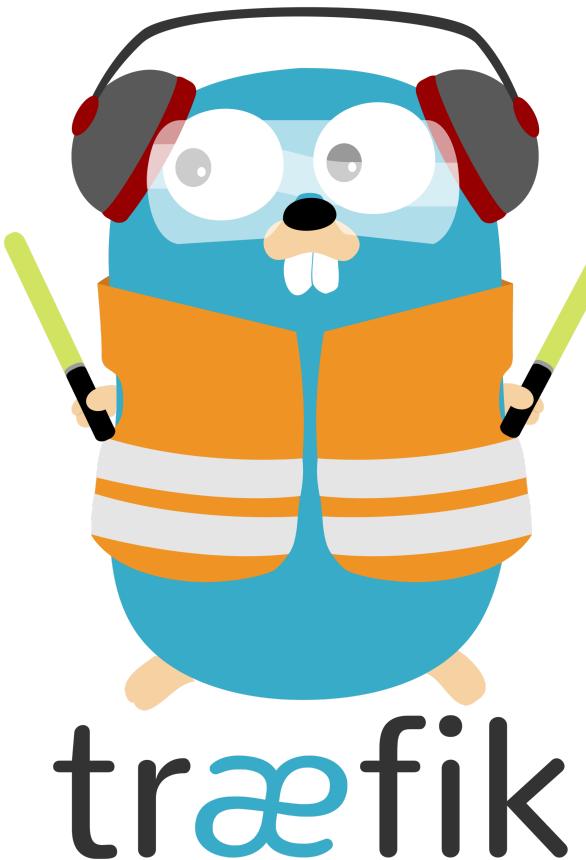


# Edge Routing And HTTPS For Everyone

## Traefik In Action!



<https://containous.github.io/slides/riga-devdays-2019>

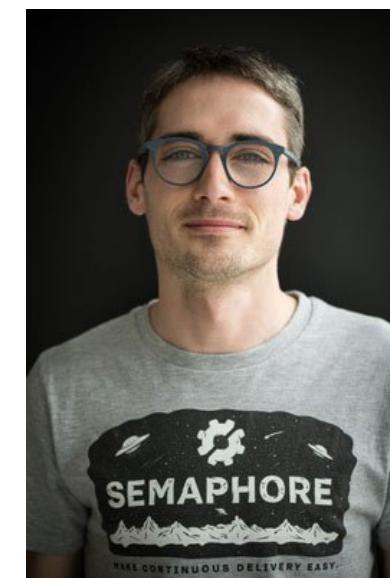
# How To Use These Slides?

- **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

Michael Matur

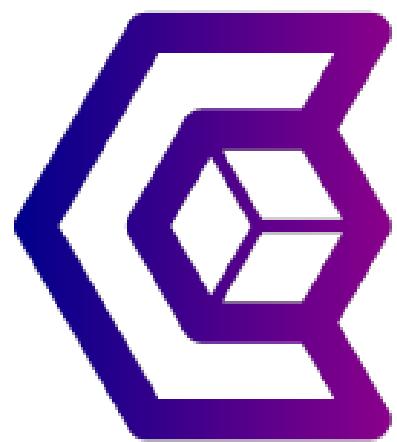
- Devops & Developer @ [Containous](#)
- Blacksmith on [Traefik](#)
-  [@michaelmatur](#)
-  [mmatur](#)



# Containous

<https://containo.us>

- We Believe in Open Source
- We Deliver Traefik
- Commercial Support for Traefik
- 22 people, 90% technical experts

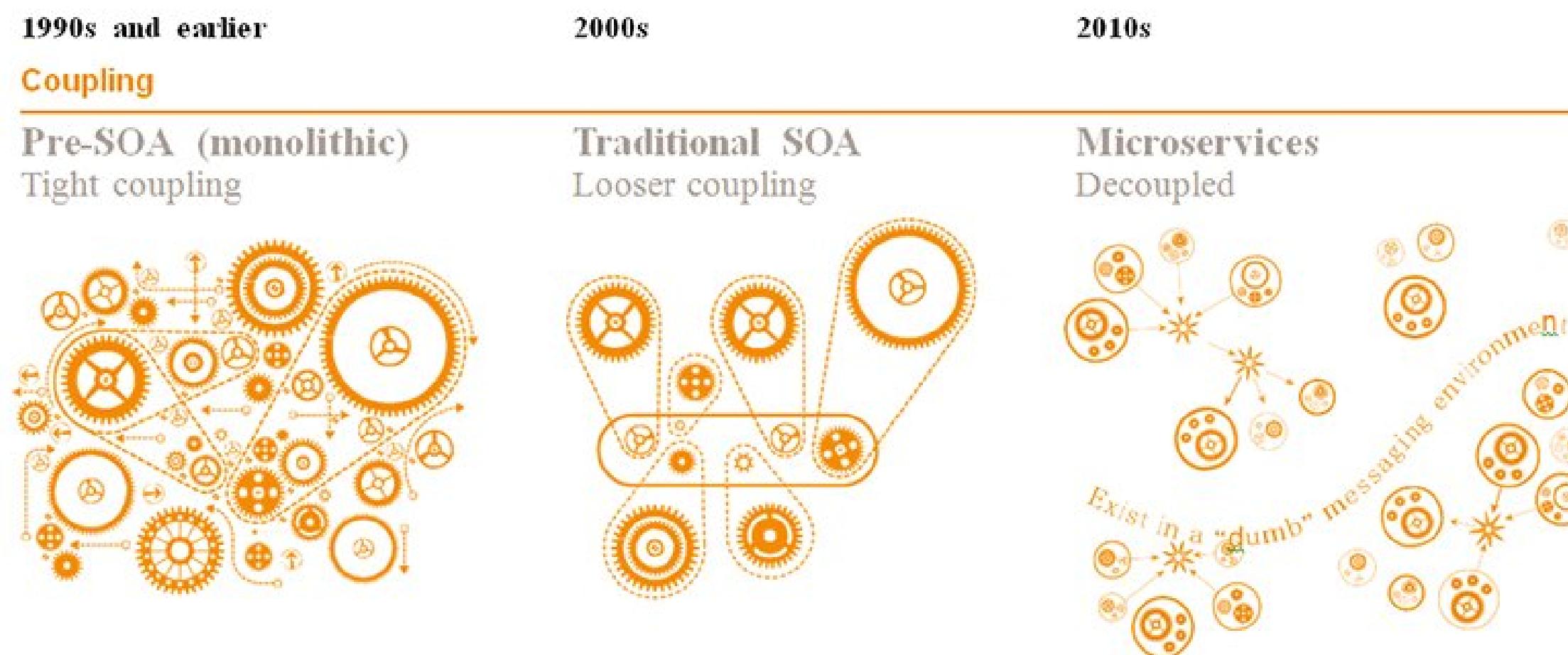


# Why Traefik?



Why, Mr Anderson?

# Evolution Of Software Design



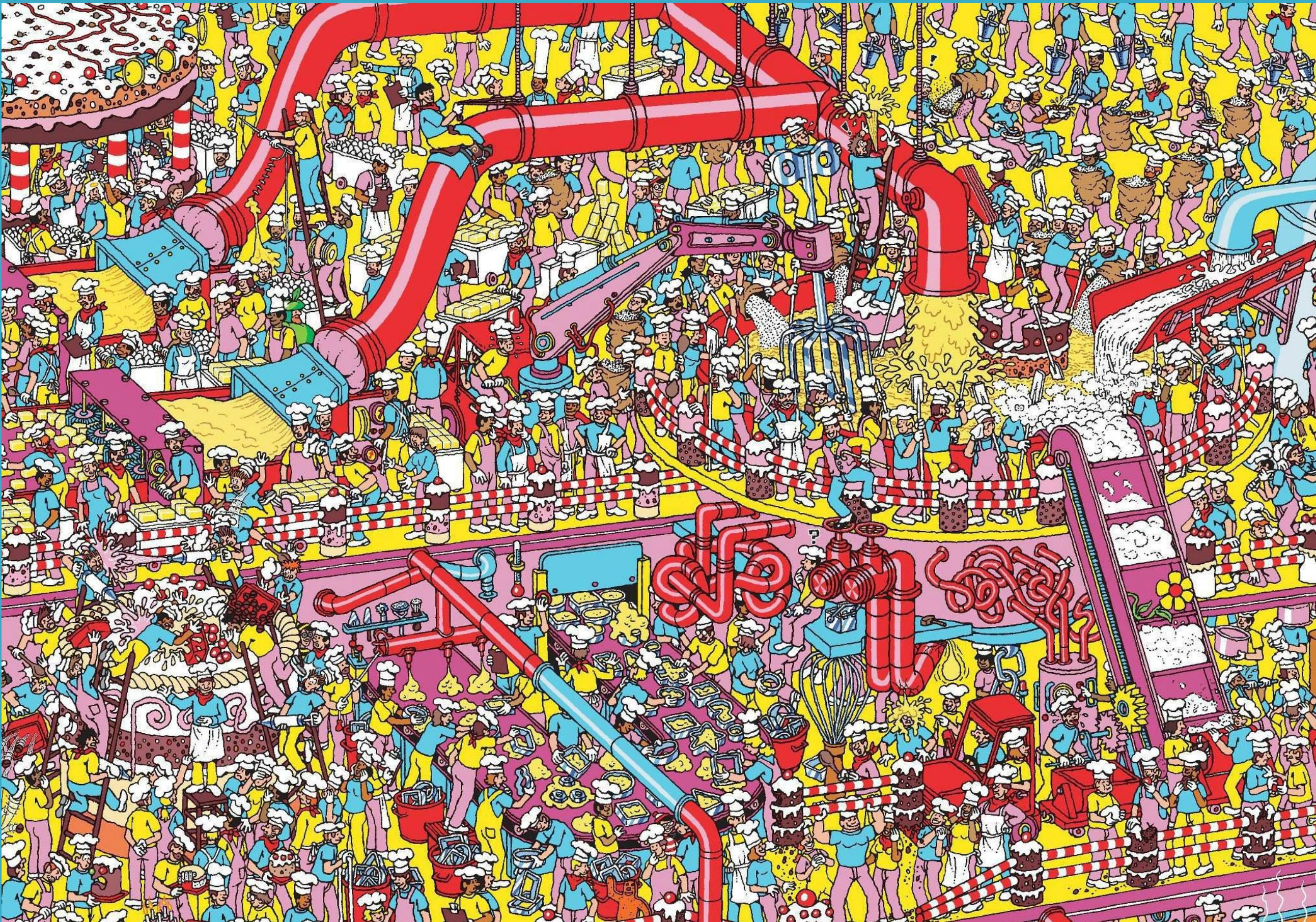
# The Premise Of Microservices...



*...And What Happens*

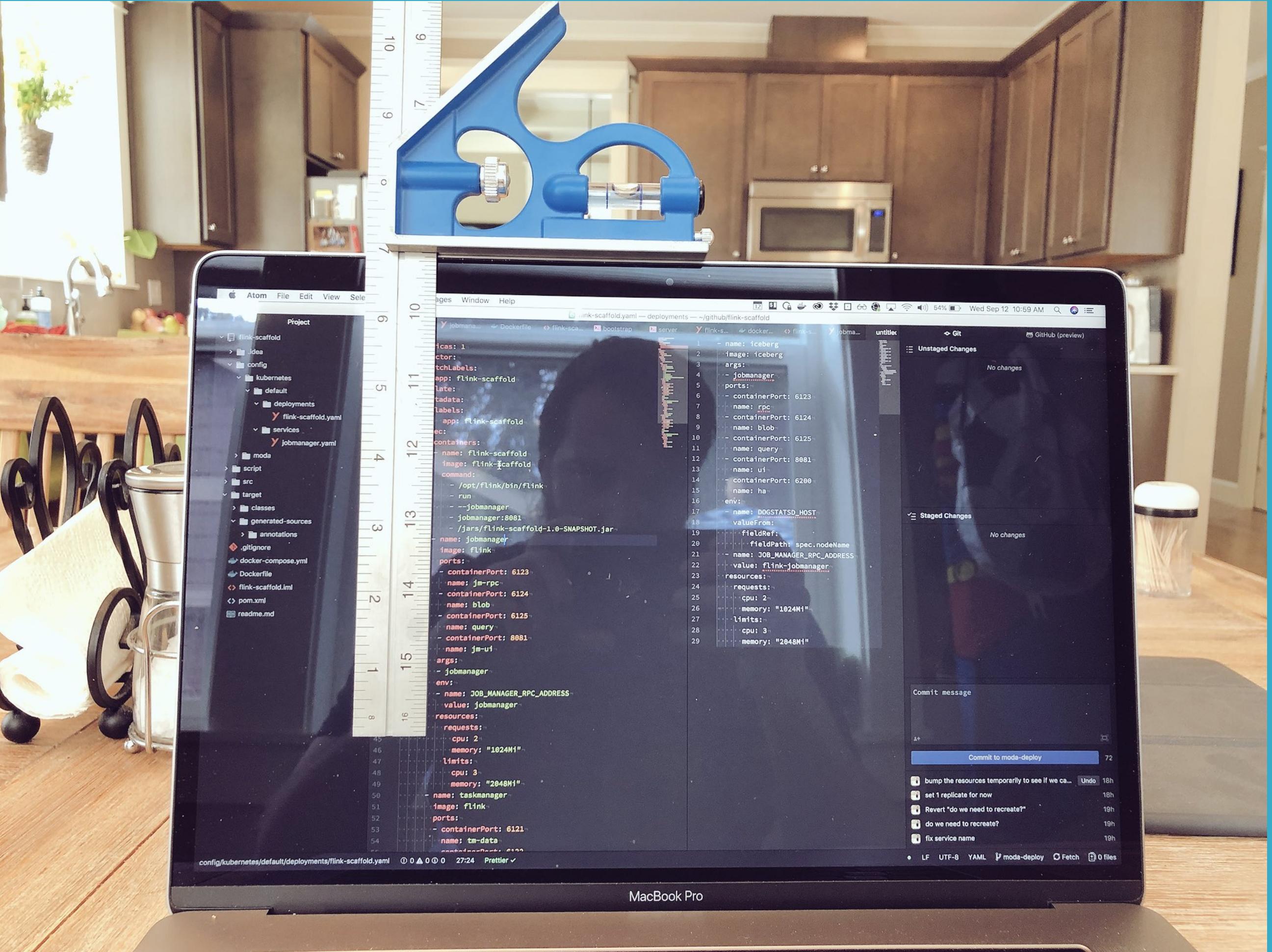


# Where Is The Service?



# Tools Of The Trade





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

# What If I Told You?



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

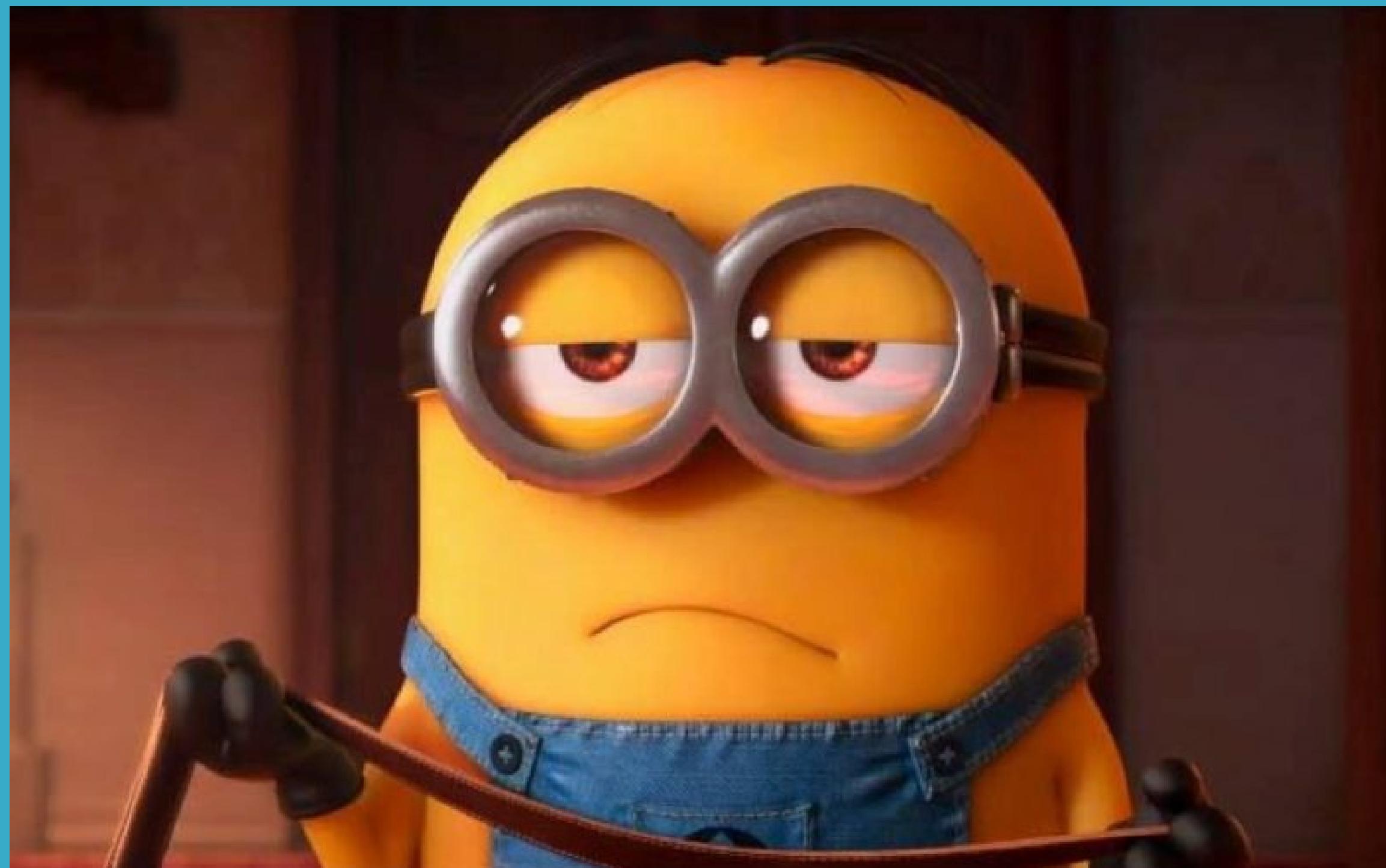
# Here Comes Traefik!



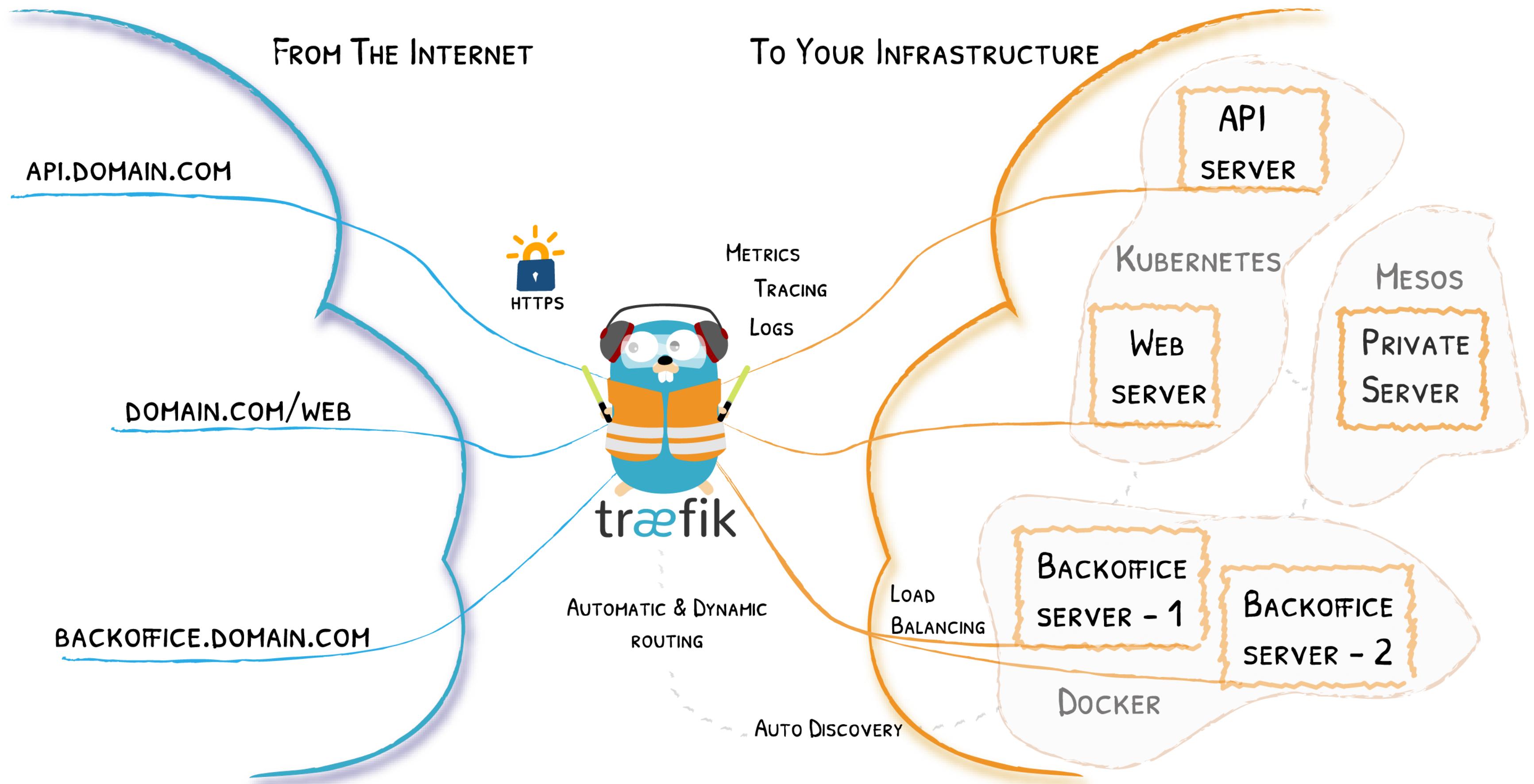
# Traefik Project

-  <https://github.com/containous/traefik>
- MIT License
- Written in Go, a popular language
- 22,000+ 
- 700M+ 
- 350+ 

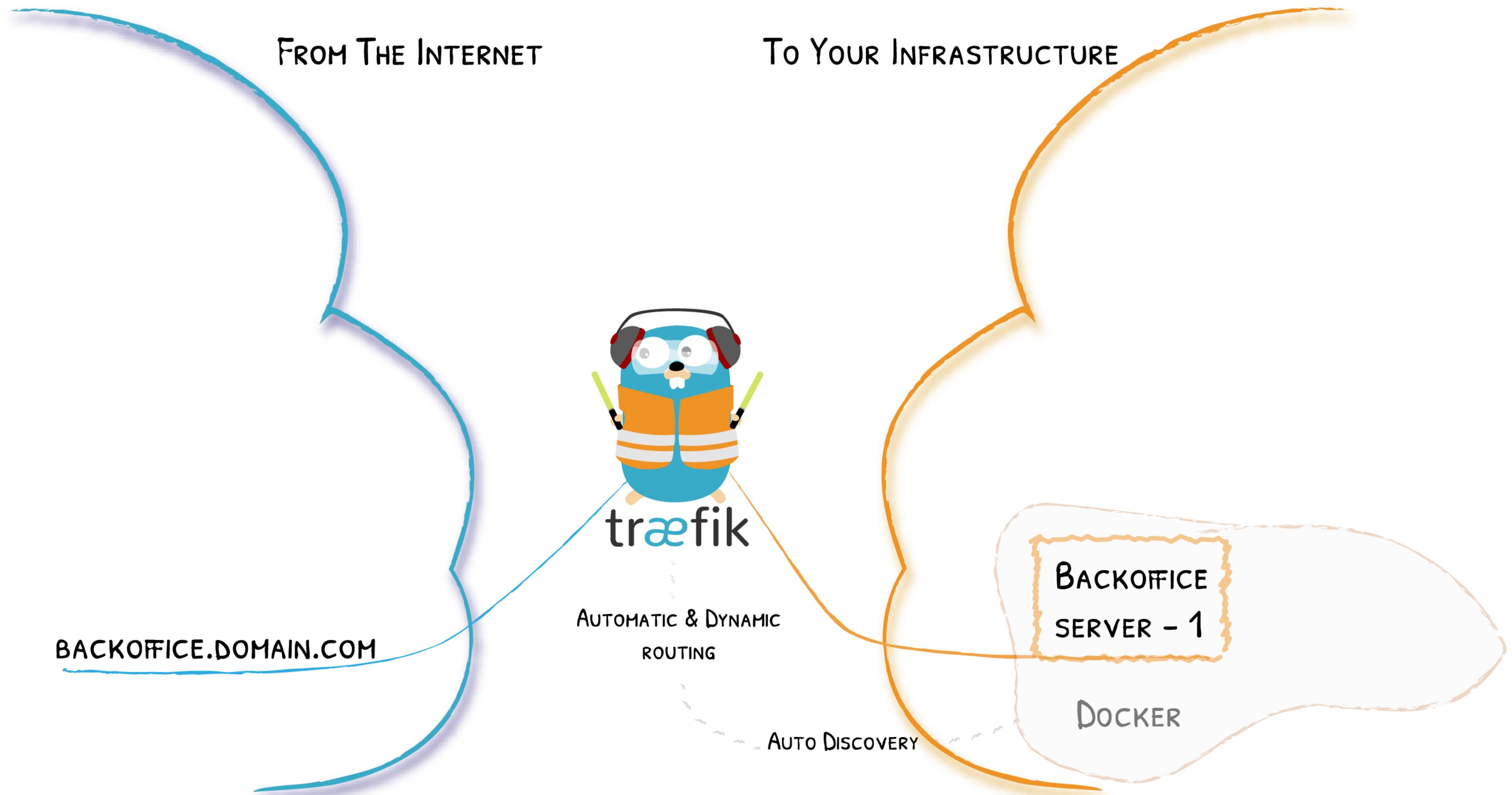
# Traefik Core Concepts



# Remember The Diagram?



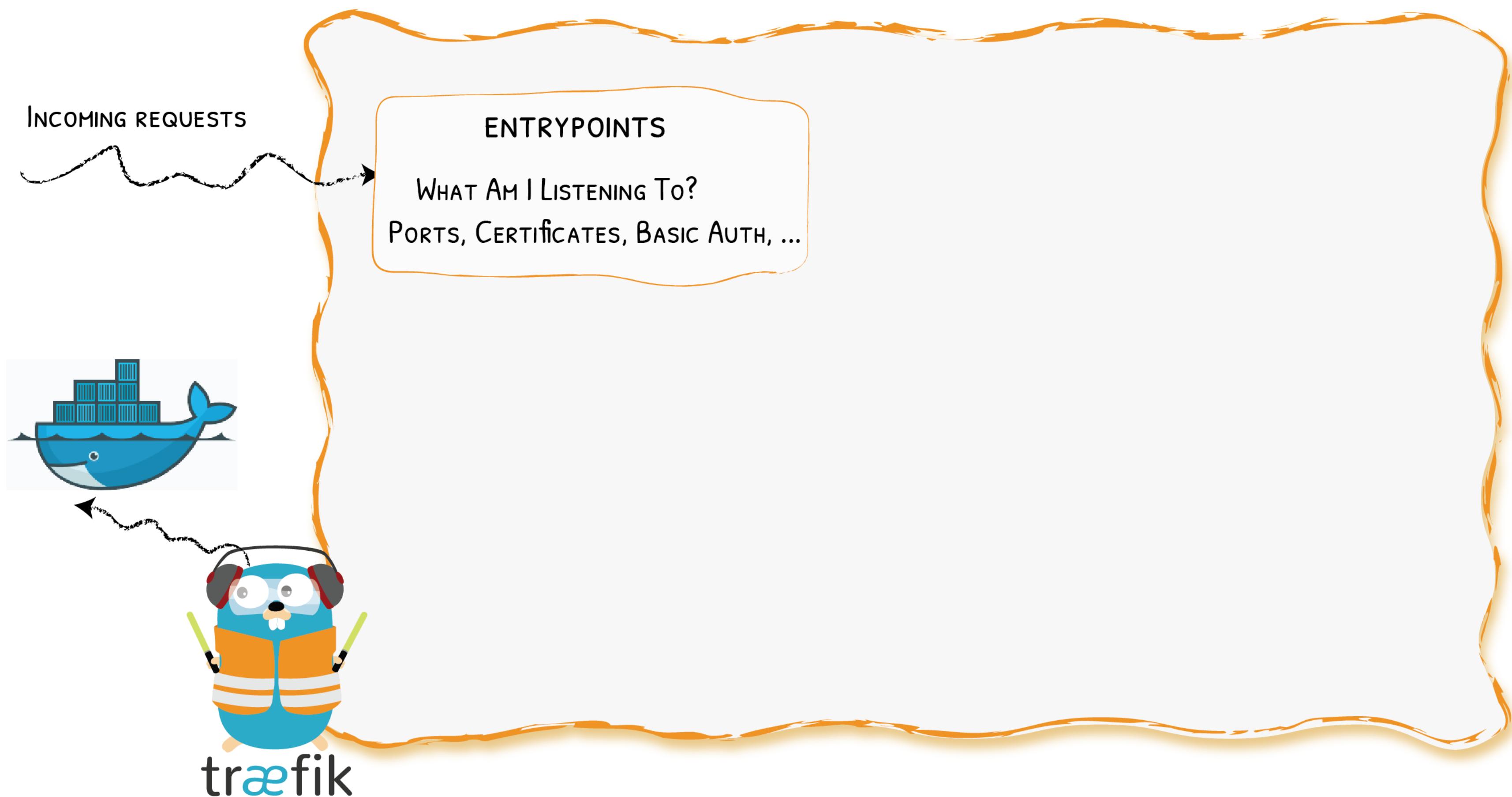
# Let's Simplify



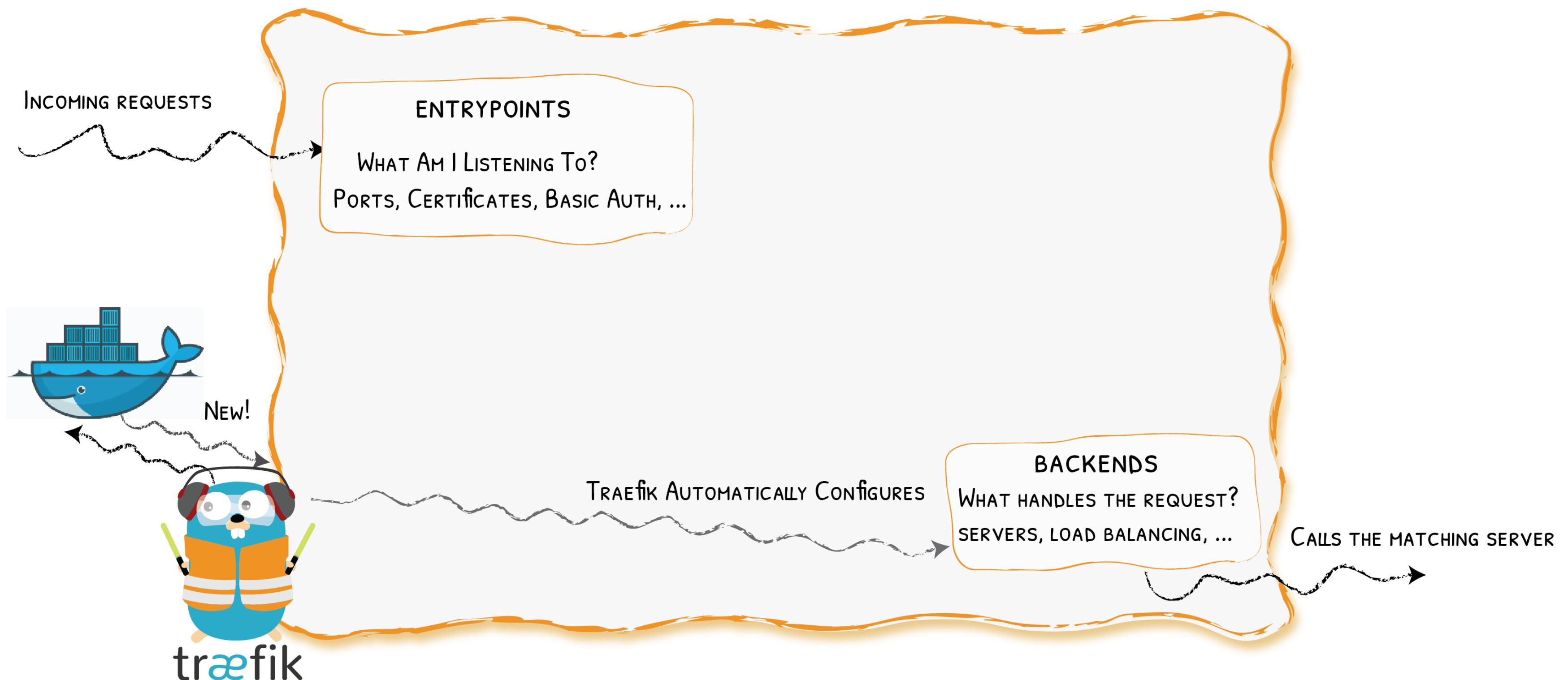
# Providers



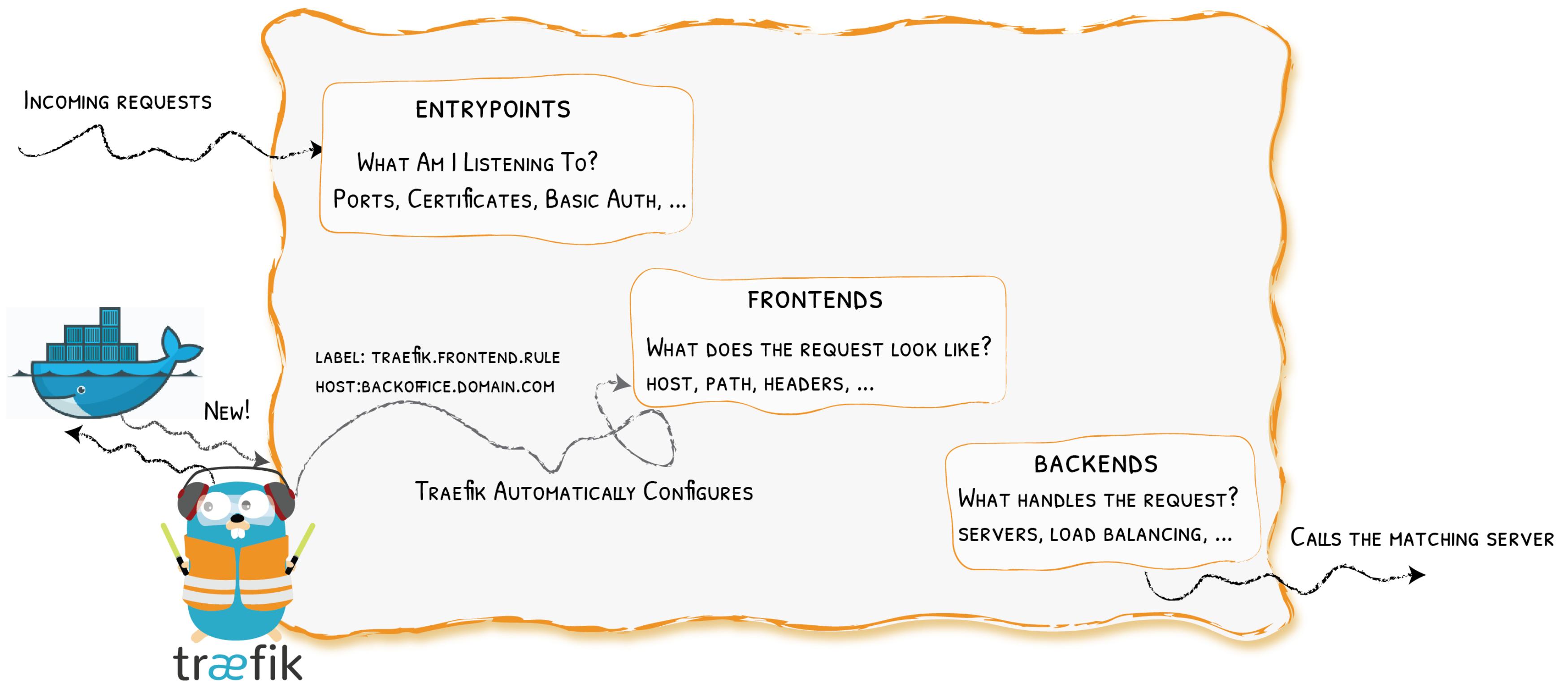
# Entrypoints



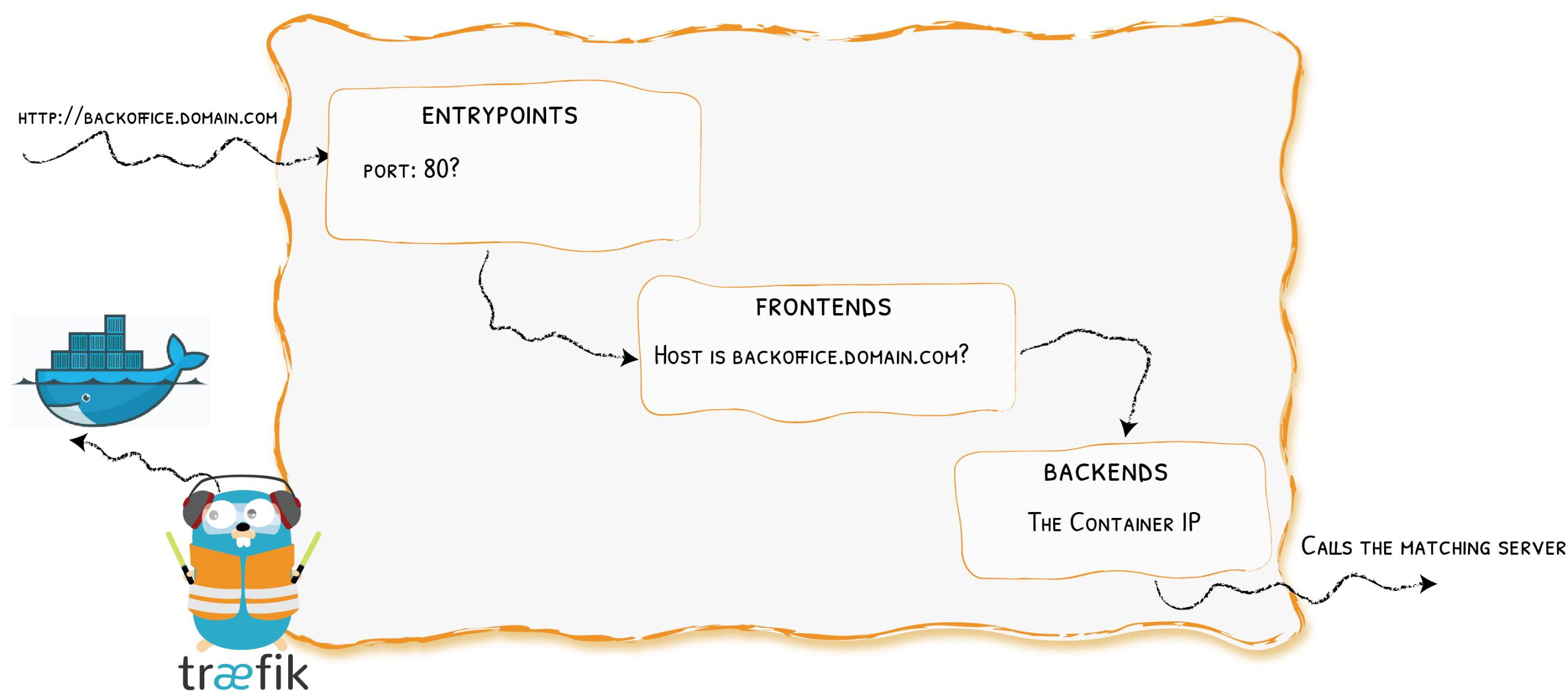
# Backends



# Frontends



# In Practice



Show Me The Configuration!

# Keep It Simple

- With 

```
entrypoint:  
  image: traefik:v1.7  
  command:  
    - "--docker"  
    - "--docker.domain=mycompany.org"  
    - "--acme.email=ssl-admin@mycompany.org"  
    - "--acme.httpChallenge.entryPoint=http"  
  # Or you could use a TOML file with "--configFile=/etc/traefik/traefik.toml  
volumes:  
  - /var/run/docker.sock:/var/run/docker.sock
```

# HTTPS For Everyone With Let's Encrypt



- TLS, DNS and HTTP challenges supported

# With 🐳: Simple Backend

```
# https://www.mycompany.org -> http://webserver:80/
webserver:
  image: nginx:alpine
  labels:
    - "traefik.frontend.rule=Host:www.mycompany.org"
```

# With Context

```
# https://mycompany.org/jenkins -> http://jenkins:8080/jenkins
jenkins:
  image: jenkins/jenkins:lts
  labels:
    - "traefik.frontend.rule=PathPrefix:/jenkins"
    - "traefik.port=8080" # Because 50000 is also exposed
  environment:
    - JENKINS_OPTS=--prefix=/jenkins
```

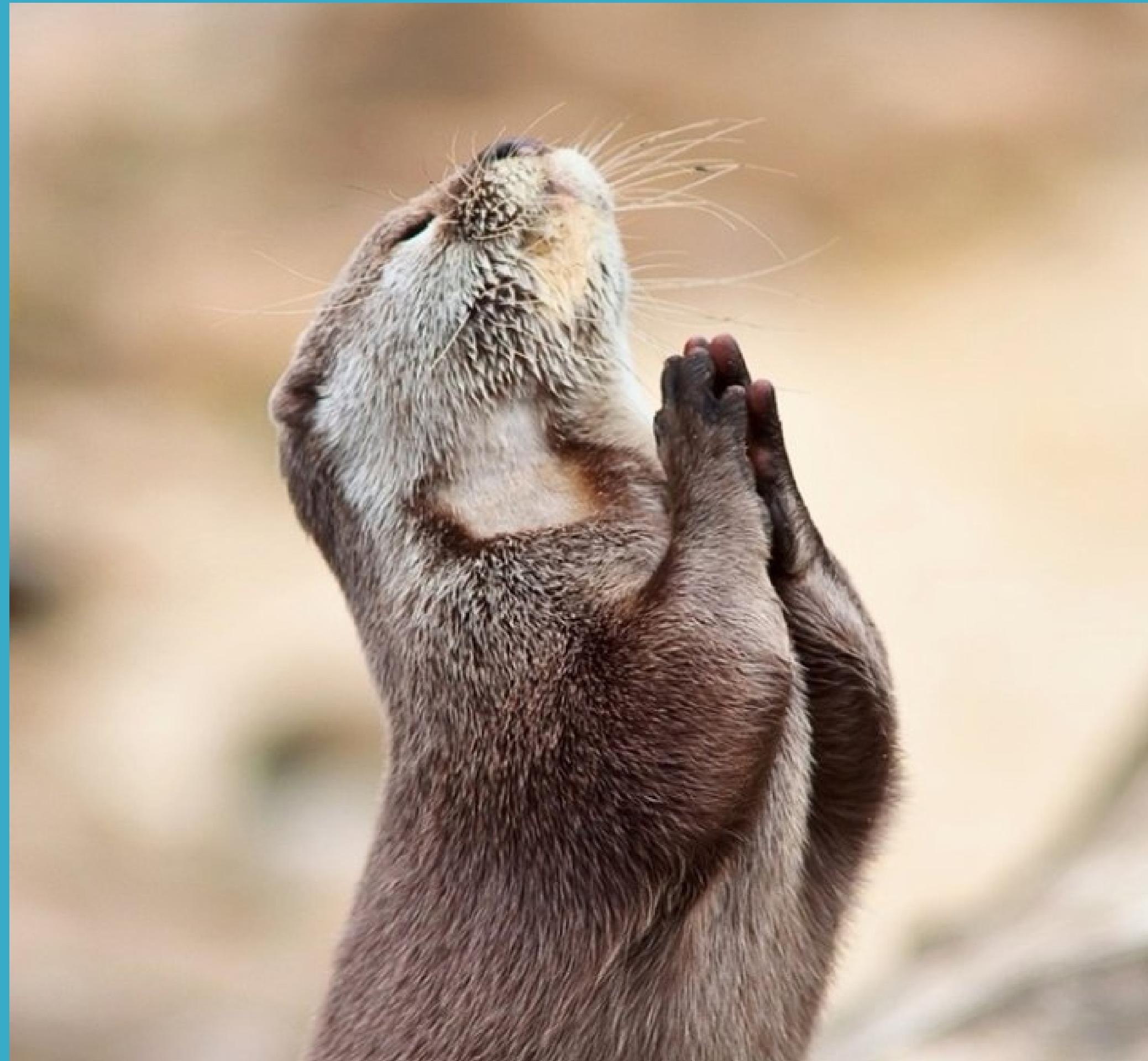
# With 🐟: Rewrites

```
# https://mycompany.org/gitserver -> http://gitserver:3000/
gitserver:
  image: gitea/gitea:1.5
  labels:
    - "traefik.frontend.rule=PathPrefixStrip:/gitserver"
    - "traefik.port=3000" # Because 22 is also exposed
```

# With 🐳: Websockets

```
# https://mycompany.org/webterminal -> http://webterminal:7681/
webterminal:
  image: ts10922/ttyd
  labels:
    - "traefik.frontend.rule=PathPrefixStrip:/webterminal"
  expose:
    - "7681"
```

# Demo



# Traefik With ⚓

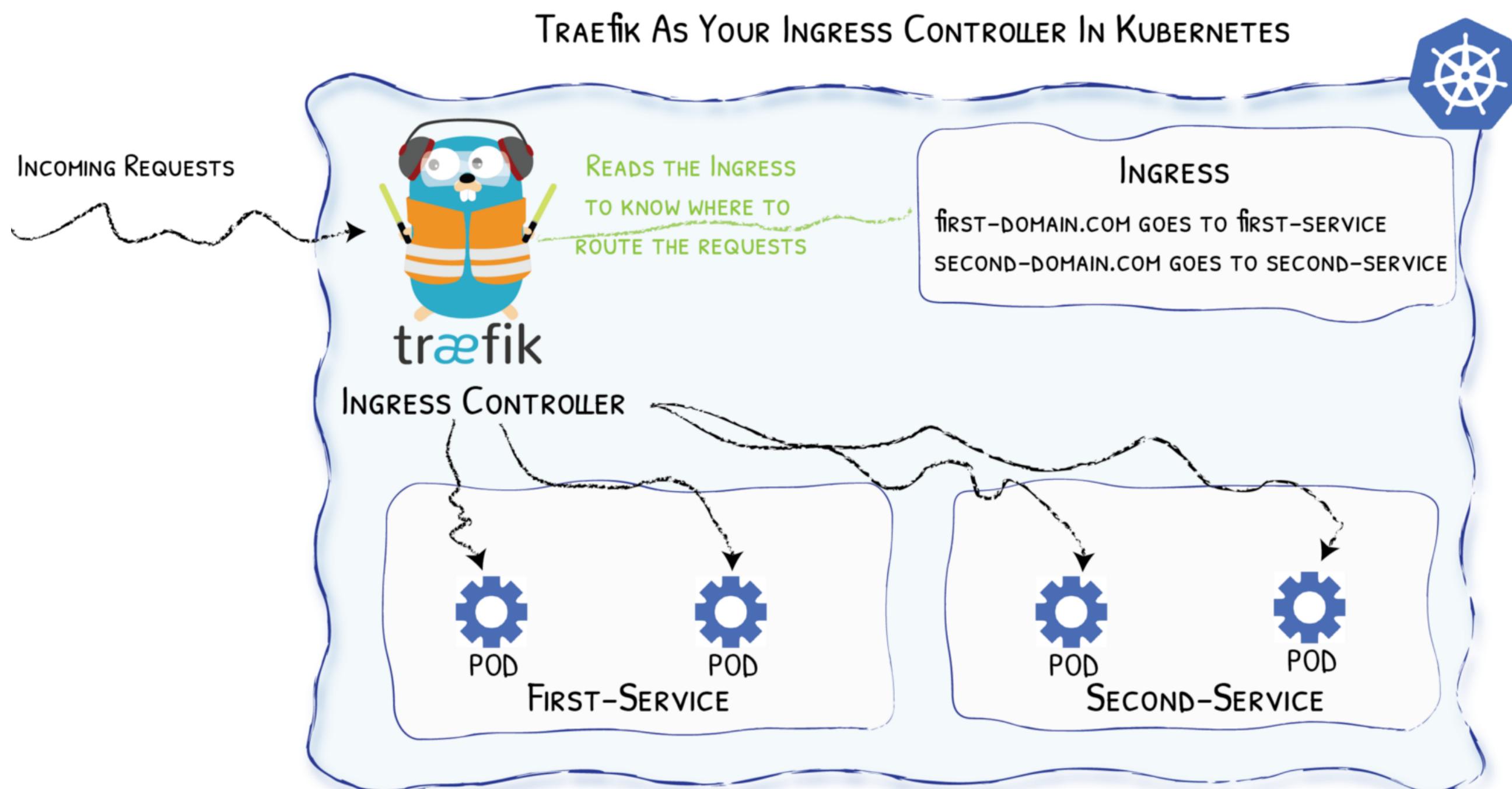


Diagram from <https://medium.com/@geraldcroes>

# Did You Say YAML?

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

# We Missed Talking About...

A cloud of network-related terms in various colors, including:

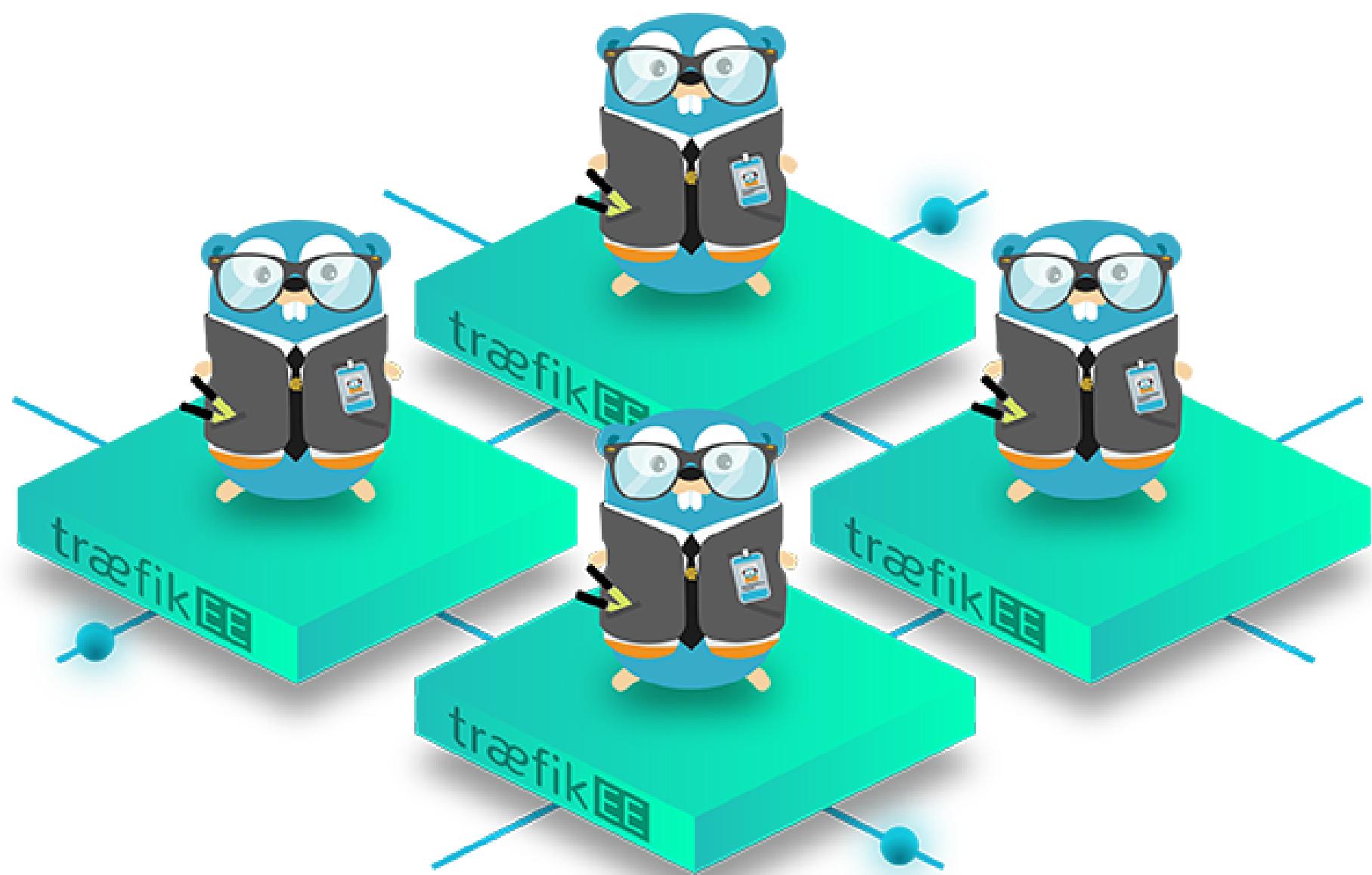
- MESOS
- ZIPKIN
- LIMITING
- KUBERNETES
- Dynamic Metrics
- HTTP ERROR
- CERTIFICATE
- TLS Reverse-Proxy
- HEADERS
- GRPC
- S3 BUCKET
- CIRCUIT BREAKERS
- DYNAMIC/WILDCARD
- Security Configurations
- Tracing PROXY
- SECRETS
- PROMETHEUS
- JAEGER
- WEBSOCKETS
- SSL
- FORWARDING
- REDIRECTS
- DOCKER
- PROTOCOL
- CHECKS
- CLUSTER AUTH
- HSTS
- RATE
- CONSUL
- SWARM MODE
- HEALTH

# The Herd



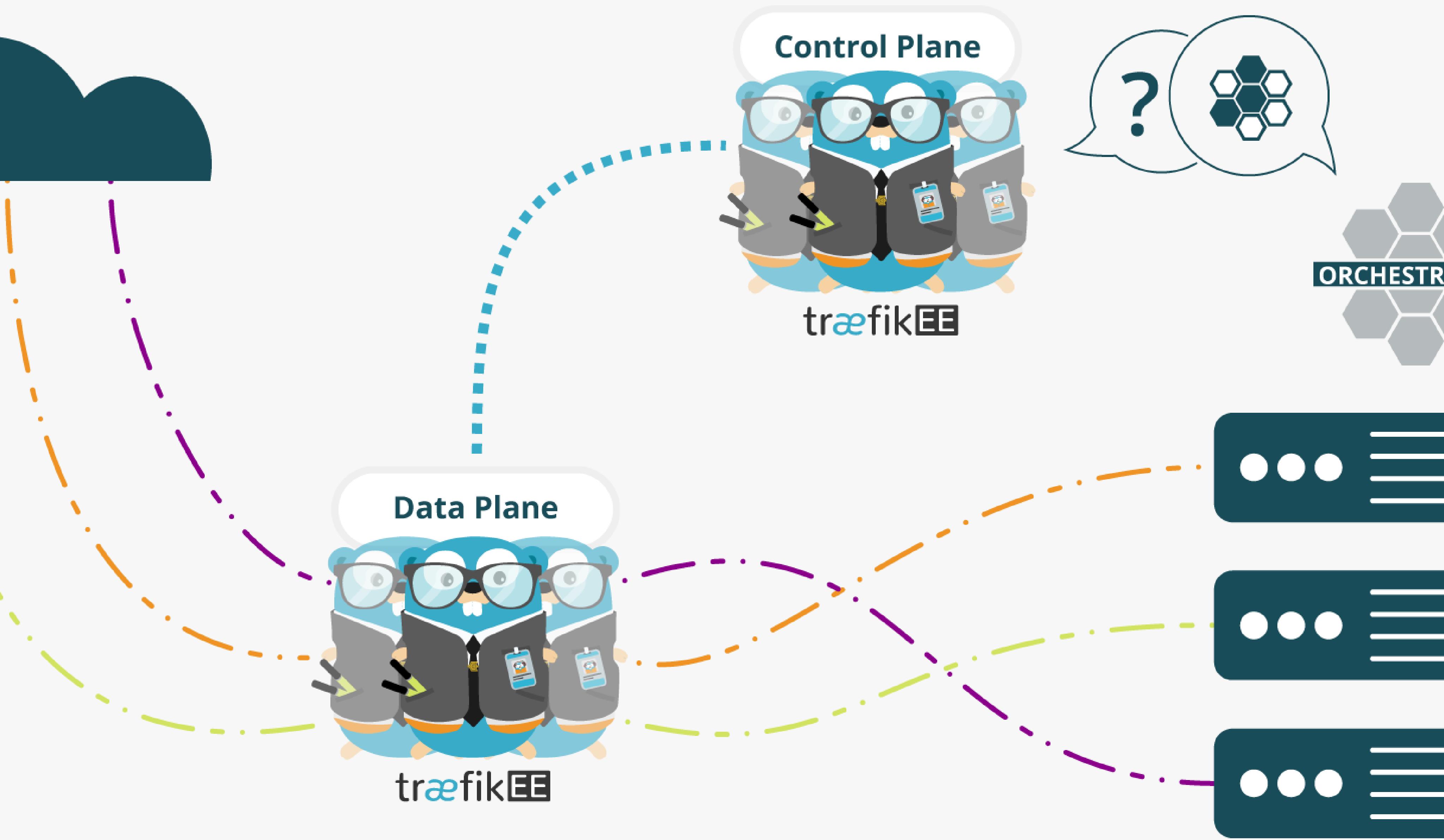
You came to the wrong neighbour

# Traefik Comes In Herd



INTERNET

TO YOUR INFRA



# HIGH AVAILABILITY

traefik ENTERPRISE EDITION



# SECURITY

traefik ENTERPRISE EDITION

# SCALABILITY

traefik ENTERPRISE EDITION

# 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>

# BACK toTRAEFIK 2.0

PART →



# Revamped Documentation

The screenshot shows the Traefik documentation website. At the top, there's a navigation bar with a search bar, a GitHub link (21k Stars - 2.1k Forks), and a logo. On the left, a sidebar lists navigation links: Welcome, Getting Started, Configuration Discovery, Routing & Load Balancing, HTTPS & TLS, Middlewares, Operations, Observability, Contributing, and Glossary. The main content area features a large, colorful diagram titled "Welcome". The diagram illustrates Traefik's role as an "Edge Router" that receives requests from the internet (e.g., API.DOMAIN.COM, DOMAIN.COM/WEB, BACKOFFICE.DOMAIN.COM) and routes them to various infrastructure components like Kubernetes, Mesos, Docker, and Backoffice servers. It highlights features such as automatic and dynamic routing, load balancing, auto-discovery, metrics tracing, and logs. Below the diagram, a text block describes Traefik as an open-source Edge Router.

**Welcome**

FROM THE INTERNET

API.DOMAIN.COM  
DOMAIN.COM/WEB  
BACKOFFICE.DOMAIN.COM

træfik

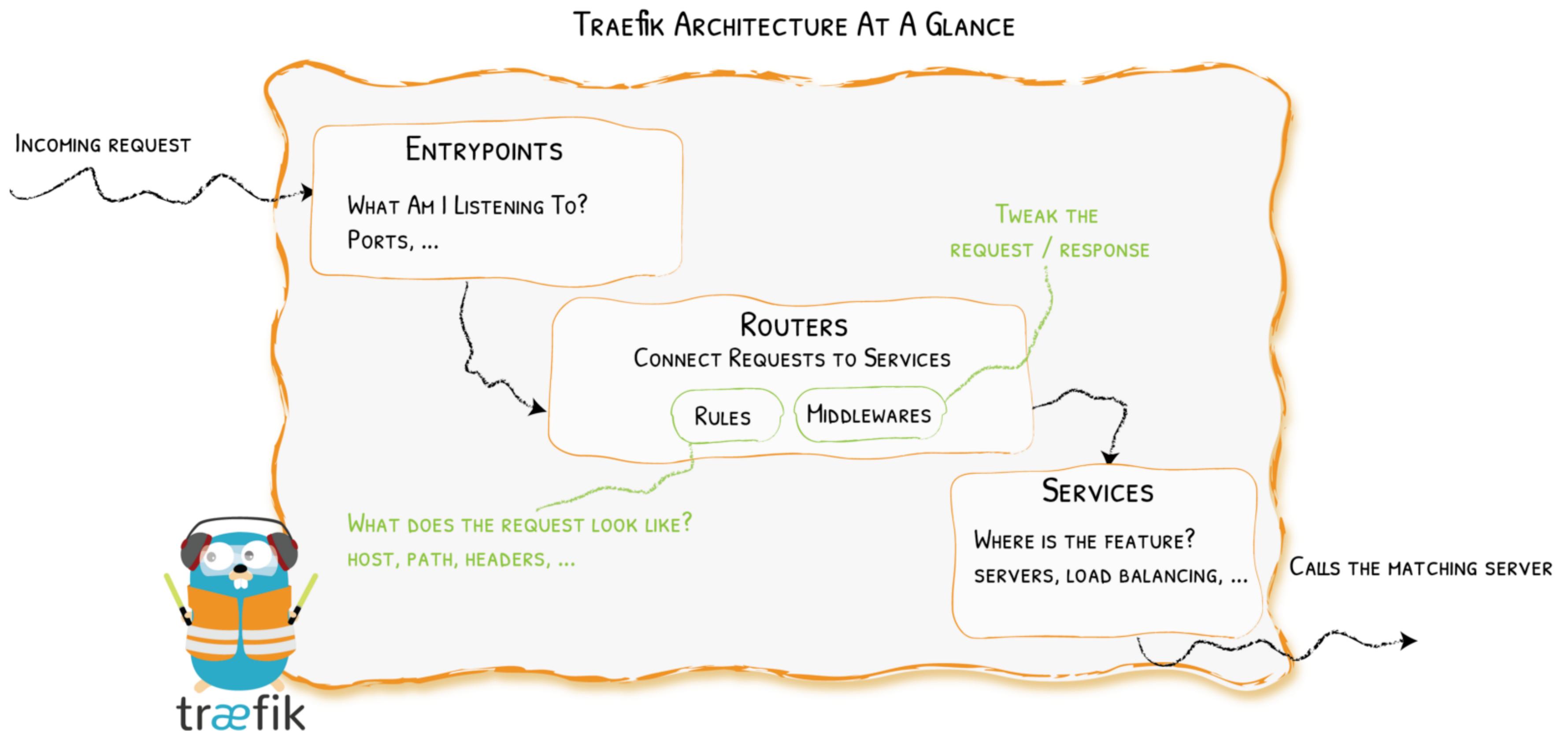
To Your Infrastructure

API SERVER  
KUBERNETES  
WEB SERVER  
MESOS  
PRIVATE SERVER  
BACKOFFICE SERVER - 1  
BACKOFFICE SERVER - 2  
DOCKER

HTTPS  
Metrics  
Tracing  
Logs  
Automatic & Dynamic Routing  
Load Balancing  
Auto Discovery

Traefik is an [open-source](#) *Edge Router* that makes publishing your services a fun and easy experience. It receives requests on behalf of your system and finds out which components are responsible for handling them.

# Clarified Concepts



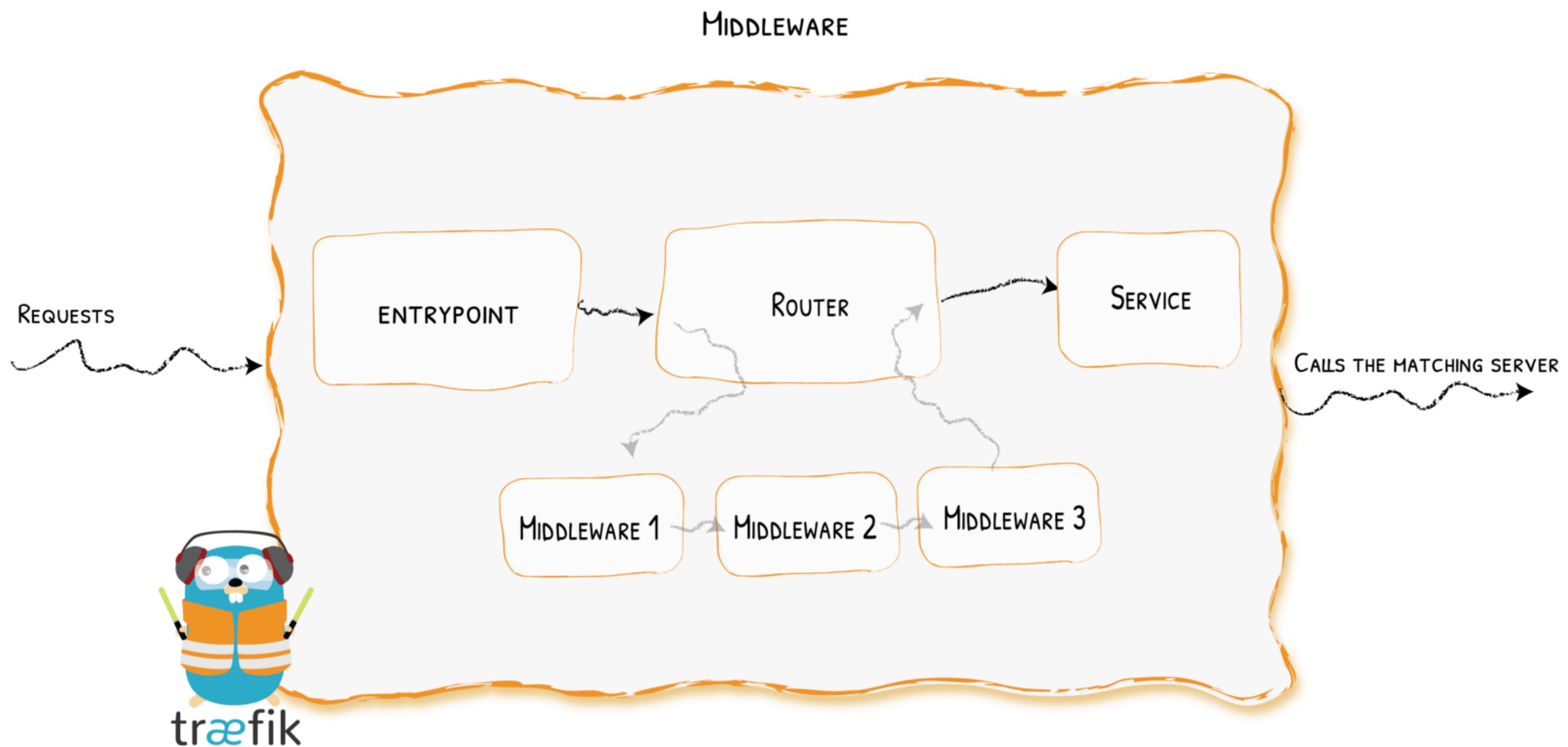
# Expressive Routing Rule Syntax



Host(`api.dom`) || (Host(`dom`) && Path(`/api`))

```
# Send both requests to backend service:  
#   https://api.mycompany.com/v2  
#   https://api-v2.mycompany.com  
  
rule=(Host('api.mycompany.com') && PathPrefix('/v2')) || Host('api-v2.mycompany.com')
```

# Middlewares





HTTP  
&  
TCP

# Quick Glance

```
[entrypoints]
  [entrypoints.web-secure]
    address = "":443
```

```
[http]
  [http.routers.to-service-1]
    rule = "Host(`demo.containous.cloud`)"
    service = "service-1"
  [http.routers.to-service-1.tls] # terminates the tls connection at HTTP
```

```
[tcp]
  [tcp.routers.to-service-2]
    rule = "HostSNI(`demo.containous.cloud`)"
    service = "service-2"
  [tcp.routers.to-service-2.tls] # terminates the tls connection at TCP
```

```
[tcp.routers.to-service-3]
  rule = "HostSNI(`demo.containous.cloud`)"
  service = "service-3"
  [tcp.routers.to-service-3.tls]
    passthrough = true # sends encrypted data "as is" to service-3
```

## *And So Much More...*

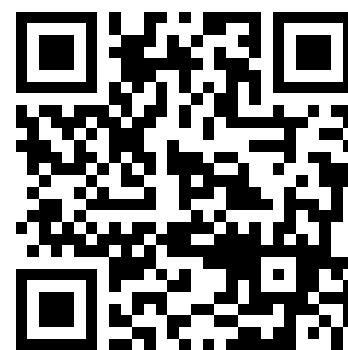
- Learn more on the blog post
- Call for contribution: Grab it, Try it, and give us your feedback!



# Thank You!

 @michaelmatur

 mmatur



- Slides (HTML): <https://containous.github.io/slides/riga-devdays-2019>
- Slides (PDF): <https://containous.github.io/slides/riga-devdays-2019/slides.pdf>
- Source on : <https://github.com/containous/slides/tree/riga-devdays-2019>