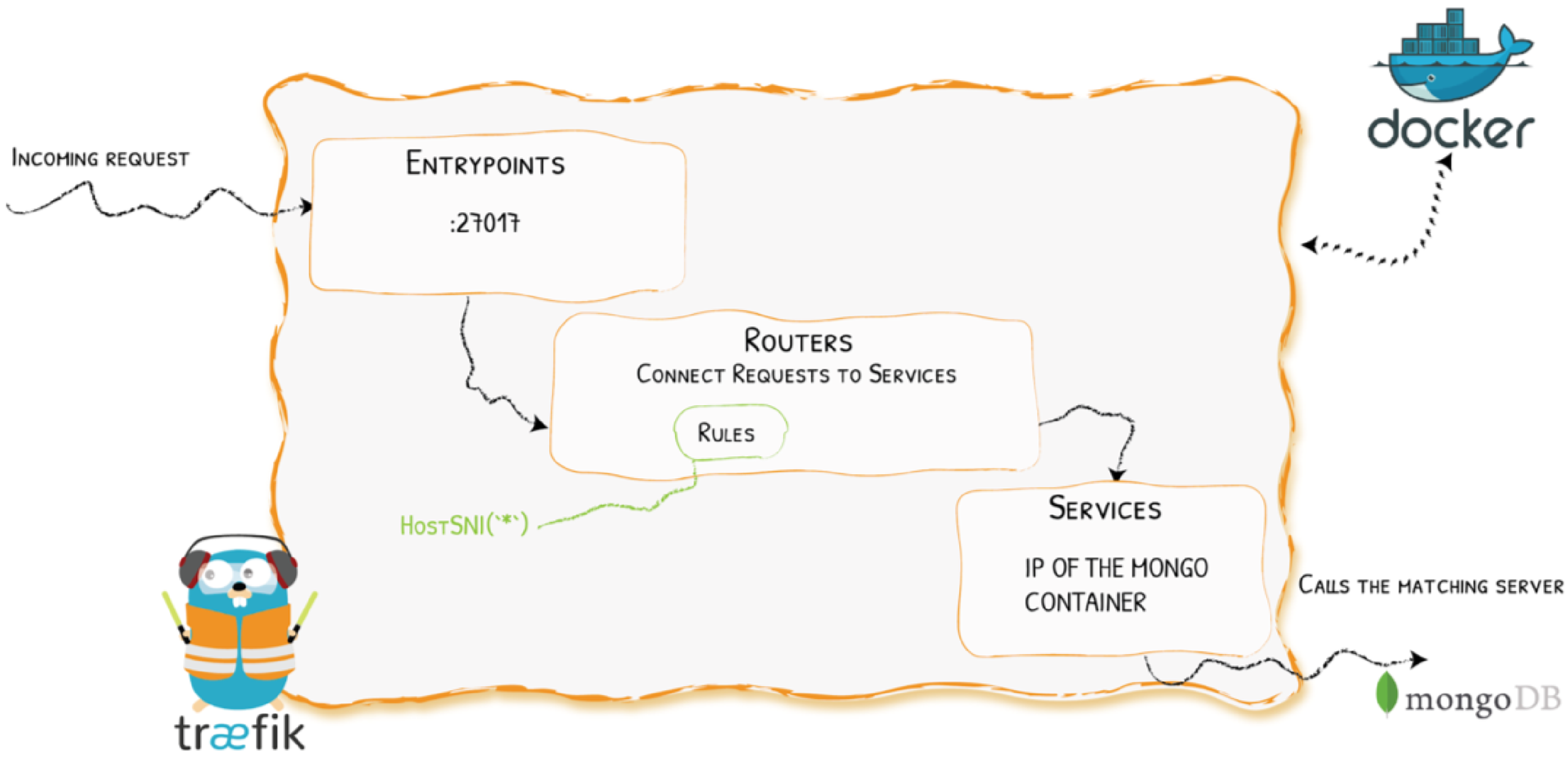
- TLS, DNS and HTTP challenges supported

Demo

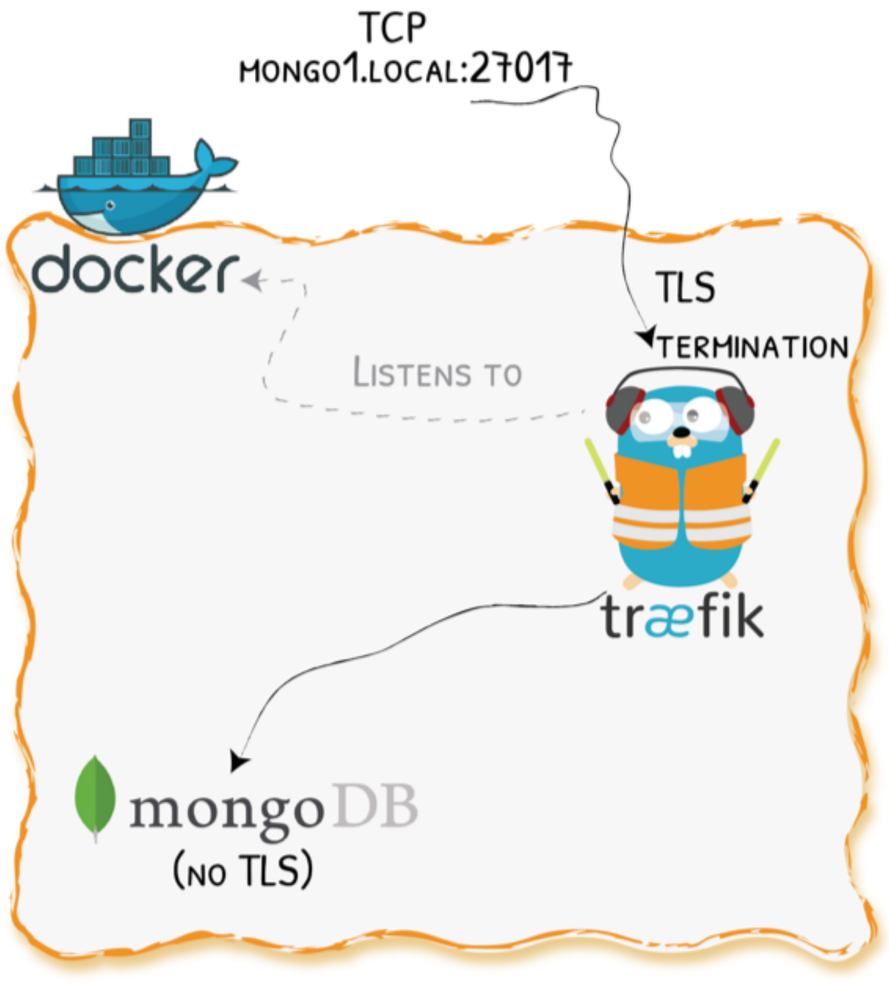
Demo 1 - Straightforward TCP Routing



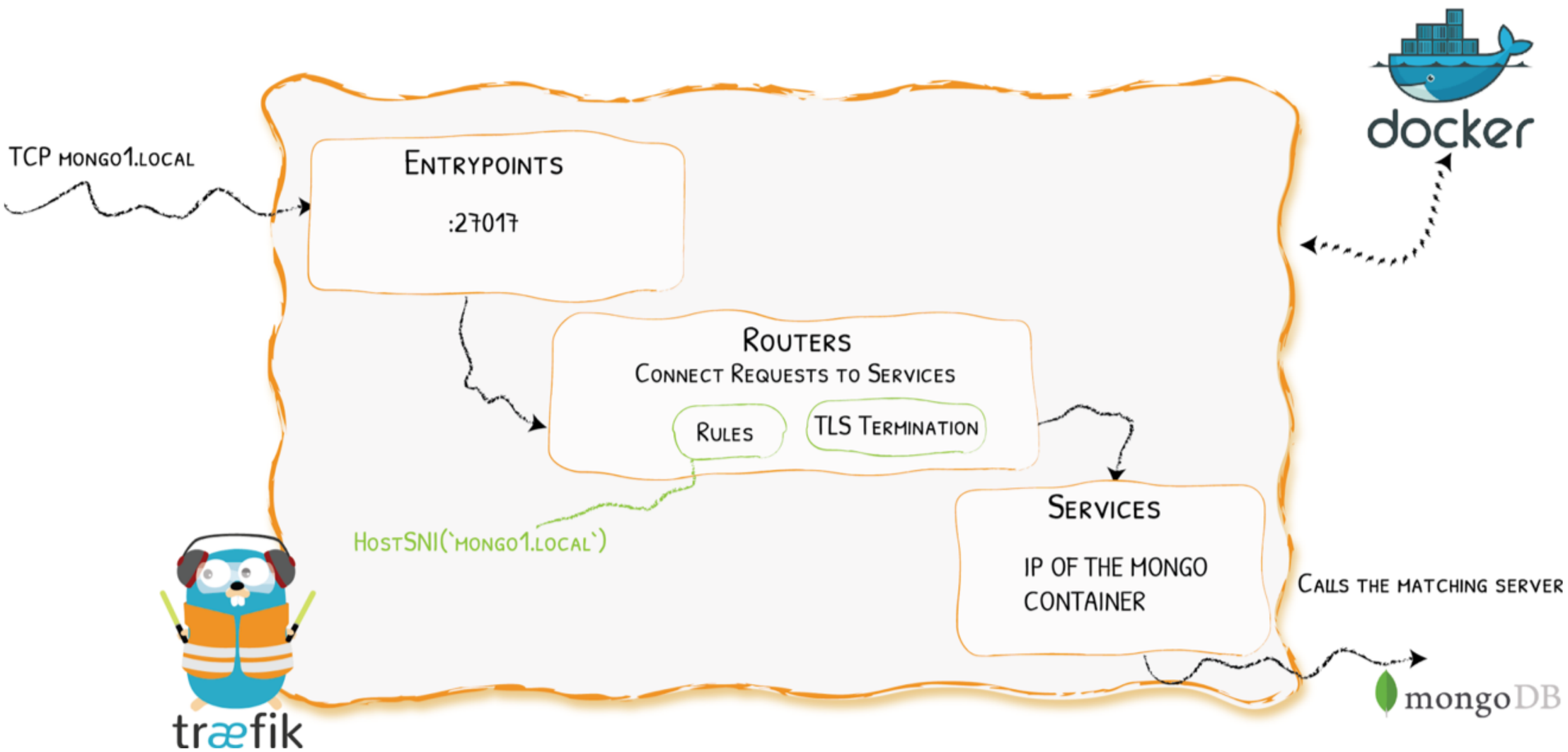
Demo 1 - Configuration



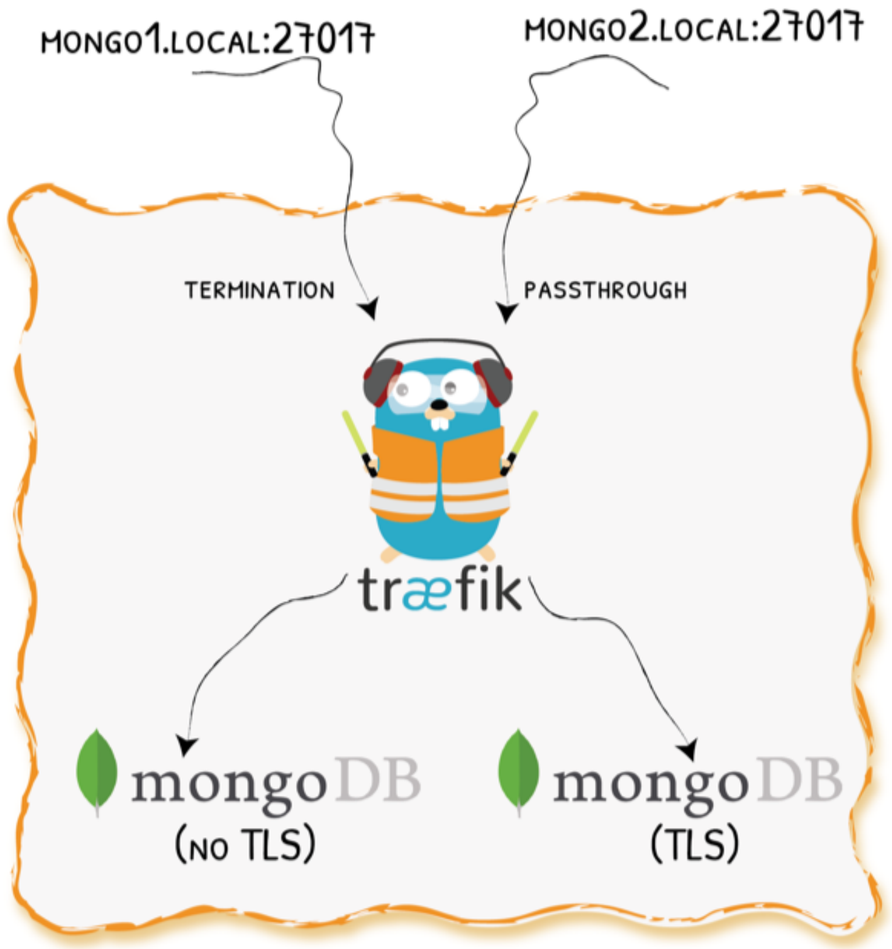
Demo 2 - Let's Add TLS To TCP With Traefik



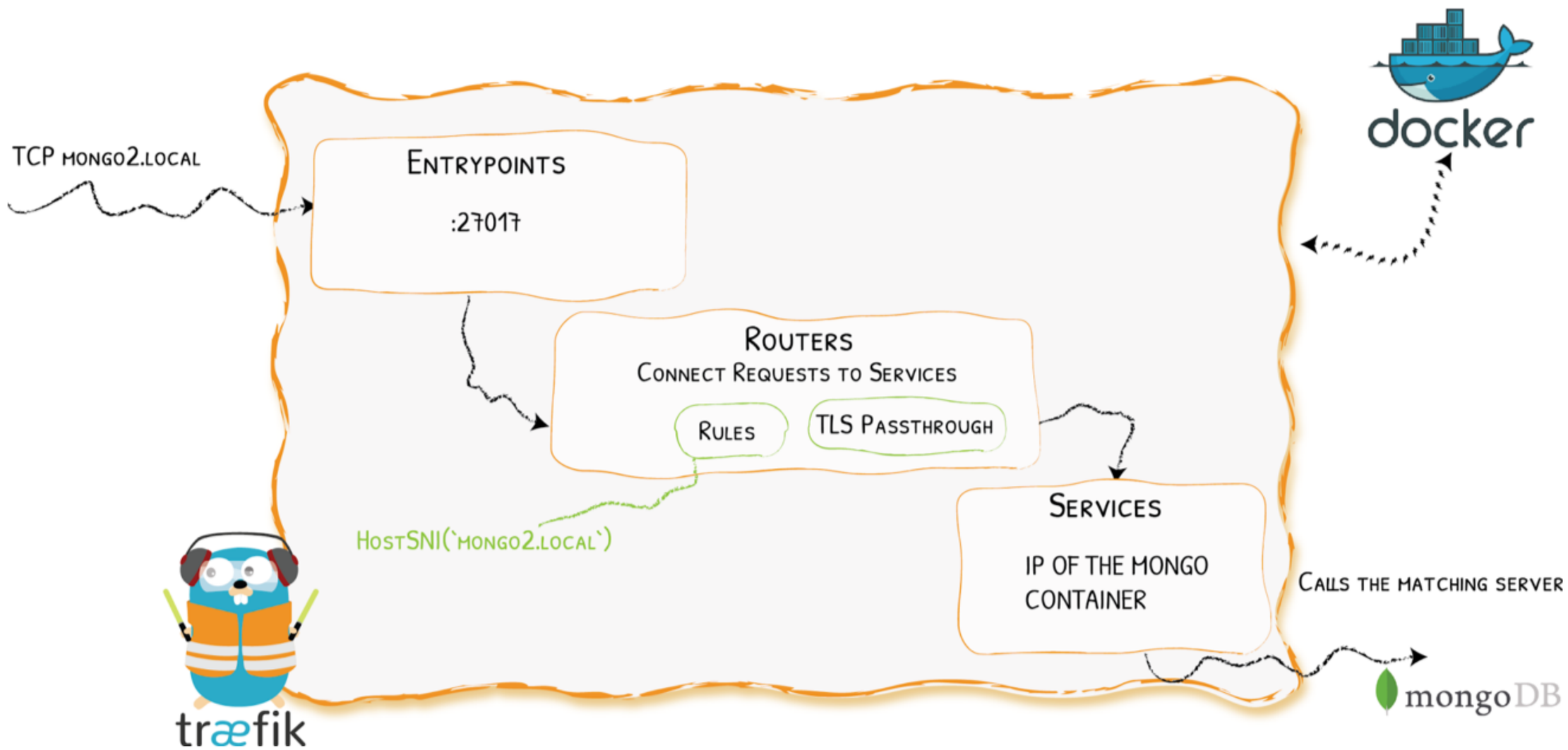
Demo 2 - Configuration



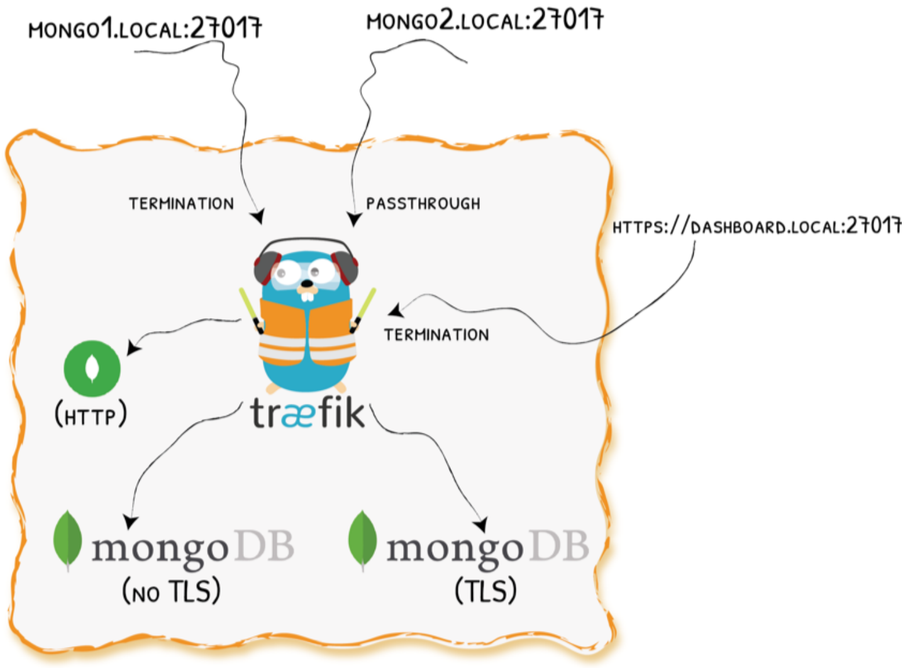
Demo 3 - SNI Routing + TLS Passthrough



Demo 3 - Configuration



Demo 4 - Muxing HTTPS And TCP On The Same Port



Demo 5 - Deploying A Simple WebApp With Lets Encrypt

More To Come For V2.0

- New WebUI
- New metrics
- UDP
- YAML
- Canary

More Info

bit.ly/traefik-v2

We Also Missed Talking About...

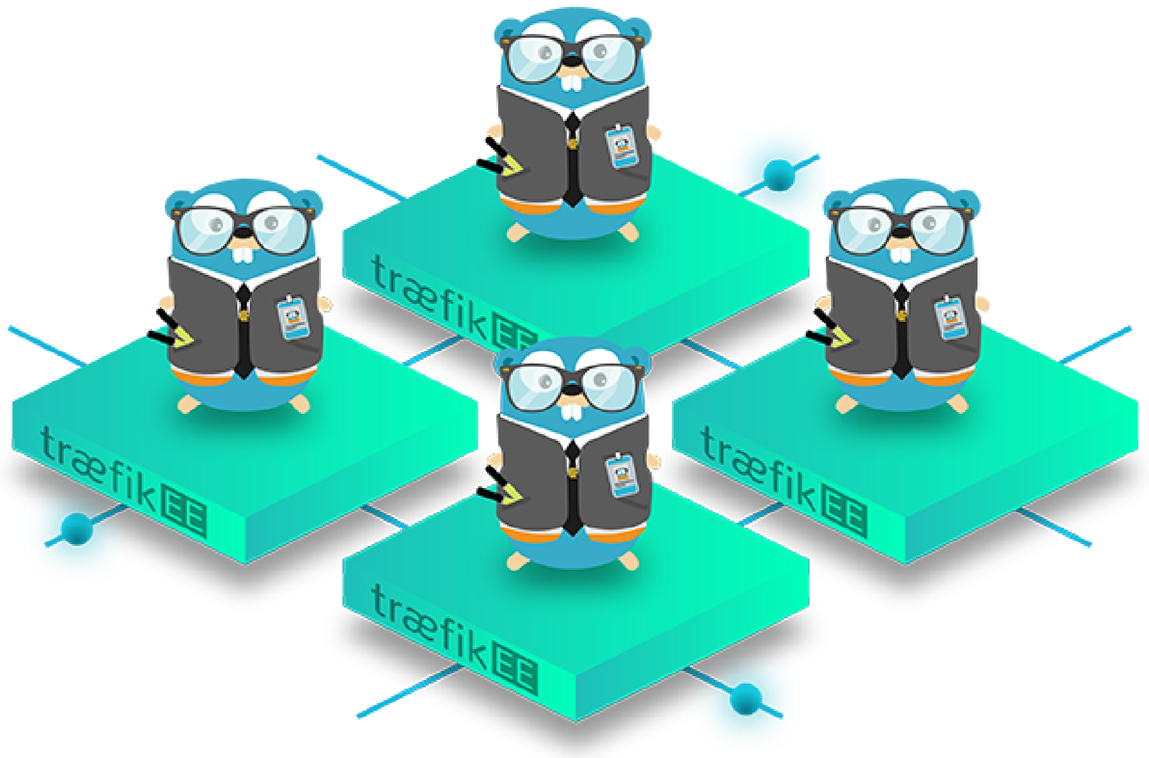
TO BE

CONTINUED...

The Herd

You came to the wrong neighbour

Traefik Comes In Herd



HIGH AVAILABILITY

træfik ENTERPRISE EDITION

SECURITY

træfik ENTERPRISE EDITION

SCALABILITY

træfik ENTERPRISE EDITION

NTERNET

TO YOUR INFRA

Control Plane

traefikEE

?

ORCHESTR

Data Plane

traefikEE

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 \
    --acme.email=ssl-admin@mycompany.org
    --acme.tlsChallenge
    ...
```

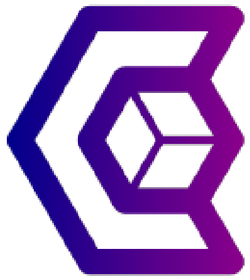
Free Trial

https://containo.us/traefikee

Thanks!


We Have
Stickers!
træfik

We Are Hiring!



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

Thank You!

@mZapfDE

SantoDE

- Slides (HTML): https://containous.github.io/slides/cloud-native-karlsruhe
- Slides (PDF): https://containous.github.io/slides/cloud-native-karlsruhe/slides.pdf