How To Easily Deploy And Manage Your Microservices With Traefik And Maesh

Whoami

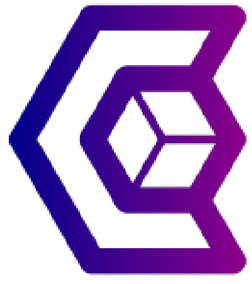
Michael Matur

- Solution Architect @ Containous
- Blacksmith on Traefik
- @michaelmatur
- mmatur

Containous

https://containo.us

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



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?

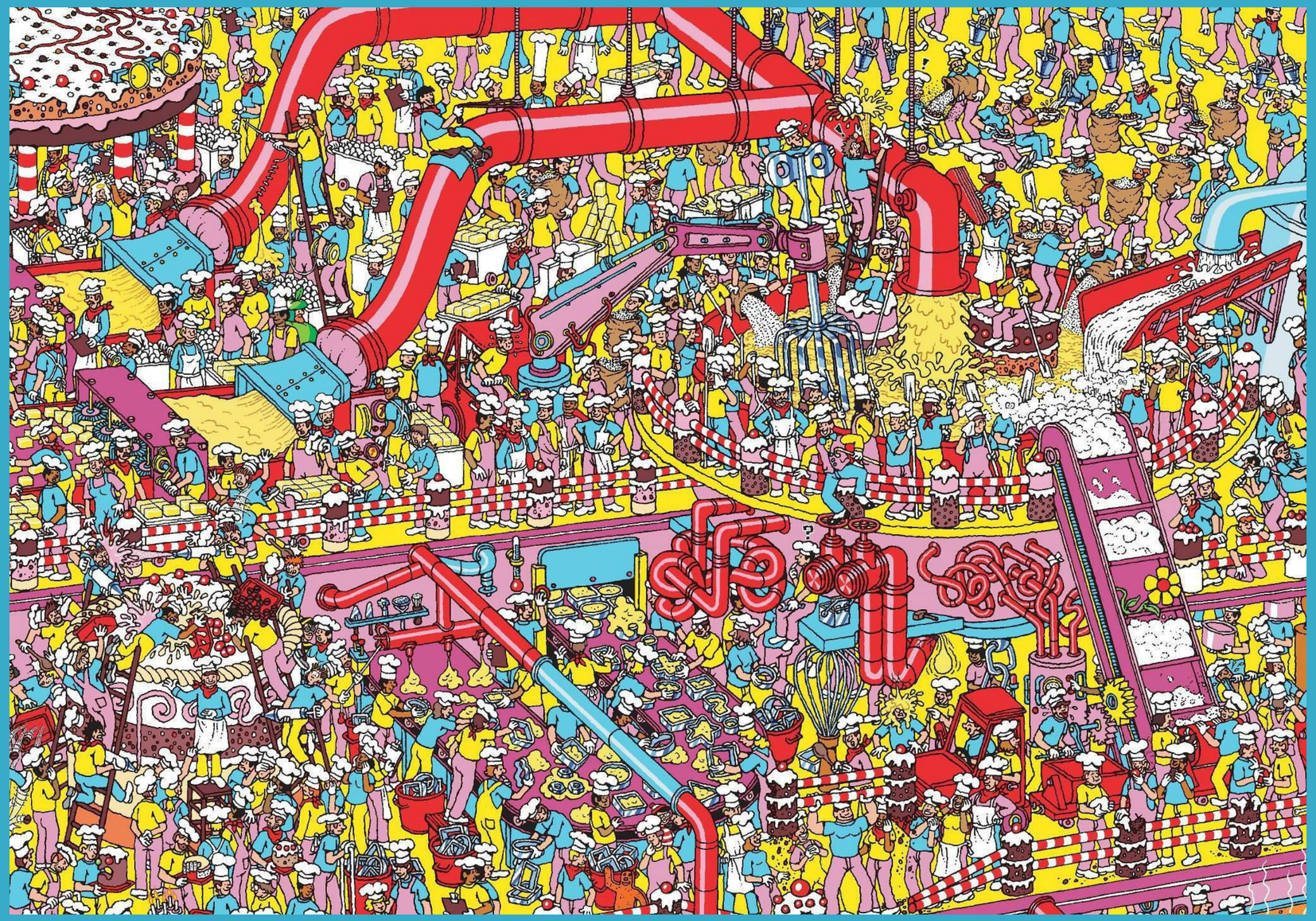
PROBABLY PIZZA-ORIENTED ARCHITECTURE

By @benorama

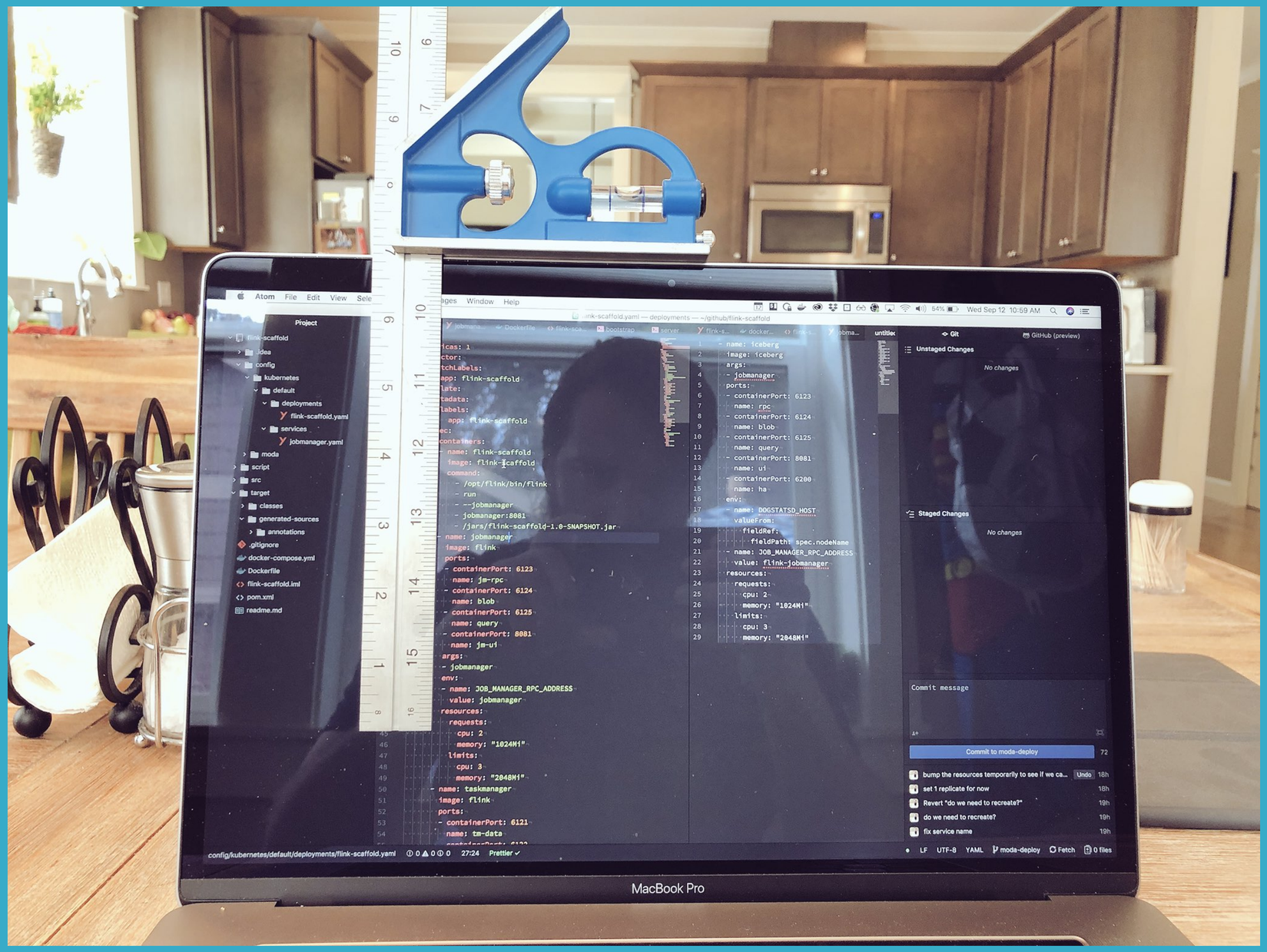
The Premise Of Microservices…

⋯And What Happens

Where’s My Service?



Tools Of The Trade

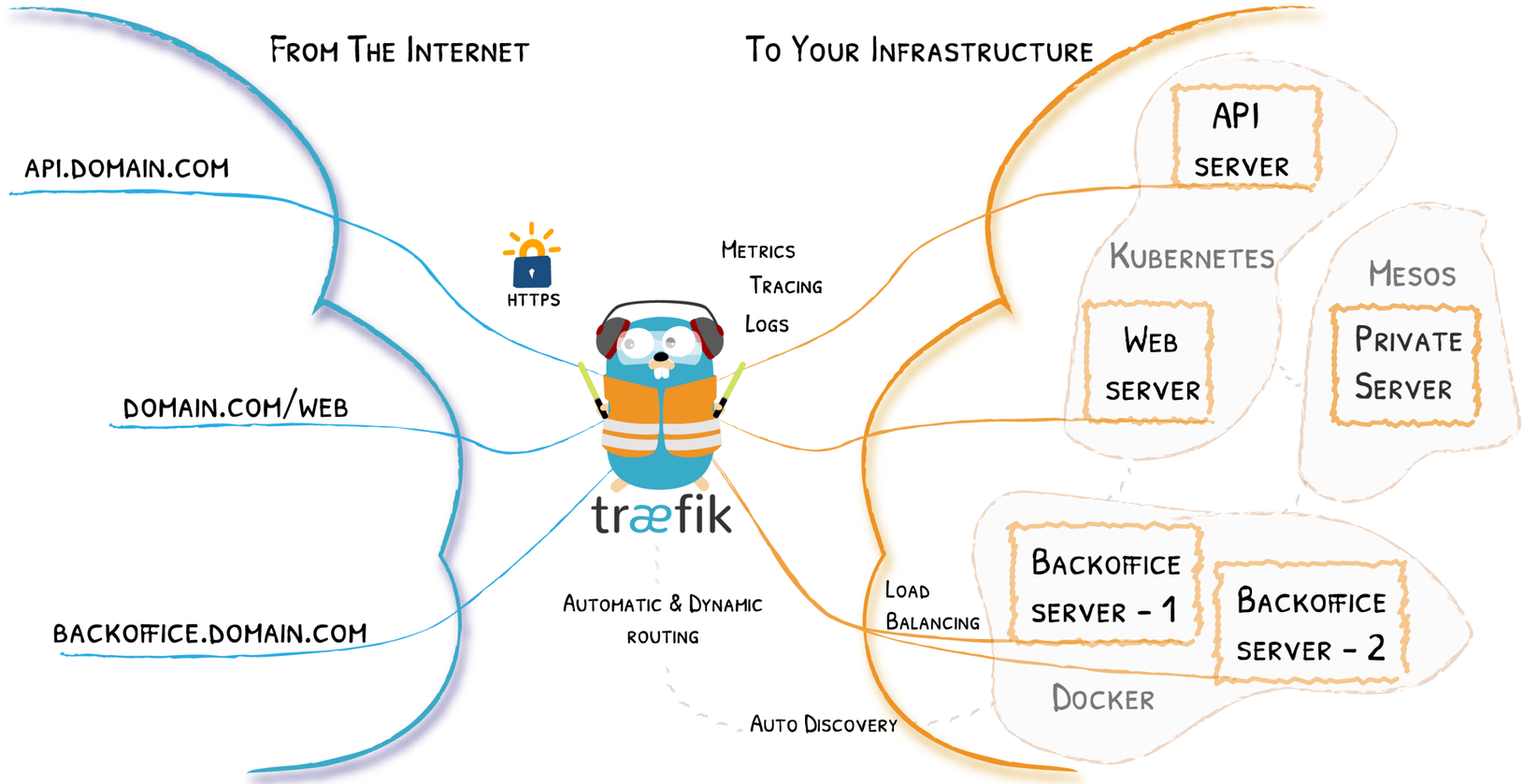


Source: https://twitter.com/Caged/status/1039937162769096704

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
- 26,000+ ☆ 1.3B+ ⬇ 400+ 👷
- Created in 2015, 4Y 🎂
- Current stable branch: `v2.0`

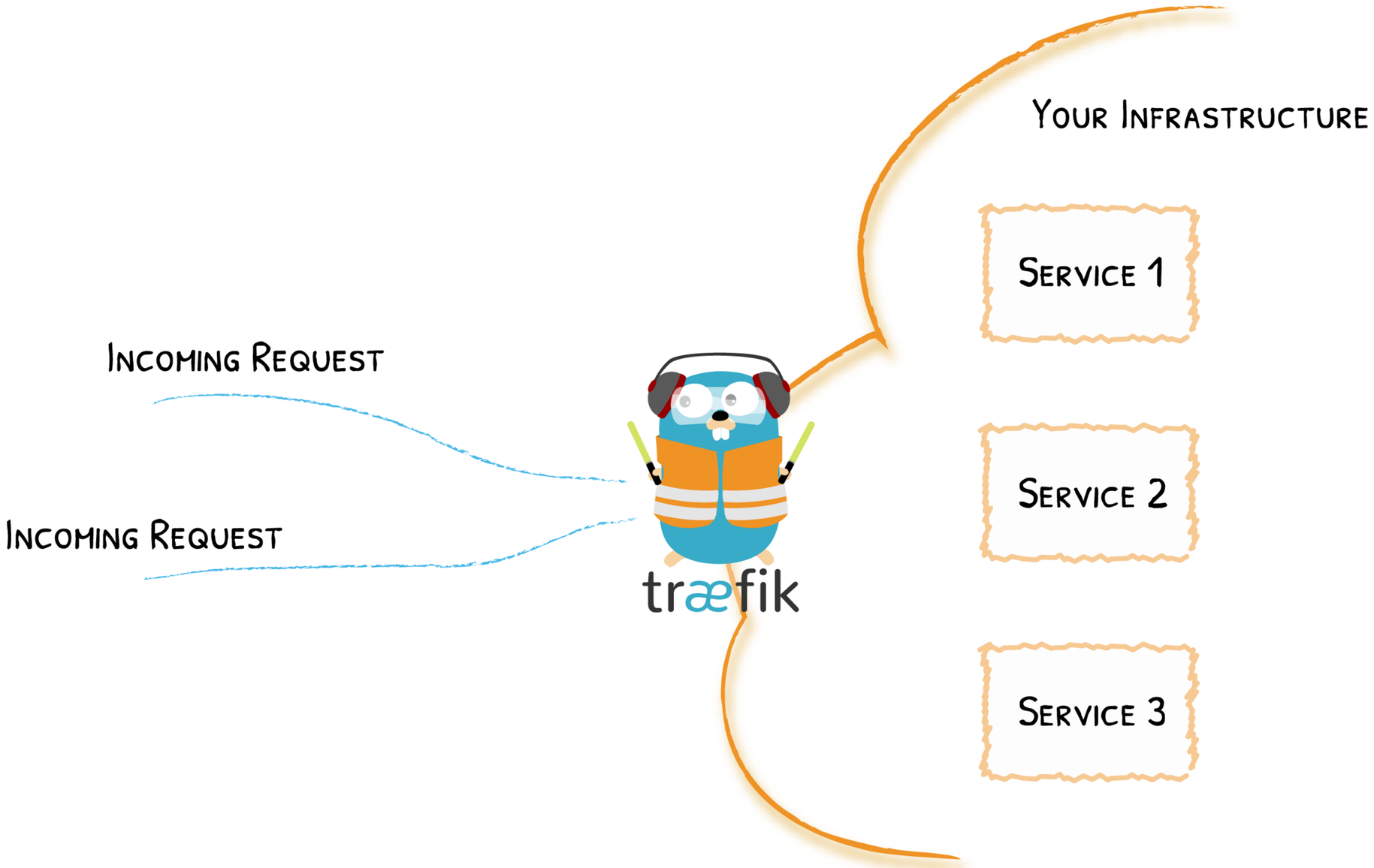
Traefik 2.0 Quick Overview

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

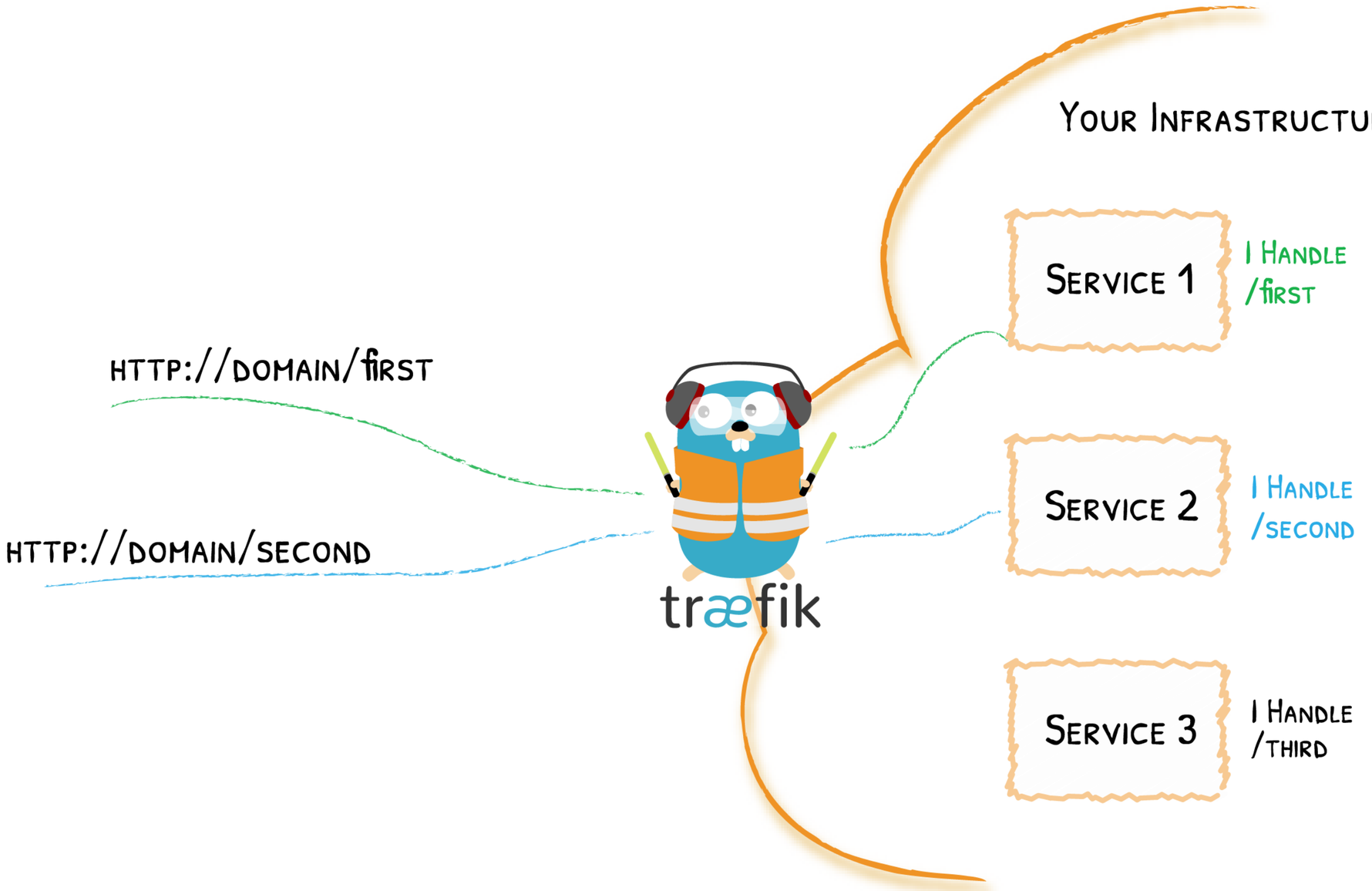
Learn more on the blog post

Traefik (V2.0) Core Concepts

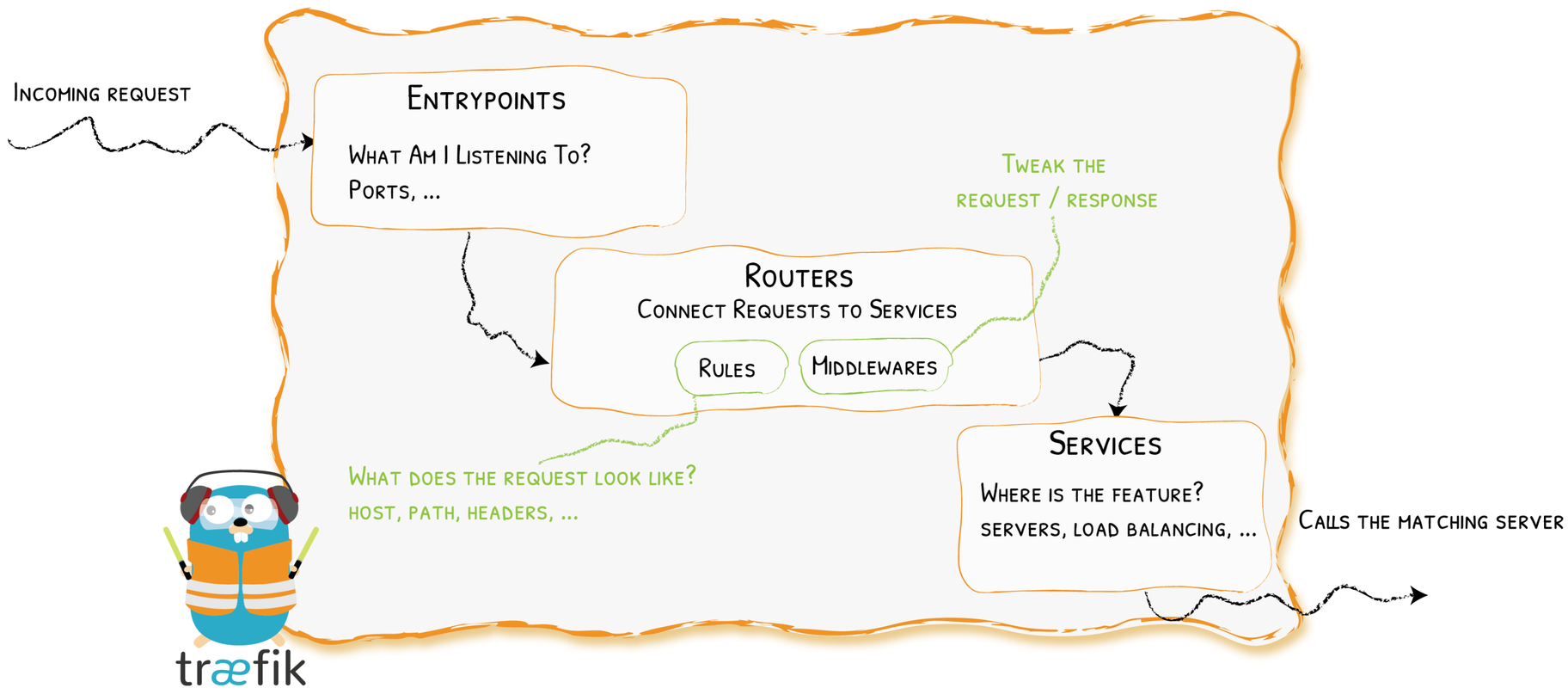
Traefik Is An Edge Router



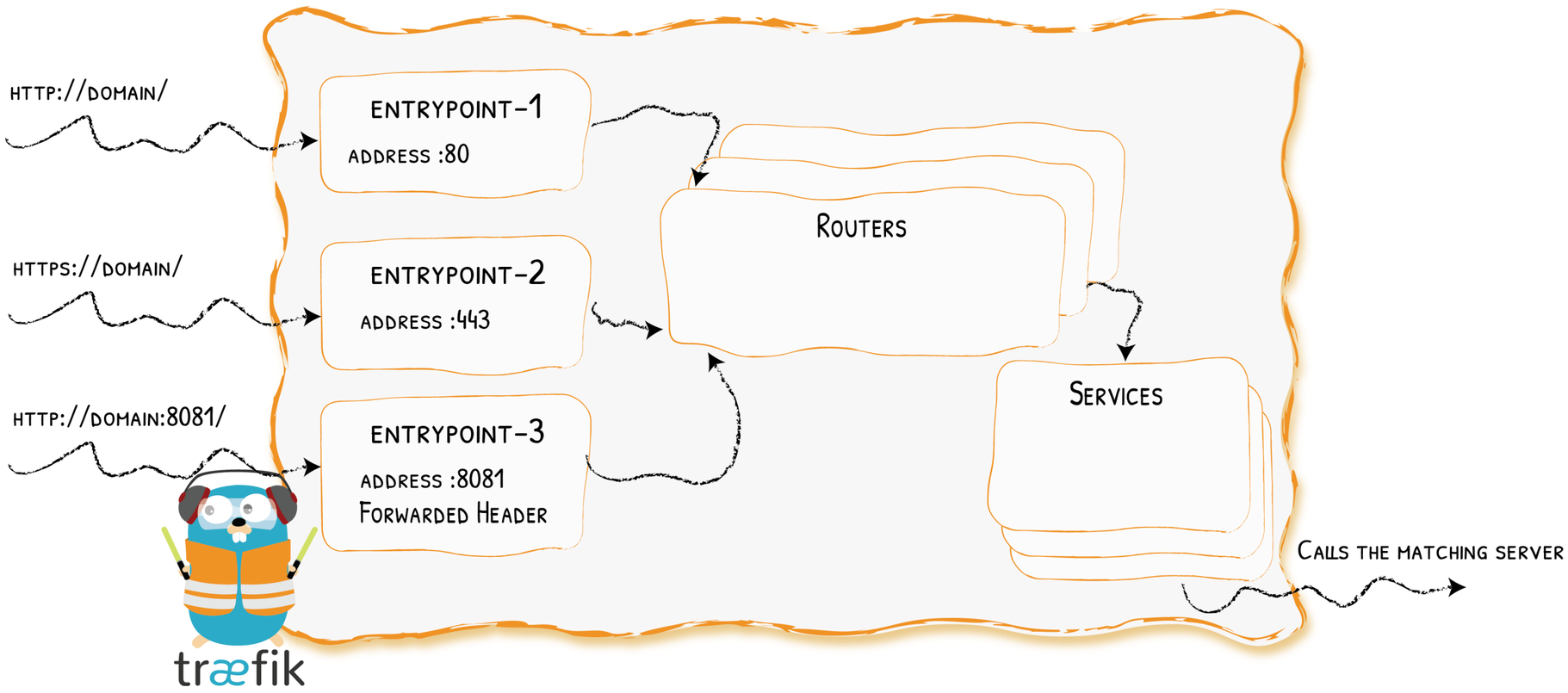
Dynamically Discovers Services



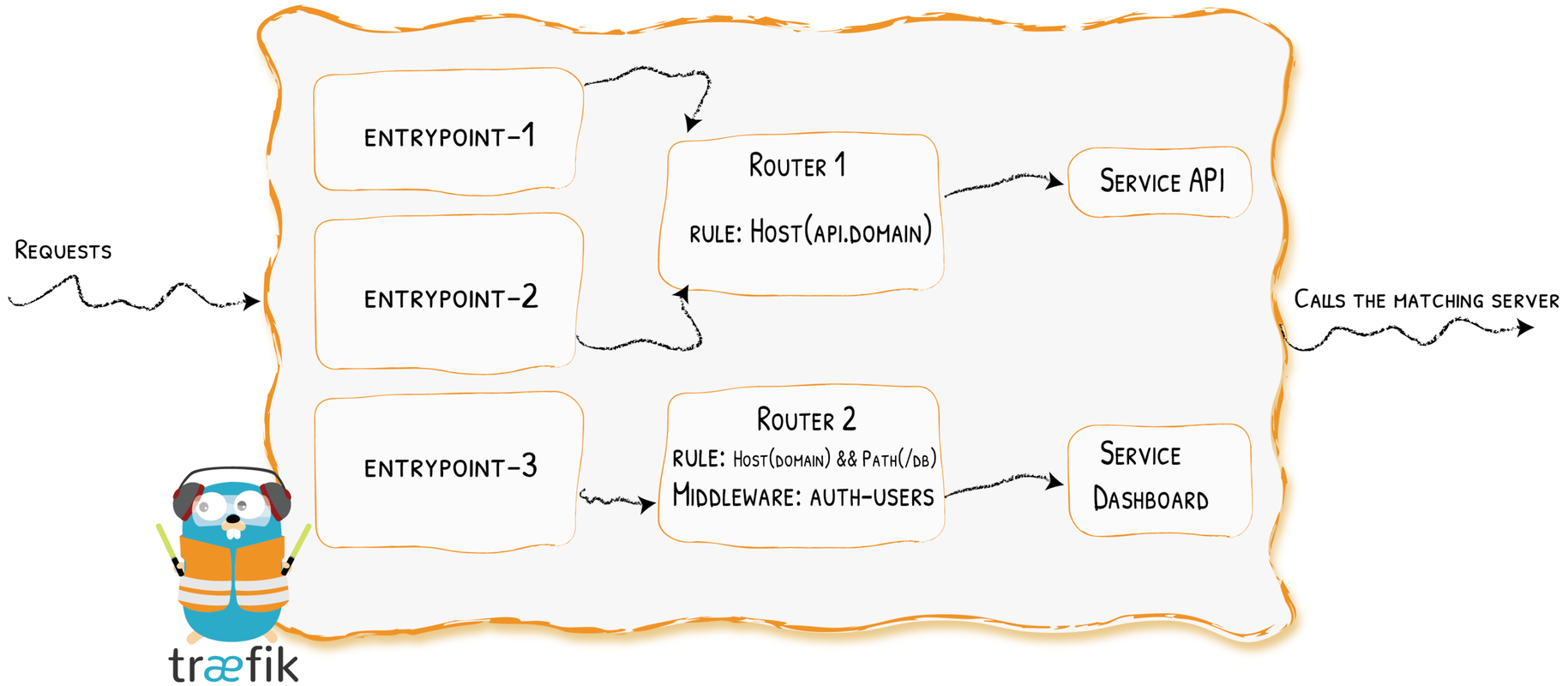
Architecture (V2.0) At A Glance



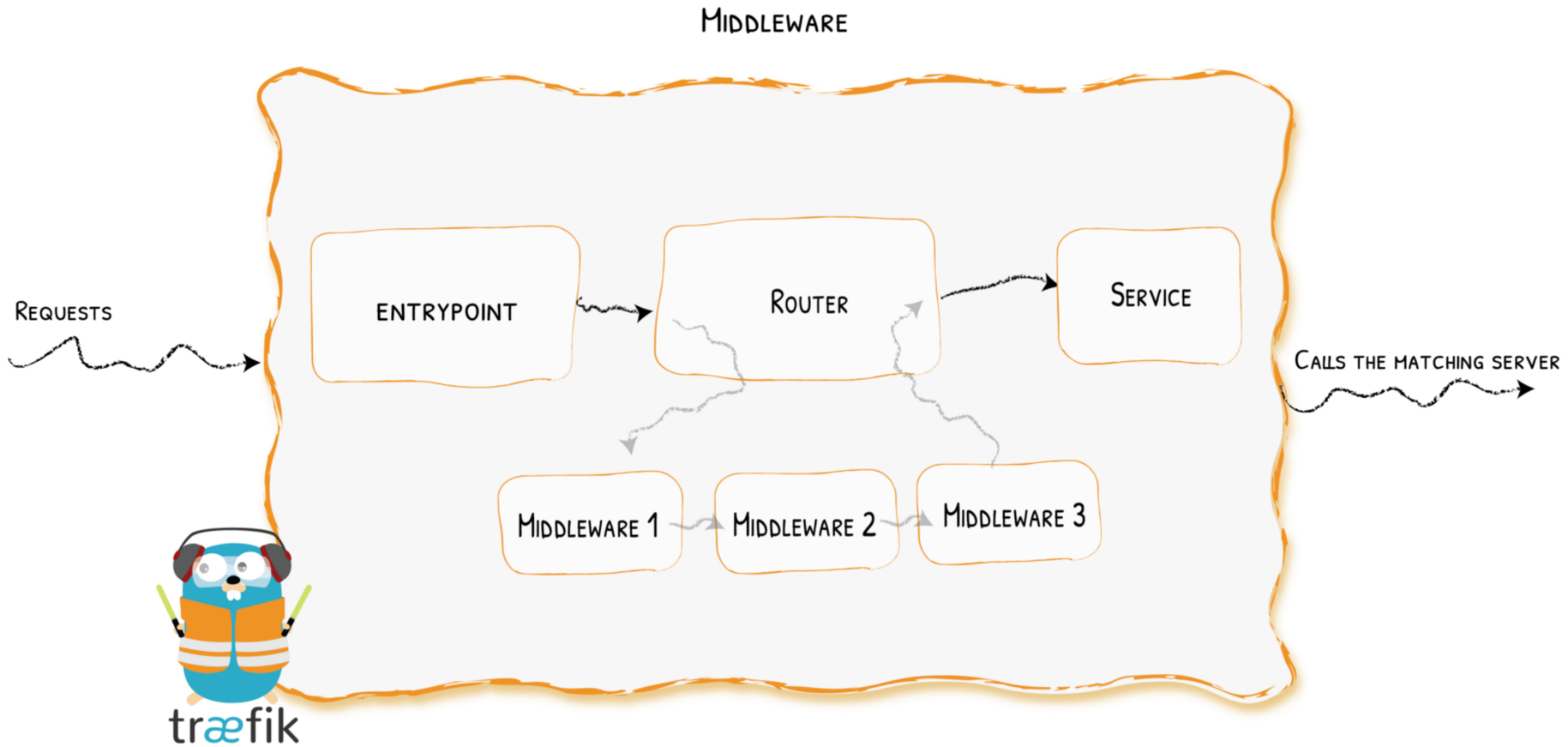
Entrypoints



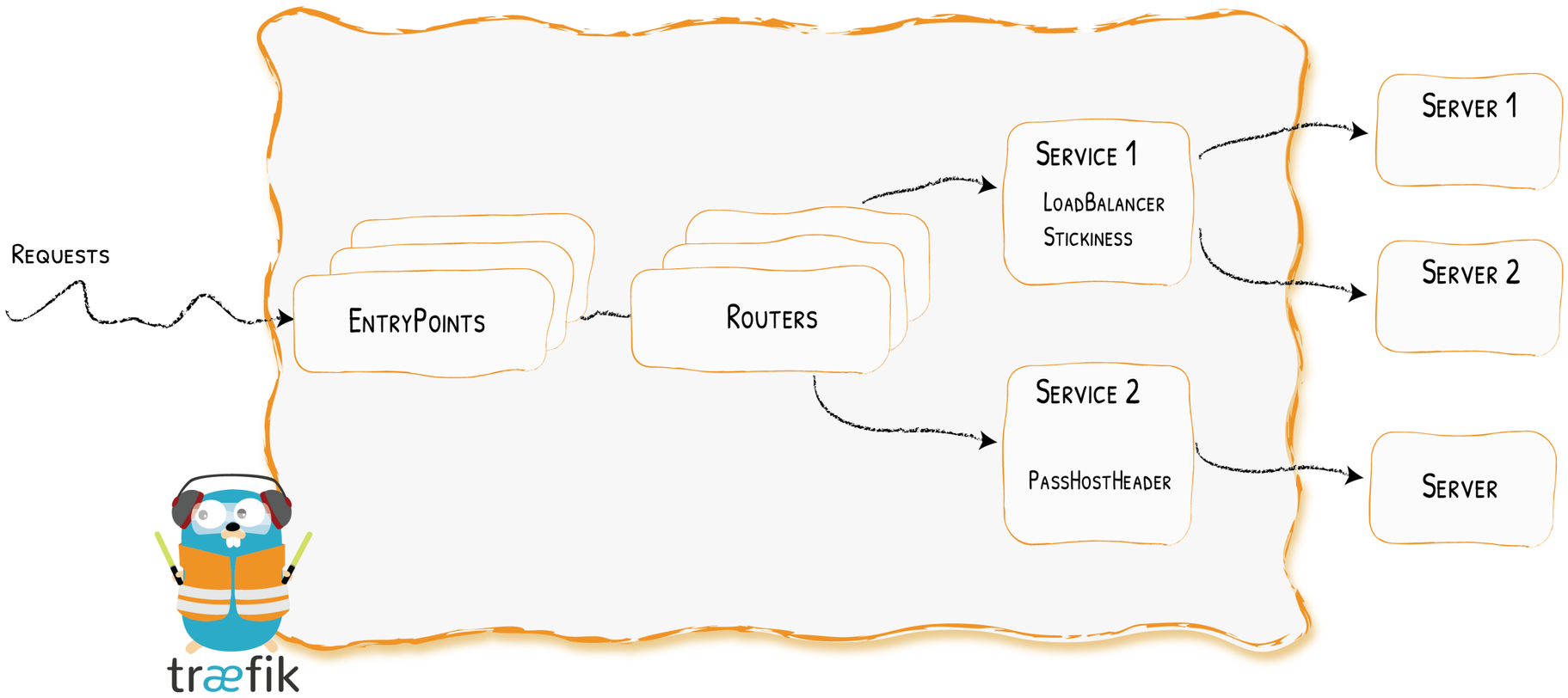
Routers


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

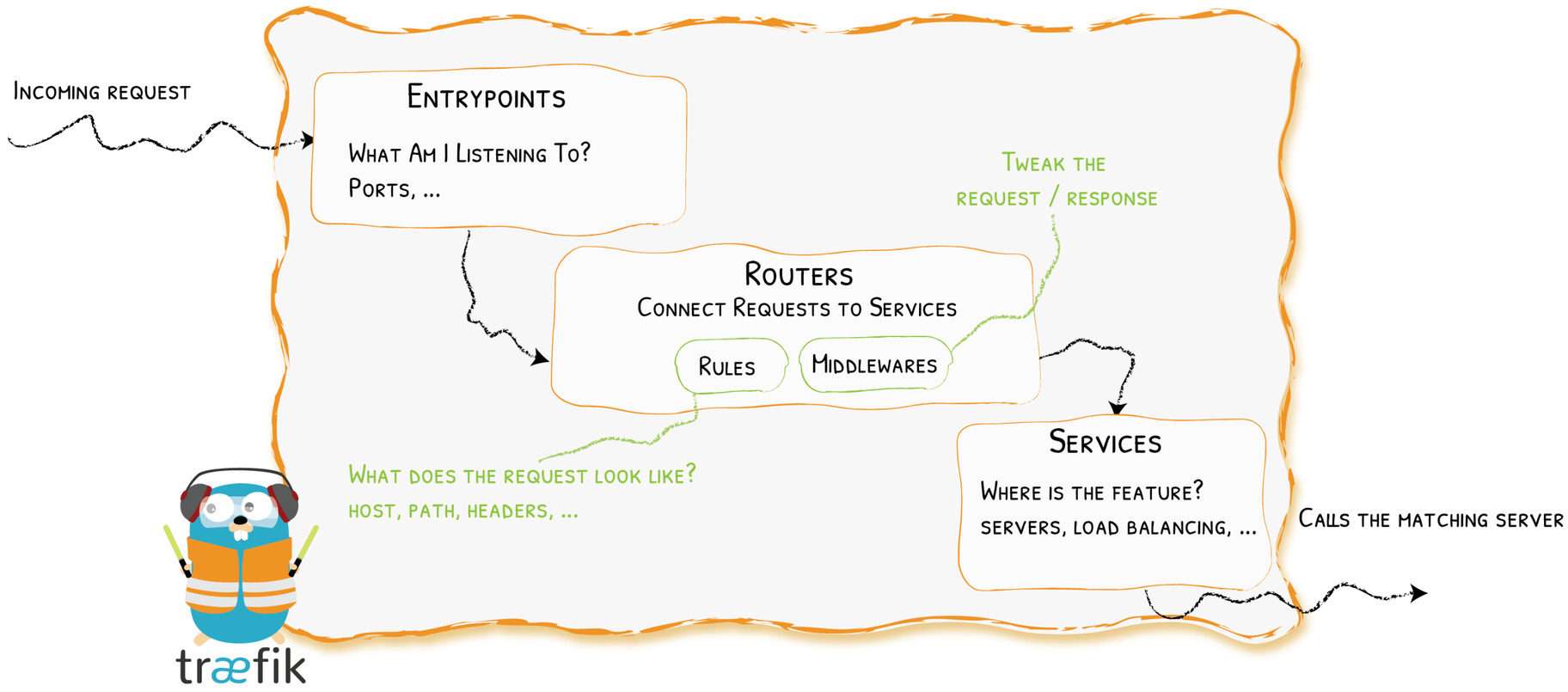
Middlewares



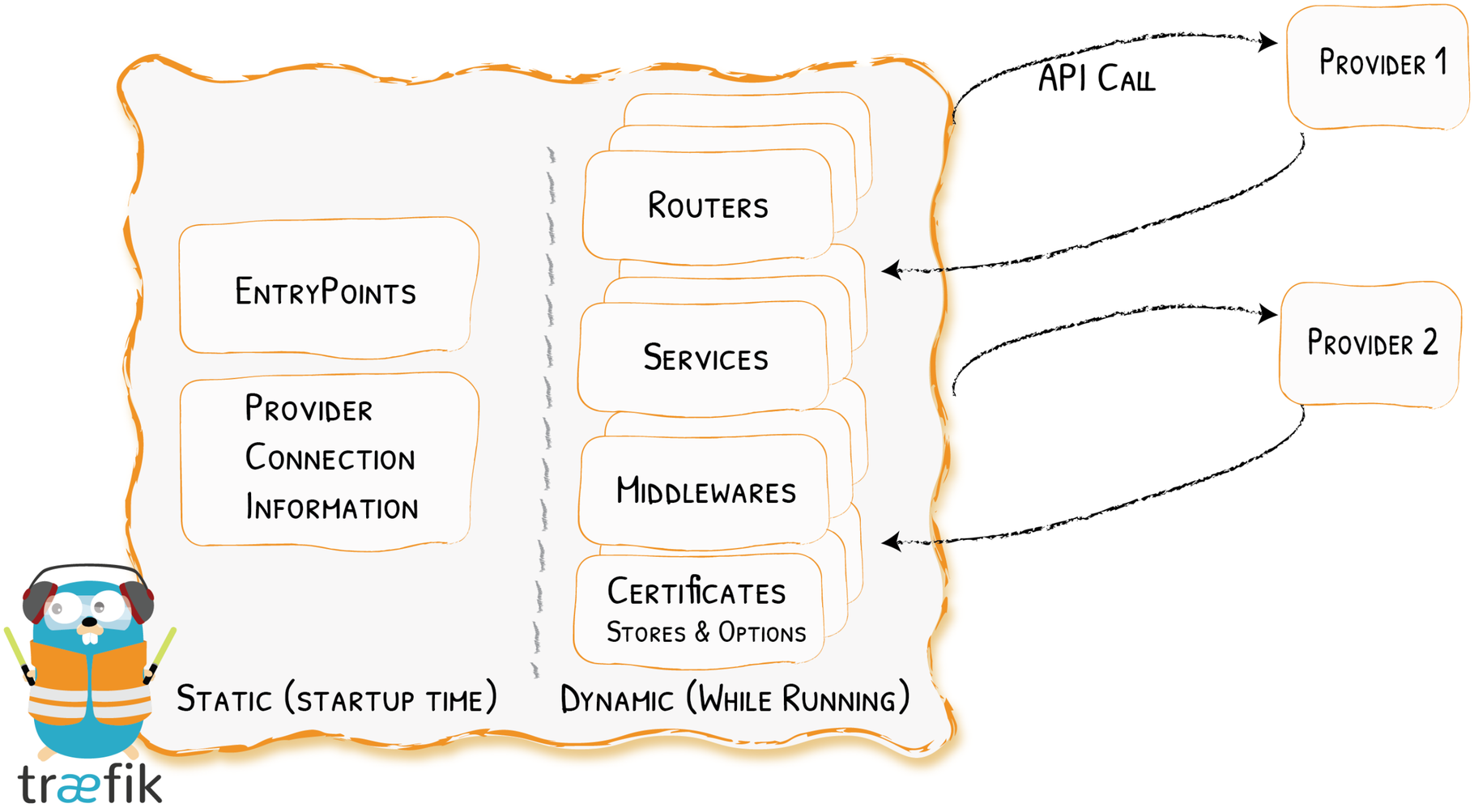
Services


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

Architecture (Again) At A Glance

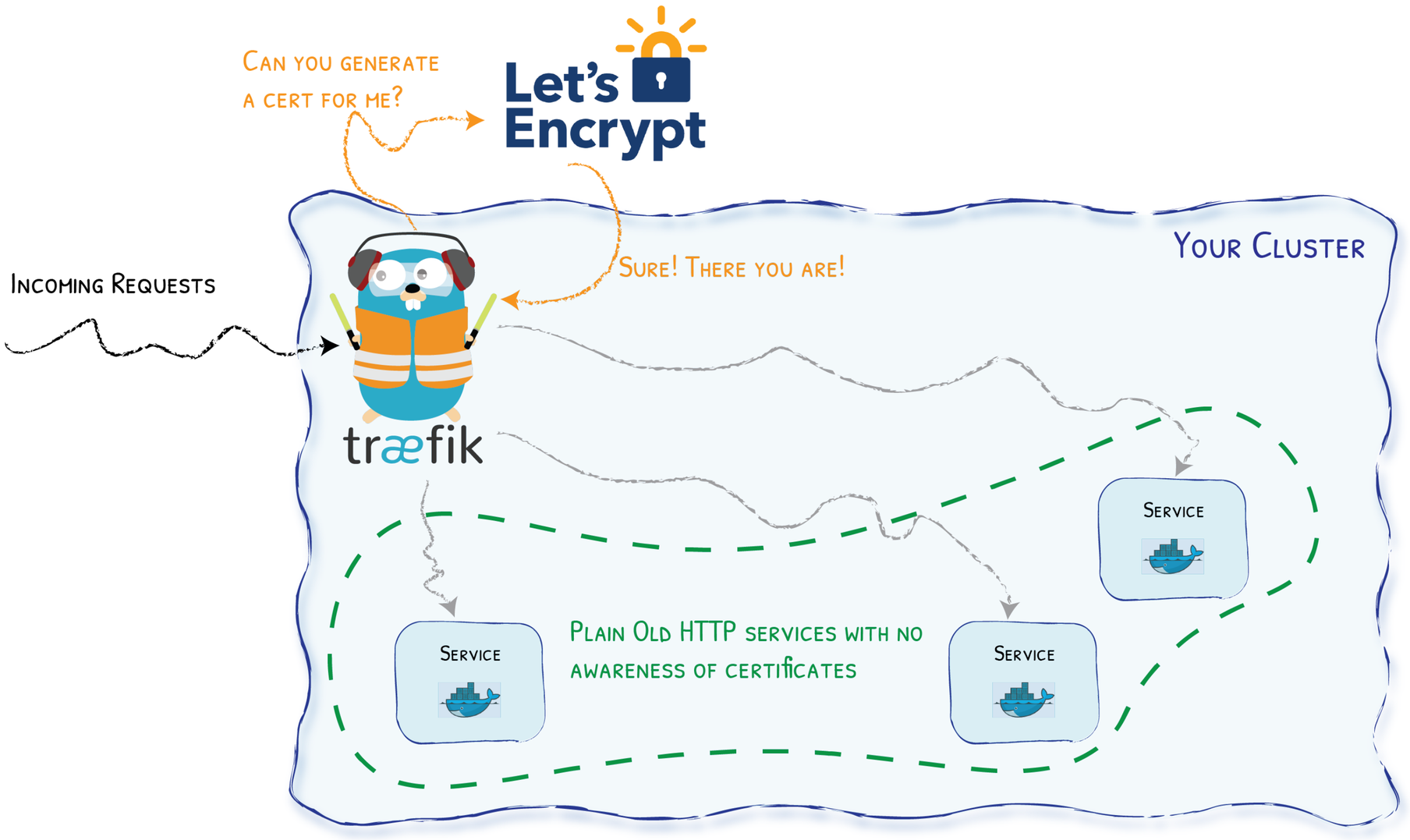


Static & Dynamic Configuration

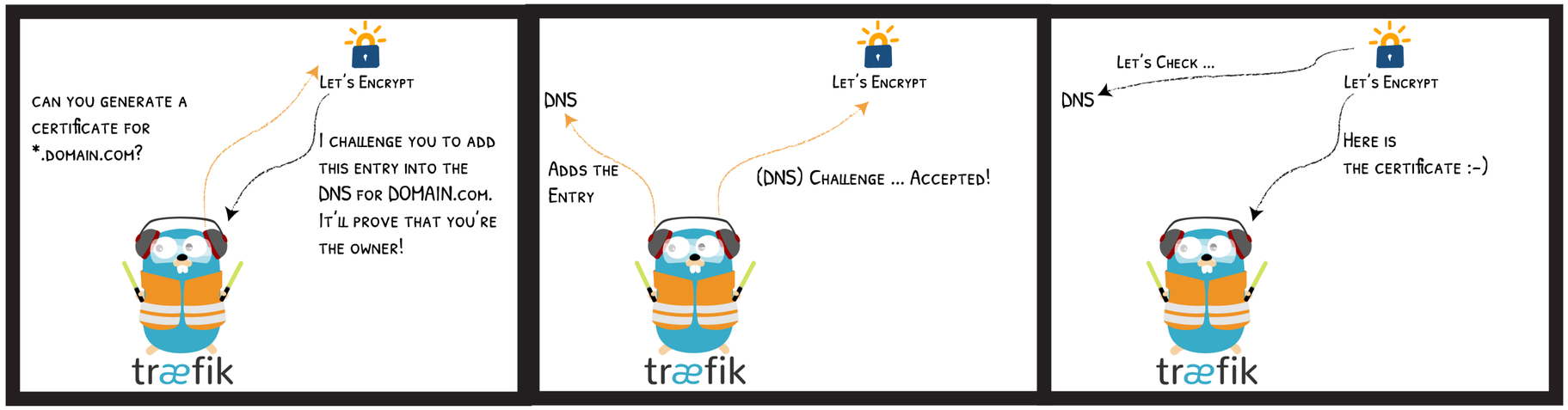


Traefik And Let’s Encrypt

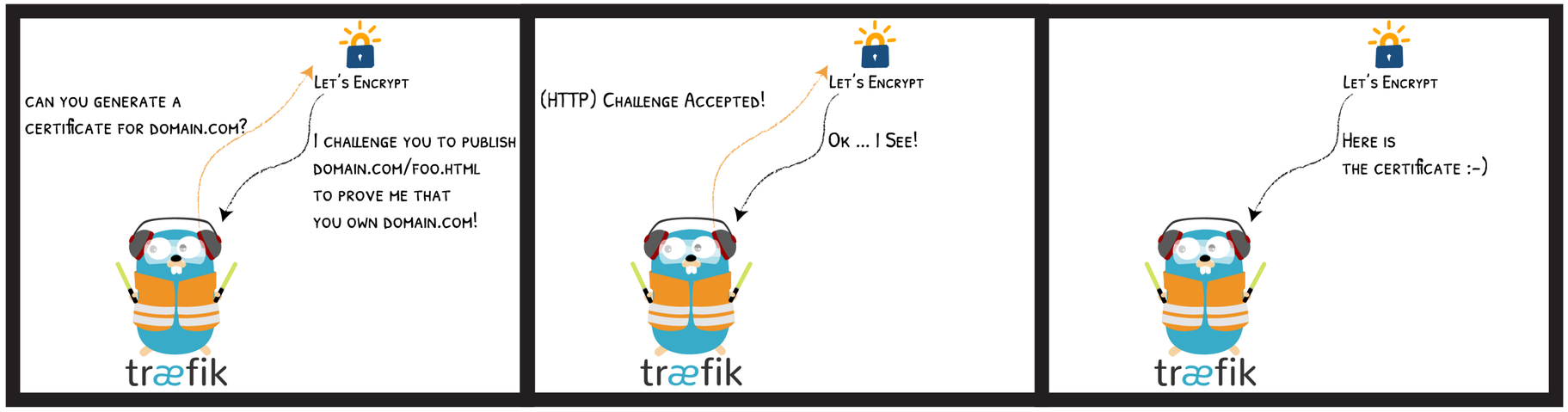
HTTPS & Let's Encrypt



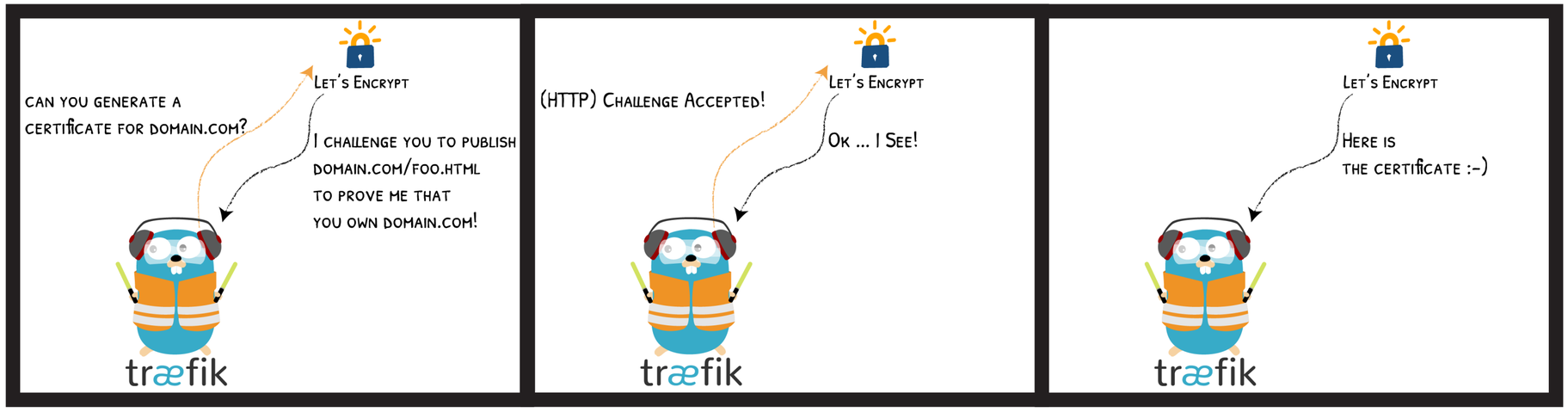
Let's Encrypt DNS Challenge



Let's Encrypt HTTP Challenge

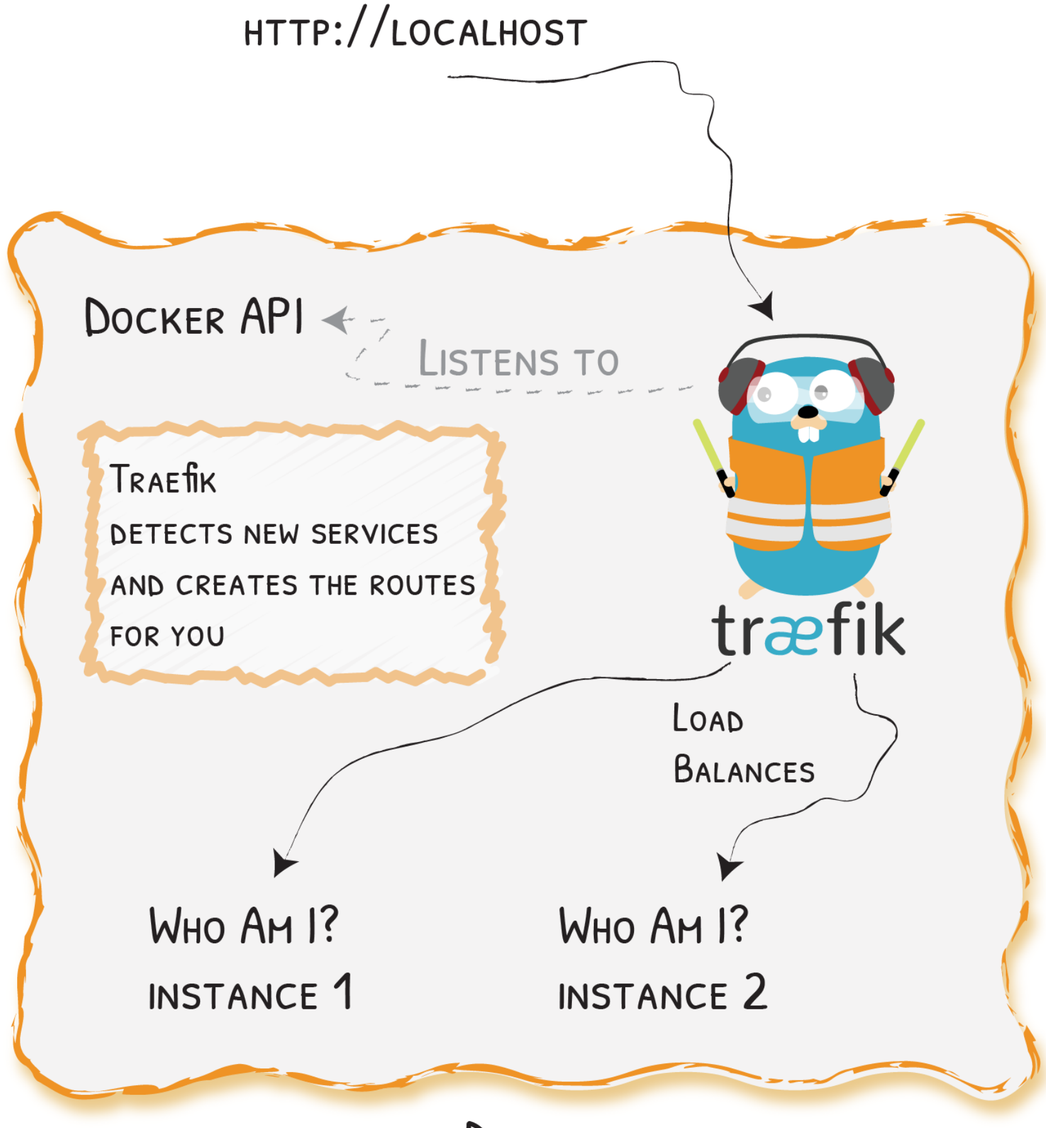


Let's Encrypt TLS Challenge



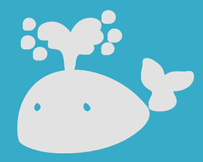
Show Me The Configuration!

Simple Example With



TRAEFIK IS THE ONLY ACCESSIBLE CONTAINER FROM THE OUTSIDE.

OTHER SERVICES ARE DEPLOYED ON DEMAND AND AVAILABLE THROUGHT TRAEFIK.

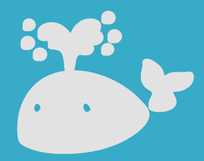
With 

- With Docker Compose:

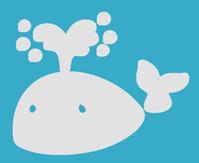
```yaml
version: '3'

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"
```

With 🐳: Websockets

```
# https://webterminal.mycompany.org -> http://webterminal/
webterminal:
    image: tsl0922/ttyd
    labels:
      - "traefik.http.routers.devbox.rule=Host(`webterminal.mycompany.org`)"
```

With File Configuration

Canaray Releases

```
http:
  services:
    canary:
      weighted:
        services:
        - name: appv1
          weight: 3 # 75%
        - name: appv2
          weight: 1 #25%
  appv1:
    loadBalancer:
      servers:
      - url: "http://private-ip-server-1/"
  appv2:
    loadBalancer:
      servers:
      - url: "http://private-ip-server-2/"
```

Traefik With ⎈

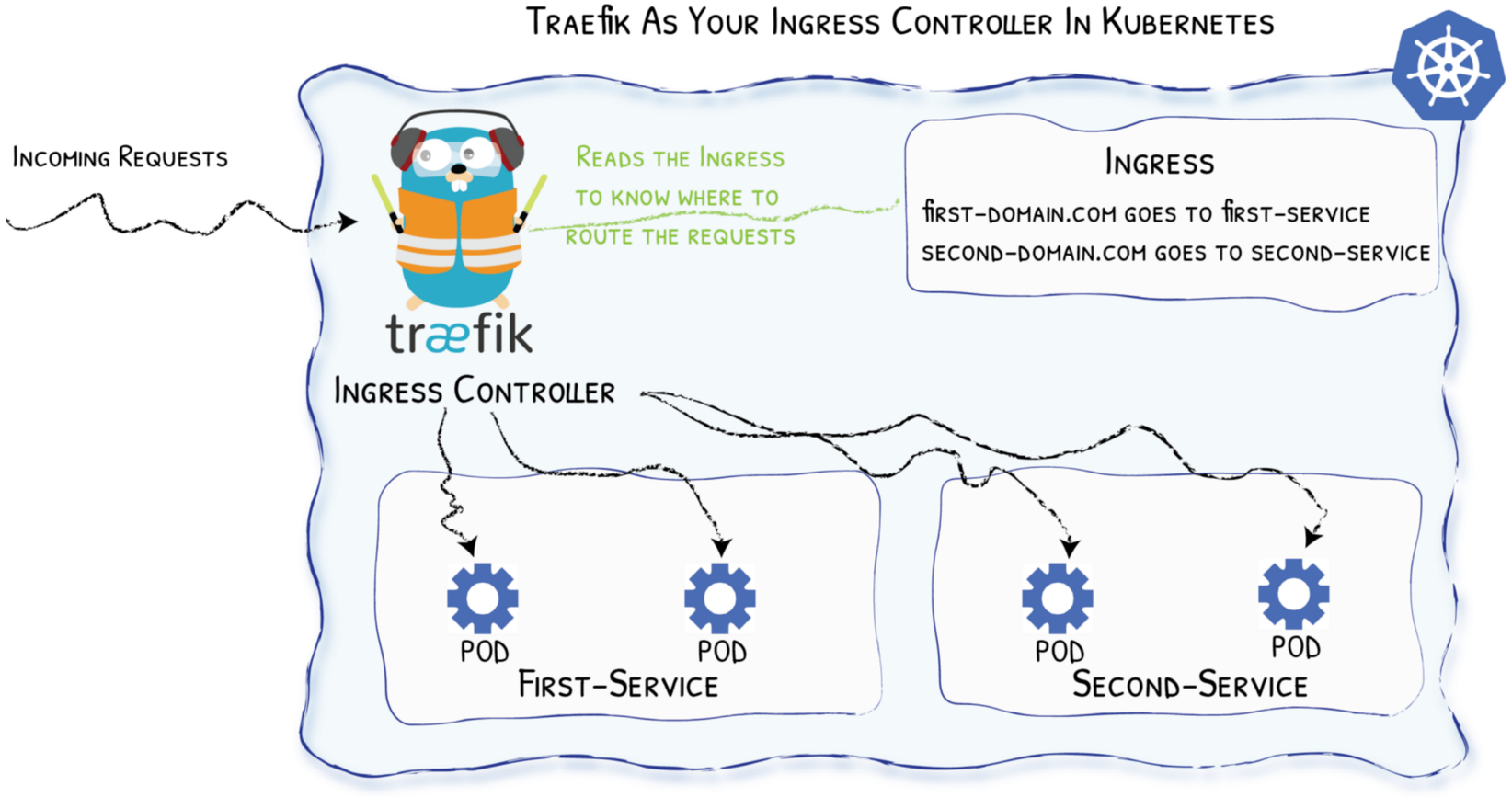


Diagram from https://medium.com/@geraldcroes

Ingress Example With ☸

```yaml
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
```

But···

Annotations

General annotations

The following general annotations are applicable on the Ingress object:

Annotation	Description
`traefik.ingress.kubernetes.io/app-root: "/index.html"`	Redirects all requests for `/` to the defined path. (1)
`traefik.ingress.kubernetes.io/error-pages: <YML>`	See custom error pages section. (2)
`traefik.ingress.kubernetes.io/frontend-entry-points: http,https`	Override the default frontend endpoints.
`traefik.ingress.kubernetes.io/pass-client-tls-cert: <YML>`	Forward the client certificate following the configuration in YAML. (3)
`traefik.ingress.kubernetes.io/pass-tls-cert: "true"`	Override the default frontend PassTLSCert value. Default: `false`.(DEPRECATED)
`traefik.ingress.kubernetes.io/preserve-host: "true"`	Forward client `Host` header to the backend.
`traefik.ingress.kubernetes.io/priority: "3"`	Override the default frontend rule priority.
`traefik.ingress.kubernetes.io/rate-limit: <YML>`	See rate limiting section. (4)
`traefik.ingress.kubernetes.io/redirect-entry-point: https`	Enables Redirect to another entryPoint for that frontend (e.g. HTTPS).
`traefik.ingress.kubernetes.io/redirect-permanent: "true"`	Return 301 instead of 302.
`traefik.ingress.kubernetes.io/redirect-regex: ^http://localhost/(.*)`	Redirect to another URL for that frontend. Must be set with `traefik.ingress.kubernetes.io/redirect-replacement`.
`traefik.ingress.kubernetes.io/redirect-replacement: http://mydomain/$1`	Redirect to another URL for that frontend. Must be set with `traefik.ingress.kubernetes.io/redirect-regex`.
`traefik.ingress.kubernetes.io/request-modifier: AddPrefix: /users`	Adds a request modifier to the backend request.
`traefik.ingress.kubernetes.io/rewrite-target: /users`	Replaces each matched Ingress path with the specified one, and adds the old path to the `X-Replaced-Path` header.

Annotations

You can add these Kubernetes annotations to specific Ingress objects to customize their behavior.

Tip

Annotation keys and values can only be strings. Other types, such as boolean or numeric values must be quoted, i.e. `"true"`, `"false"`, `"100"`.

Note

The annotation prefix can be changed using the `--annotations-prefix` command line argument, but the default is `nginx.ingress.kubernetes.io`, as described in the table below.

Name	type
nginx.ingress.kubernetes.io/app-root	string
nginx.ingress.kubernetes.io/affinity	cookie
nginx.ingress.kubernetes.io/affinity-mode	"balanced" or "persistent"
nginx.ingress.kubernetes.io/auth-realm	string
nginx.ingress.kubernetes.io/auth-secret	string
nginx.ingress.kubernetes.io/auth-secret-type	string
nginx.ingress.kubernetes.io/auth-type	basic or digest
nginx.ingress.kubernetes.io/auth-tls-secret	string
nginx.ingress.kubernetes.io/auth-tls-verify-depth	number
nginx.ingress.kubernetes.io/auth-tls-verify-client	string
nginx.ingress.kubernetes.io/auth-tls-error-page	string
nginx.ingress.kubernetes.io/auth-tls-pass-certificate-to-upstream	"true" or "false"
nginx.ingress.kubernetes.io/auth-url	string
nginx.ingress.kubernetes.io/auth-cache-key	string

☸ CRD - Custom Resources Definition

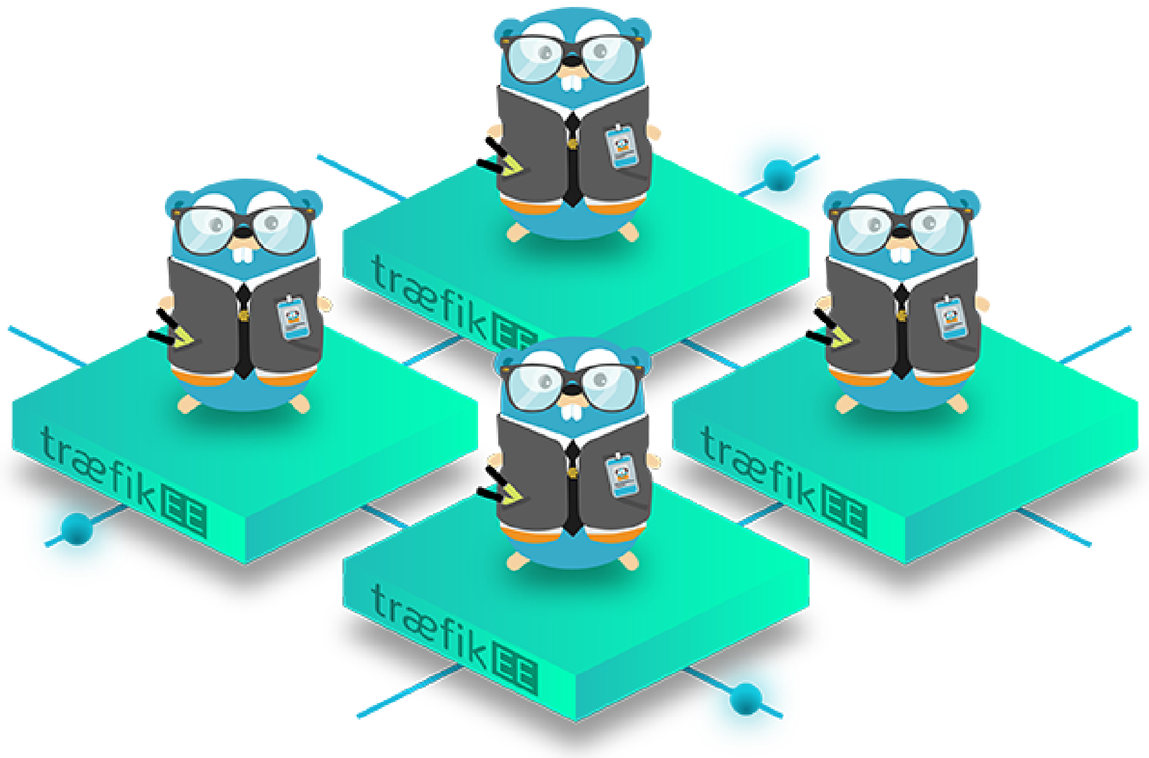
```yaml
# File "webapp.yaml"
apiVersion: traefik.containo.us/v1alpha1
kind: IngressRoute
metadata:
  name: simpleingressroute
spec:
  entryPoints:
    - web
  routes:
  - match: Host(`localhost`) && PathPrefix(`/whoami`)
    kind: Rule
    services:
    - name: webapp
      port: 80
```

```
$ 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 Also Comes In Herd



HIGH AVAILABILITY

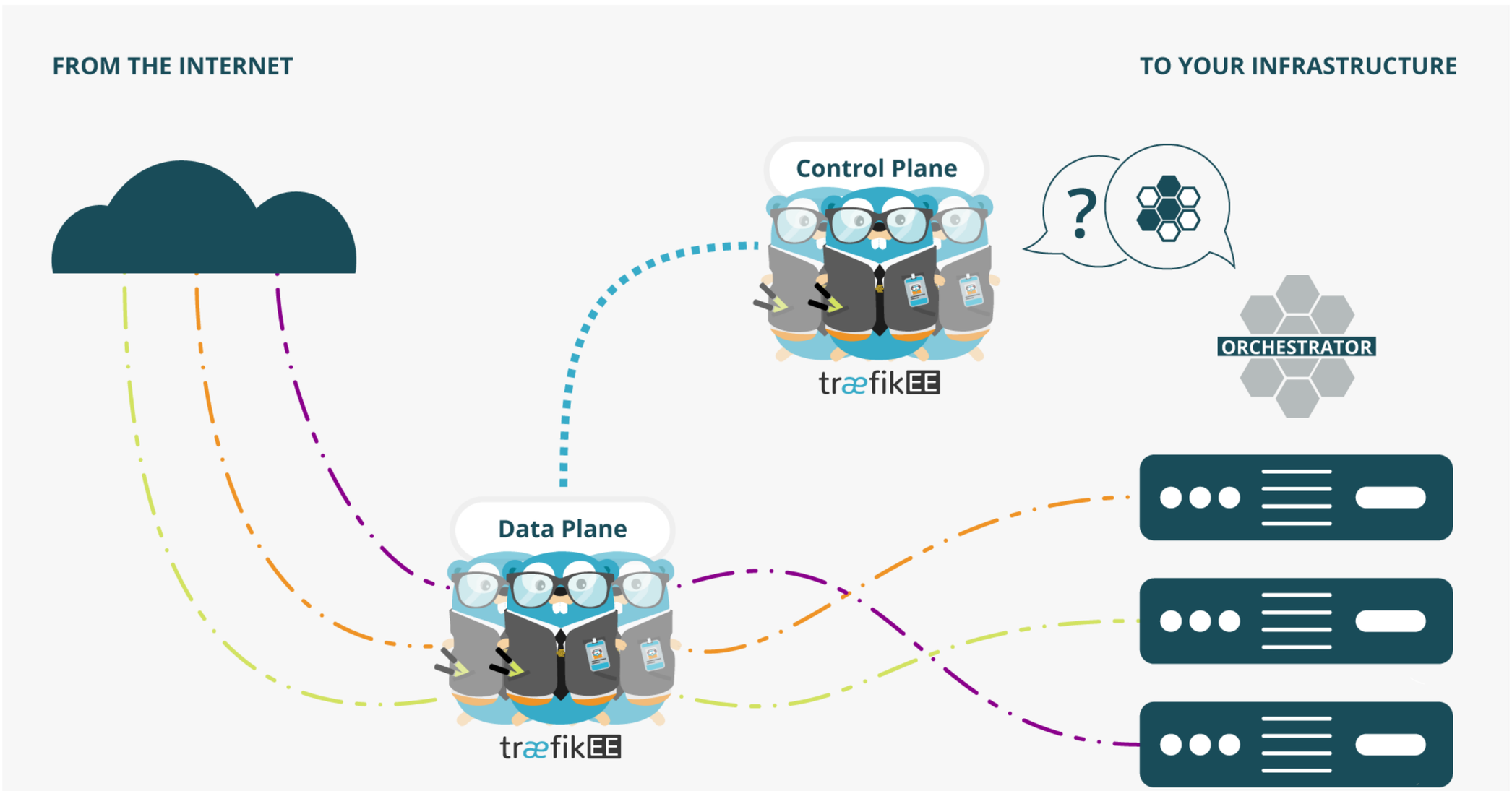
træfik ENTERPRISE EDITION

SECURITY

træfik ENTERPRISE EDITION

SCALABILITY

træfik ENTERPRISE EDITION


FROM THE INTERNET
TO YOUR INFRASTRUCTURE
Control Plane
træfikEE
?
ORCHESTRATOR
Data Plane
træfikEE

As Simple As Traefik

- Install it:

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

- Configure it:

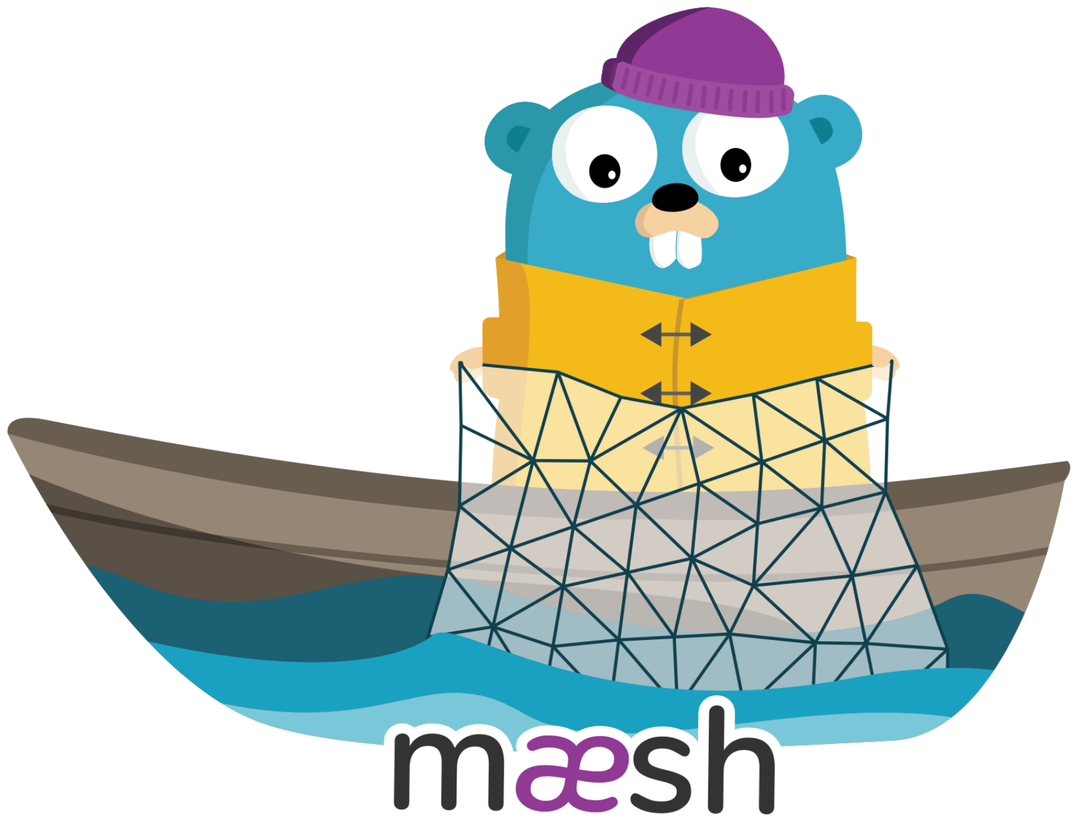
```
# Routing Configuration, same as Traefik's
traefikeectl deploy \
  --defaultentrypoints=http,https \
  --entryPoints='Name:http Address::80' \
  --entryPoints='Name:https Address::443 TLS' \
  --logLevel=INFO \
  --kubernetes
```

Free Trial

https://containo.us/traefikee

East / West Traefik

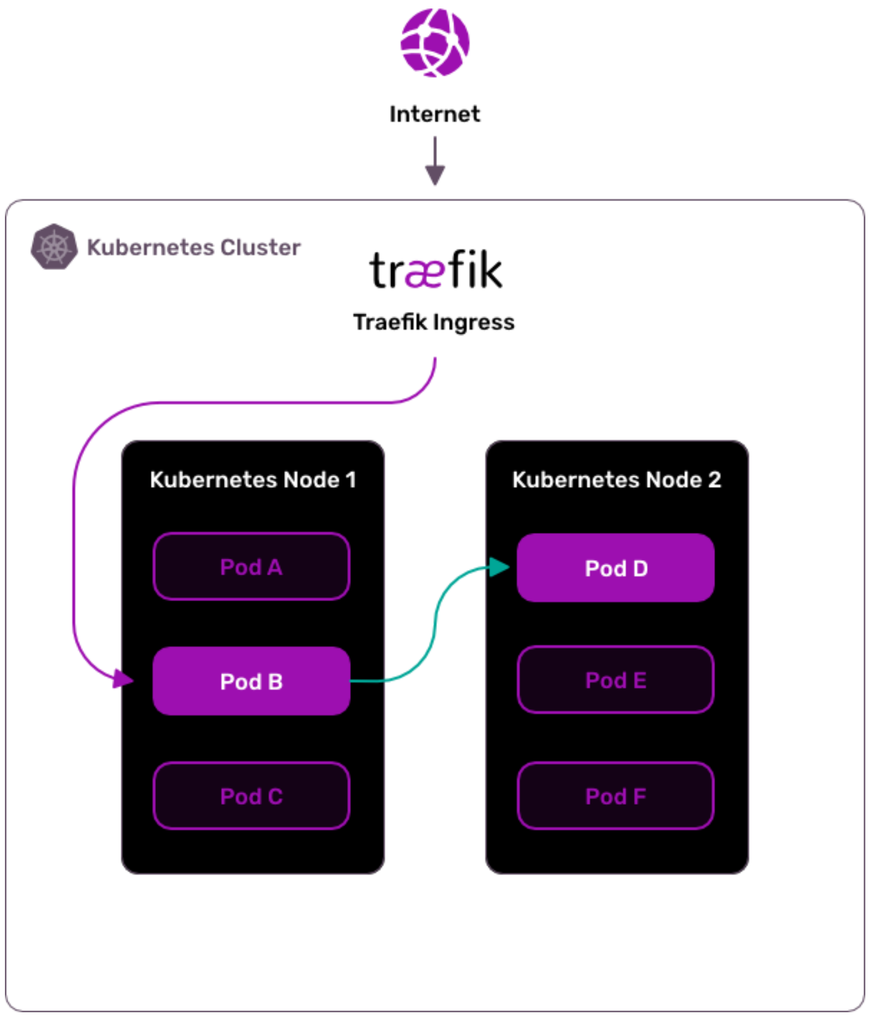
Say Hello To Maesh

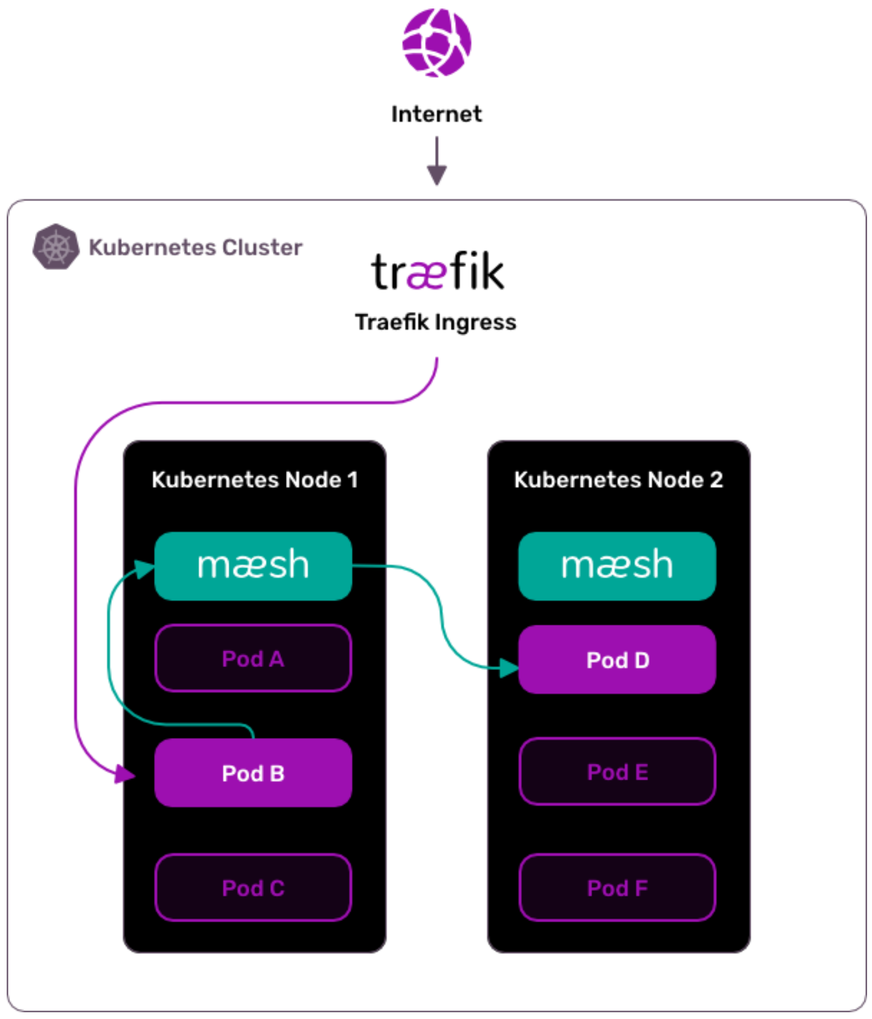


What Is Maesh?

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
```

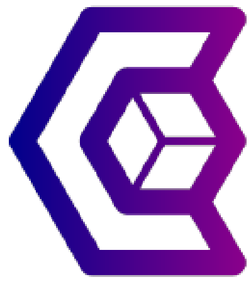
A Closer Look To SMI Objects

```
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!
træfik

We Are Hiring!



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

Thank You!

- @michaelmatur
- mmatur

- Slides (HTML): https://containous.github.io/slides/poss-2019
- Slides (PDF): https://containous.github.io/slides/poss-2019/slides.pdf
- Source on mmathttps://github.com/containous/slides/tree/poss-2019ur