

Harness Your Routing Super-Hero Power In Kubernetes With Traefik, Maesh And Konvoy



traefik



How To Access These Slides?



- Slides (HTML): <https://containous.github.io/slides/d2iq-virtual-event>
- Slides (PDF): <https://containous.github.io/slides/d2iq-virtual-event/slides.pdf>
- Source on : <https://github.com/containous/slides/tree/d2iq-virtual-event>

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

- Damien DUPORTAL:
 - Træfik's Developer  Advocate @ Containous
 -  @DamienDuportal
 -  dduortal





<https://containo.us>

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

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

Why Konvoy?

D2IQ

- Formerly known as Mesosphere
- "Day-Two-I-Q"
- A smarter approach to "Day 2 Operations"

Day 2 Operations

"Day 2" refers to the phase of the development lifecycle that follows initial deployment where the real application demands exist.

KSphere

Embrace Kubernetes when:

- Beginning your journey 🐥
- Preparing for Day 2 🦅

KSphere Offer

- Technical Solutions:
 - Konvoy
 - MKE (Mesosphere Kubernetes Engine)
- Services:
 - Professionnal Services
 - Training
 - Support

What Is Konvoy?

A packaged  Kubernetes  with integrated operational services .

Why Using Konvoy?

- Gain Flexibility Across Any Infrastructure
- Manage Operations With Ease
- Ensure Rapid Technology Adoption and Scale
- Harness Premiere Domain Expertise

Konvoy

Konvoy Concepts

- Standalone **Native** Distribution of Kubernetes
- "One button push" User Experience
- Packaged with a set of services for **Operations**

Konvoy Architecture

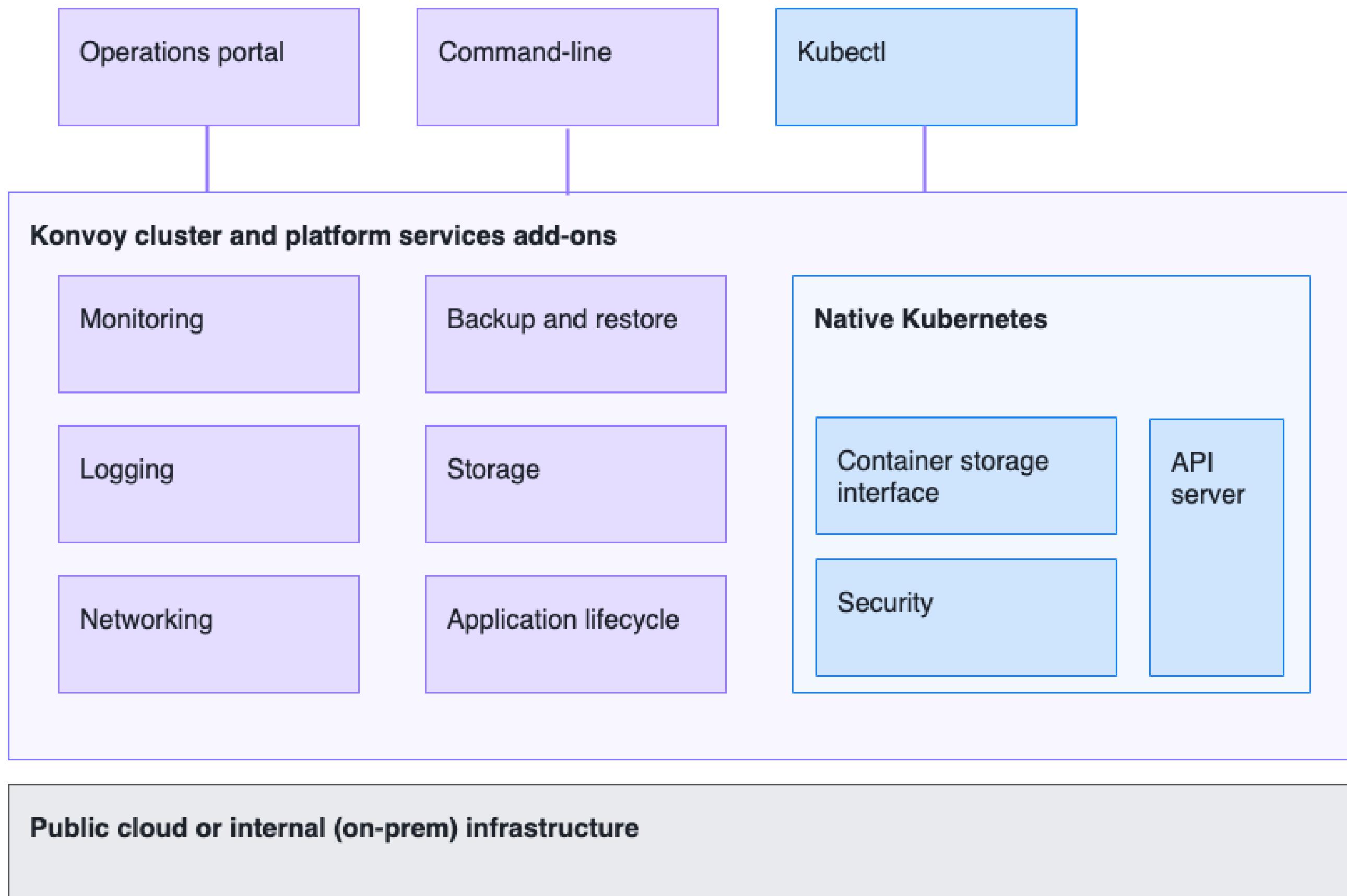


Diagram from [Konvoy Documentation](#)

Quick Start

Install a Konvoy Cluster in AWS EC2:

- Prepare installation:

```
$ konvoy init --provisioner=aws
```

- Run installation:

```
$ konvoy init --provisioner=aws
```

- Use it:

```
$ konvoy apply kubeconfig && kubectl cluster-info  
  
Kubernetes master is running at (...)  
KubeDNS is running at (...)  
kubernetes-dashboard is running at (...)
```

Konvoy Operations

- Operations Portal
- Network: CoreDNS, Calico, MetalLB, Traefik
- Security: Identity Management, SSO, TLS
- Logging: Fluentbit, Elasticsearch, Kibana
- Monitoring and Metrics: Prometheus, Grafana
- Back up and restore: Velero

Operations Portal

The screenshot displays the Konvoy Cluster Operations Portal interface. At the top left is the cluster name "Konvoy Cluster". Below it is a section for "General Cluster Information" showing a Kubernetes icon, status "Active", and version "v1.15.3". There are "View Docs" and "Dashboard" buttons. To the right are three circular progress charts: "CPU Requests" at 74% (16 Cores), "Memory Requests" at 36% (60.936 GiB), and "Ephemeral Storage Requests" at 0% (287.960 GiB). The main area is titled "Manage" and shows sections for "Monitoring", "Networking", and "Logging". Under "Monitoring", there are cards for "Grafana" (version 6.1.6) and "Prometheus" (version 2.9.1), both marked as "Enabled". Under "Logging", there is a card for "Kibana" (version 6.7.0) also marked as "Enabled". Each card includes "View Docs" and "Dashboard" buttons.

Konvoy Cluster

General Cluster Information

Kubernetes
Status: Active
Version: v1.15.3

[View Docs](#) [Dashboard](#)

CPU Requests
74%
16 Cores

Memory Requests
36%
60.936 GiB

Ephemeral Storage Requests
0%
287.960 GiB

Manage

All Monitoring Networking Logging

Grafana
Monitoring
6.1.6
[View Docs](#) [Dashboard](#)

Kibana
Logging
6.7.0
[View Docs](#) [Dashboard](#)

Prometheus
Monitoring
2.9.1
[View Docs](#) [Dashboard](#)

Operation Portal Components

Manage

All Monitoring Networking Logging

The screenshot shows a grid of six components, each with a status indicator (blue checkmark) and a purple 'Dashboard' button.

| Category | Component | Version | Status |
|------------|--------------------------|---------|---------|
| Monitoring | Grafana | 6.1.6 | Enabled |
| Logging | Kibana | 6.7.0 | Enabled |
| Monitoring | Prometheus | 2.9.1 | Enabled |
| Monitoring | Prometheus Alert Manager | 0.16.2 | Enabled |
| Networking | Traefik | 1.68.4 | Enabled |

View Docs Dashboard

Manage

All Monitoring Networking Logging



Monitoring
Grafana
6.1.6

[View Docs](#) [Dashboard](#)



Logging
Kibana
6.7.0

[View Docs](#) [Dashboard](#)



Monitoring
Prometheus
2.9.1

[View Docs](#) [Dashboard](#)



Monitoring
Prometheus Alert Manager
0.16.2

[View Docs](#) [Dashboard](#)



Networking
Traefik
1.68.4

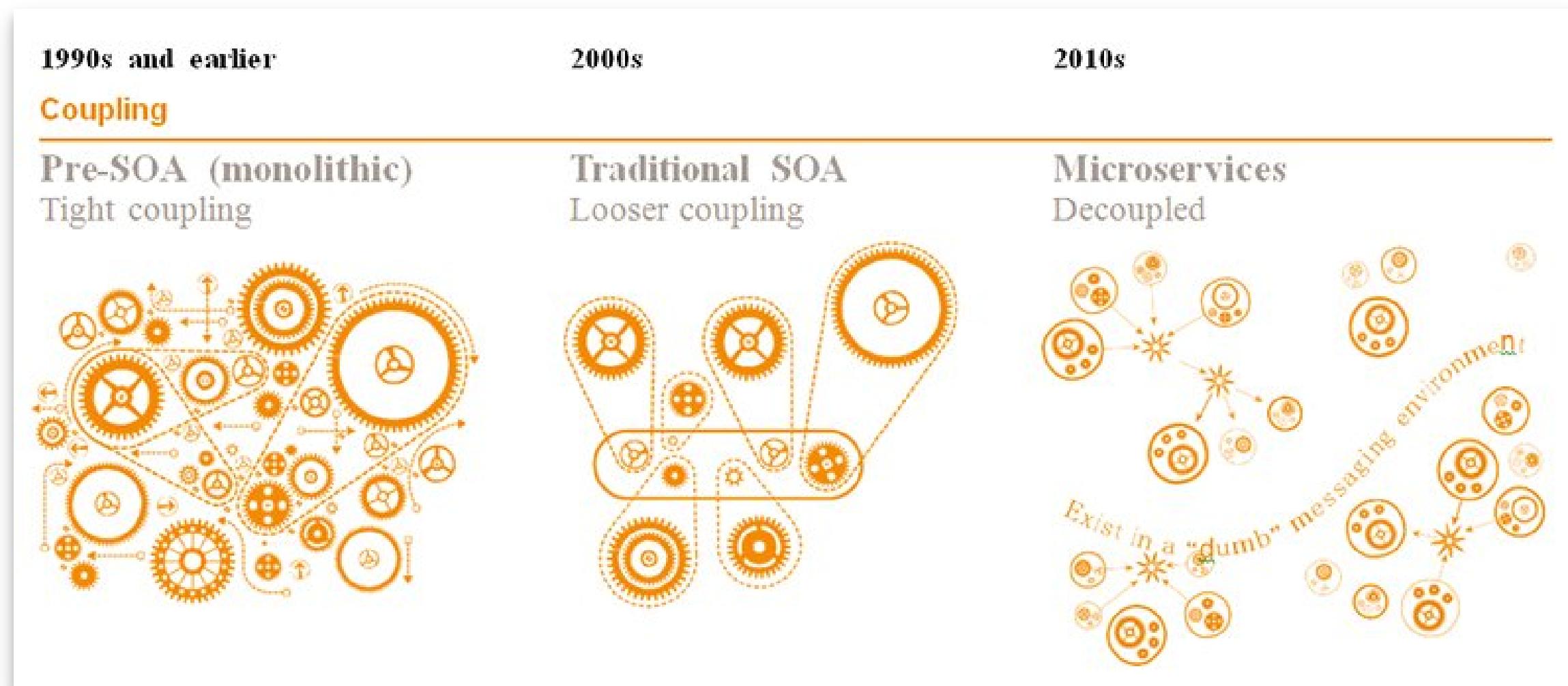
[View Docs](#) [Dashboard](#)

Why Traefik?



Why, Mr Anderson?

Evolution Of Software Design



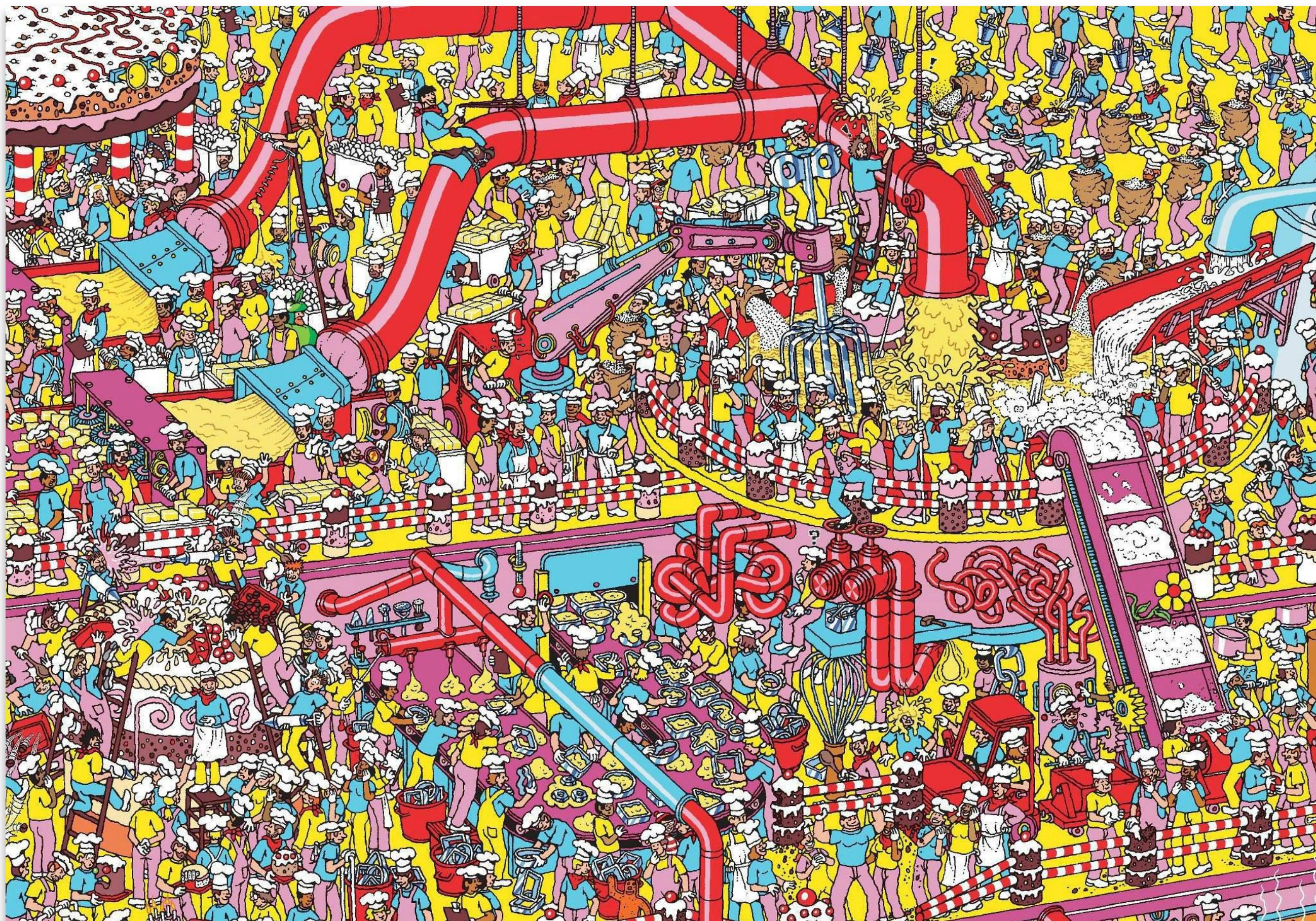
The Premise Of Microservices...



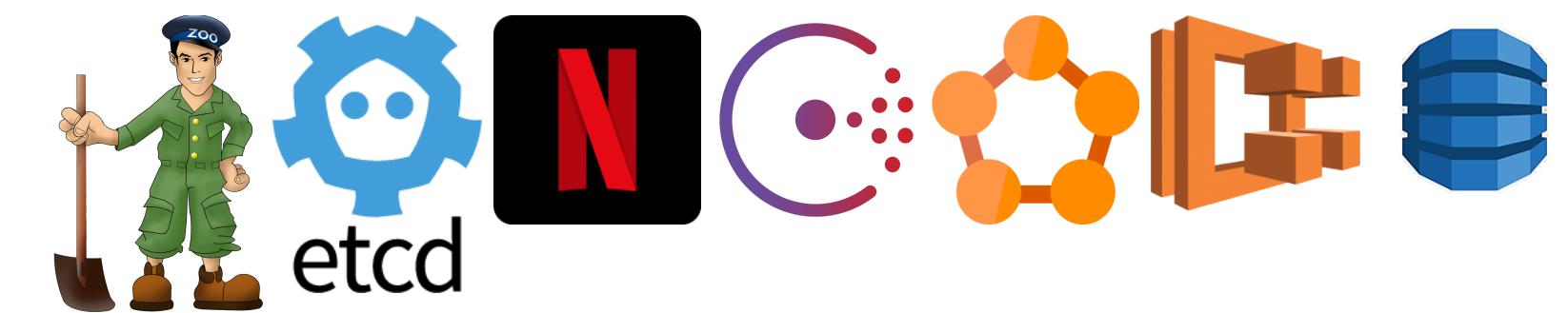
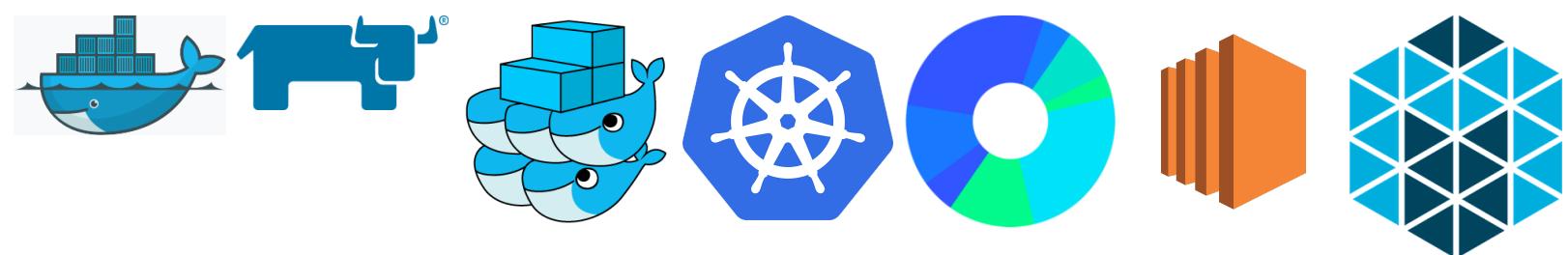
...And What Happens



Where's My Service?



Tools Of The Trade



MacBook Pro



The image shows a MacBook Pro with its screen open, displaying a code editor (Atom) running on a Linux terminal window. The terminal window is showing a YAML configuration file for a Flink job manager. The code editor has several tabs open, including 'flink-scaffold.yaml', 'Dockerfile', 'flink-scaffold.aml', 'bootstrap', 'server', 'flink-s...', 'docker...', 'flink-s...', 'jobma...', 'untitled', 'Git', and 'GitHub (preview)'. The GitHub interface shows an 'Unstaged Changes' section with 'No changes' and a 'Staged Changes' section with 'No changes'. A commit message is visible at the bottom: 'Commit to moda-deploy' with a timestamp of 72 minutes ago. The commit message itself is: 'bump the resources temporarily to see if we ca...'. Below the commit message is a list of recent commits:

- bump the resources temporarily to see if we can... 18h
- set 1 replicate for now 18h
- Revert "do we need to recreate?" 19h
- do we need to recreate? 19h
- fix service name 19h

At the bottom of the screen, the status bar shows 'LF' (Linux), 'UTF-8', 'YAML', 'moda-deploy', 'Fetch', and '0 files'.

```
Project
flink-scaffold
  idea
  config
    kubernetes
      default
        deployments
          flink-scaffold.yaml
        services
          jobmanager.yaml
  moda
  script
  src
  target
  classes
  generated-sources
    annotations
    .gitignore
  docker-compose.yml
  Dockerfile
  flink-scaffold.aml
  pom.xml
  readme.md

config/kubernetes/default/deployments/flink-scaffold.yaml ① 0 ▲ 0 ① 0 27:24 Prettier ✓
```

```
flink-scaffold:
  app: flink-scaffold
  late:
  metadata:
    labels:
      app: flink-scaffold
  spec:
    containers:
      - name: flink-scaffold
        image: flink-scaffold
        command:
          - /opt/flink/bin/flink
          - run
          - --jobmanager
          - jobmanager:8081
          - /jars/flink-scaffold-1.0-SNAPSHOT.jar
        name: jobmanager
        image: flink
        ports:
          - containerPort: 6123
            name: jm-rpc
          - containerPort: 6124
            name: blob
          - containerPort: 6125
            name: query
          - containerPort: 8081
            name: jm-ui
        args:
          - jobmanager
        env:
          - name: JOB_MANAGER_RPC_ADDRESS
            value: jobmanager
        resources:
          requests:
            cpu: 2
            memory: "1024M"
          limits:
            cpu: 3
            memory: "2048M"
      - name: taskmanager
        image: flink
        ports:
          - containerPort: 6121
            name: tm-data
  containerPort: 6120
```

```
1   - name: iceberg
2     image: iceberg
3     args:
4       - jobmanager
5     ports:
6       - containerPort: 6123
7       - name: rpc
8       - containerPort: 6124
9       - name: blob
10      - containerPort: 6125
11      - name: query
12      - containerPort: 8081
13      - name: ui
14      - containerPort: 6200
15      - name: ha
16      env:
17        - name: DOGSTATSD_HOST
18          valueFrom:
19            fieldRef:
20              fieldPath: spec.nodeName
21            name: JOB_MANAGER_RPC_ADDRESS
22            value: flink-jobmanager
23            resources:
24              requests:
25                - cpu: 2
26                - memory: "1024M"
27              limits:
28                - cpu: 3
29                - memory: "2048M"
```

Git
GitHub (preview)

Unstaged Changes

No changes

Staged Changes

No changes

Commit message

Commit to moda-deploy 72

bump the resources temporarily to see if we can... 18h

set 1 replicate for now 18h

Revert "do we need to recreate?" 19h

do we need to recreate? 19h

fix service name 19h

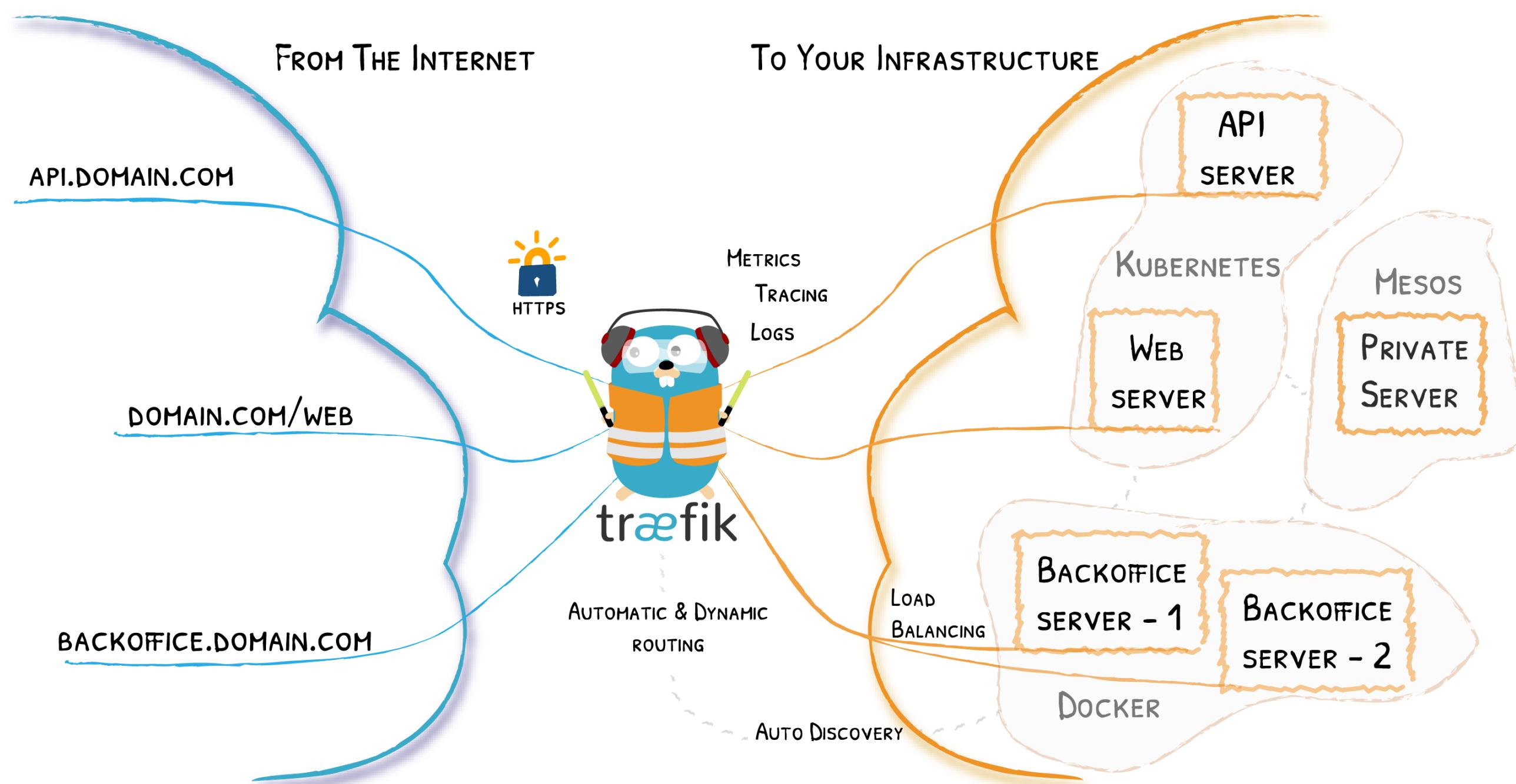
LF UTF-8 YAML moda-deploy Fetch 0 files

What If I Told You?



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

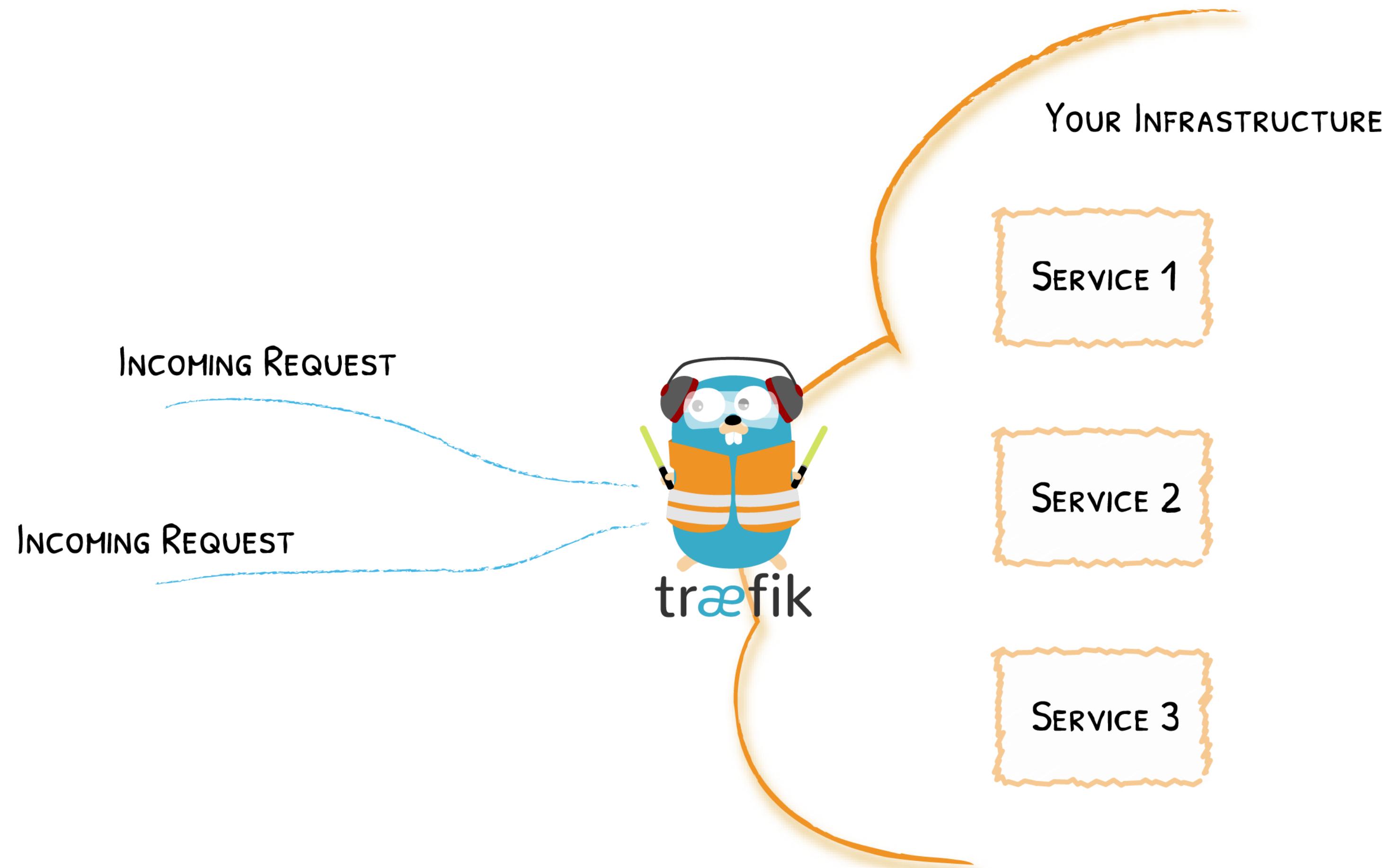
Here Comes Traefik!



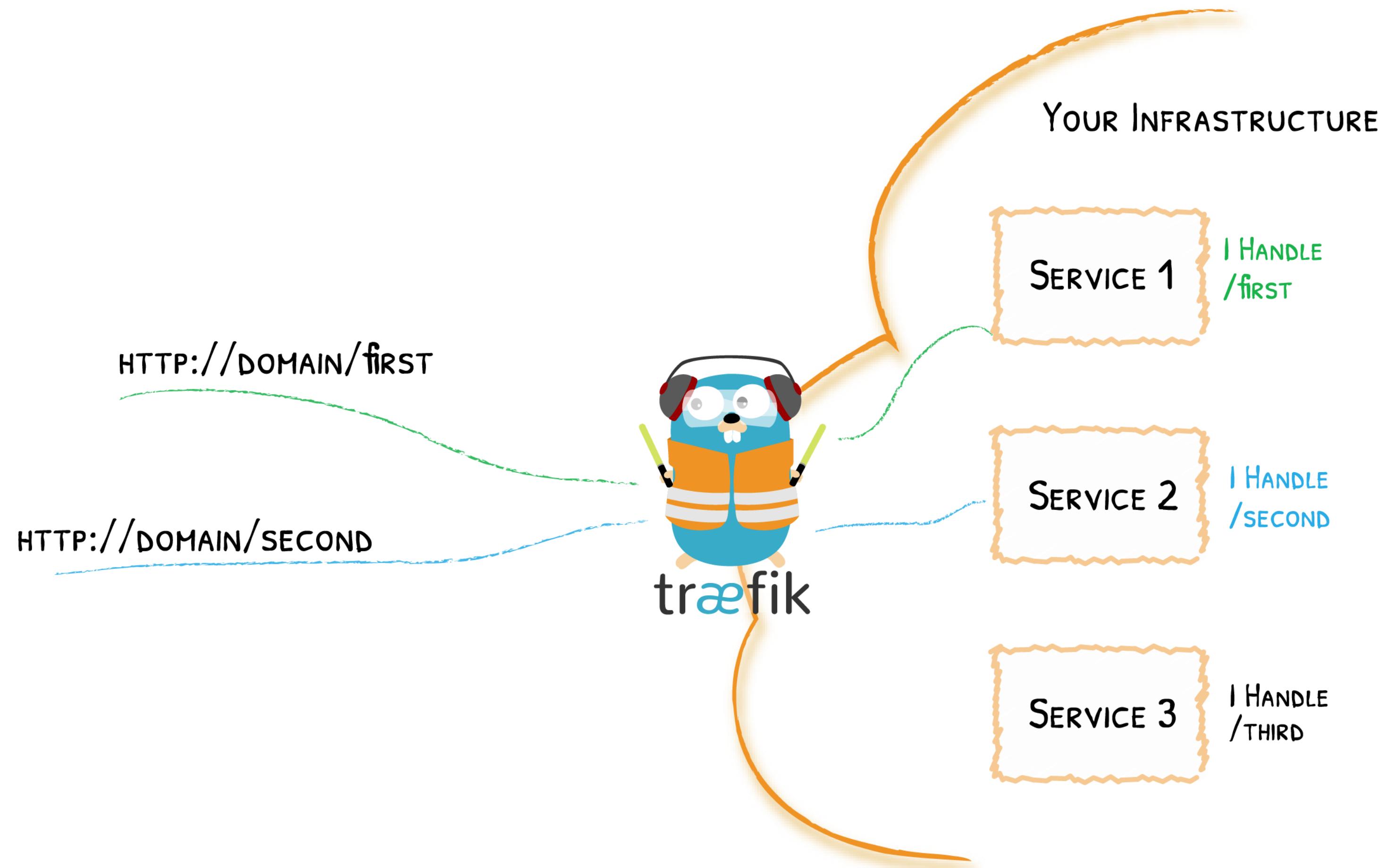
Traefik Project

-  <https://github.com/containous/traefik>
- MIT License
- Written in Go
- 24,000+ ⭐ 1B+ ↓ 400+ 
- Created in 2015, 4Y 
- Current stable branch: v2 . 0

Traefik Is An Edge Proxy



It Dynamically Discovers Services



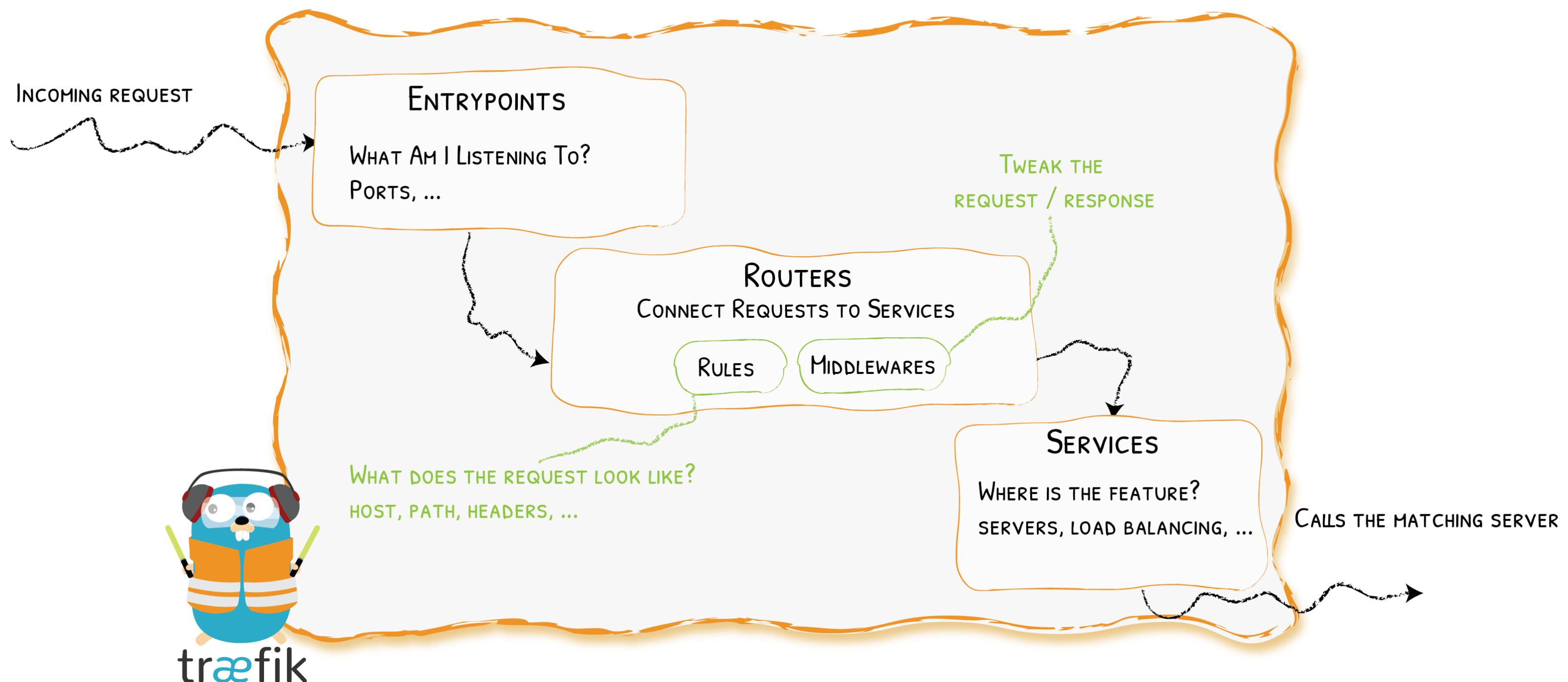
Demo Time!

Konvoy and Traefik v1

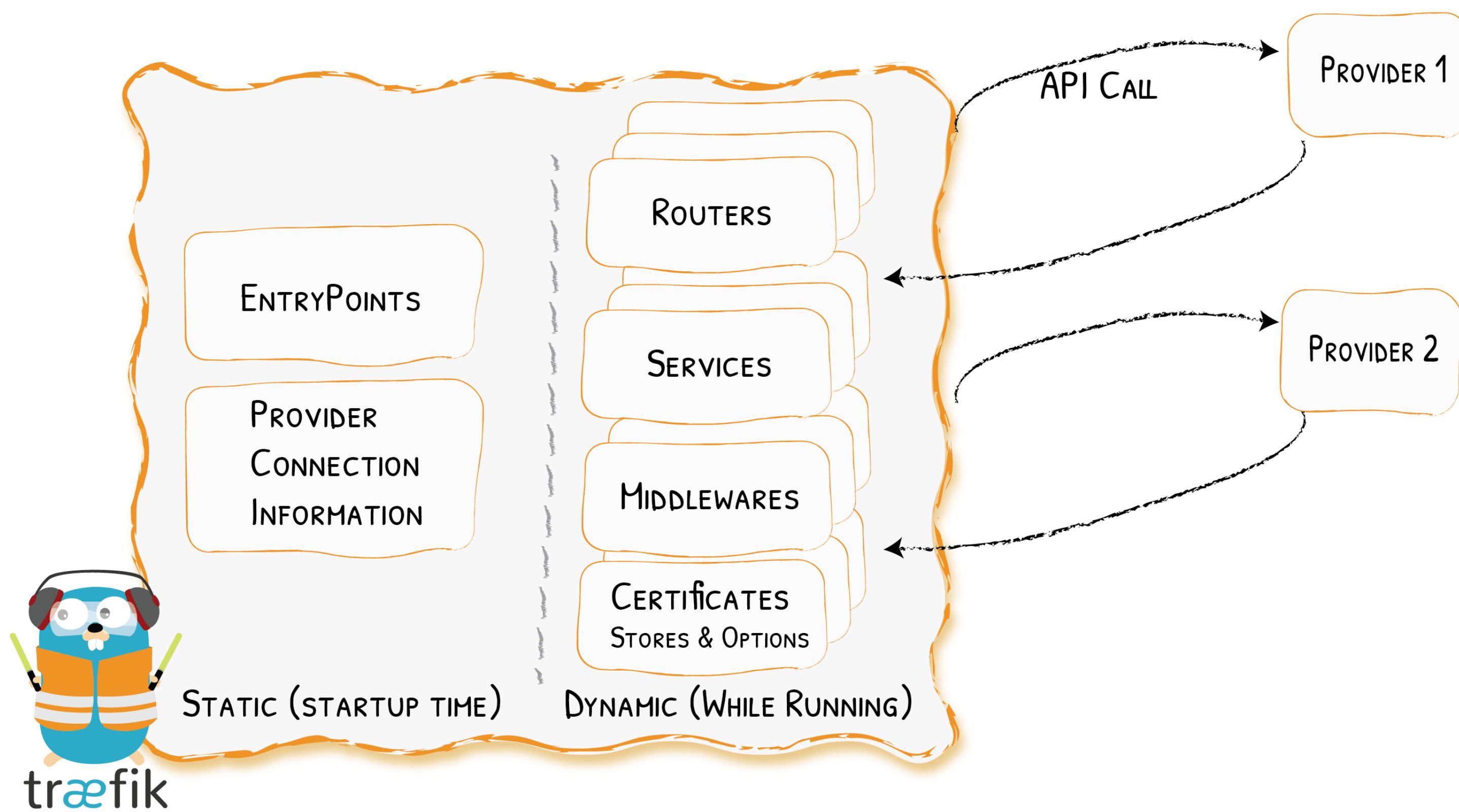
Traefik V2



Architecture At A Glance



Static & Dynamic Configuration



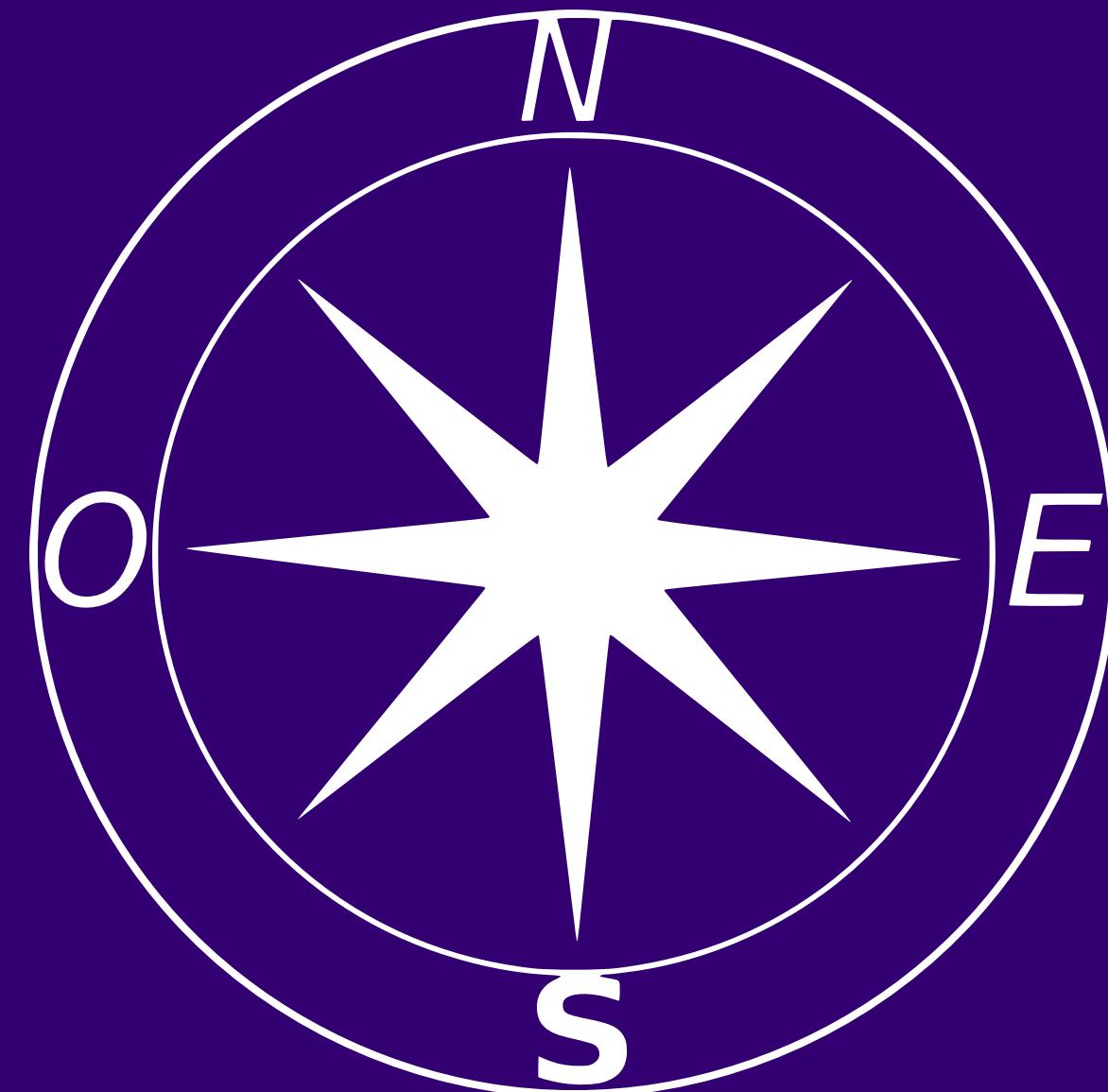
Demo Time (2)!

Traefik v2 in Konvoy

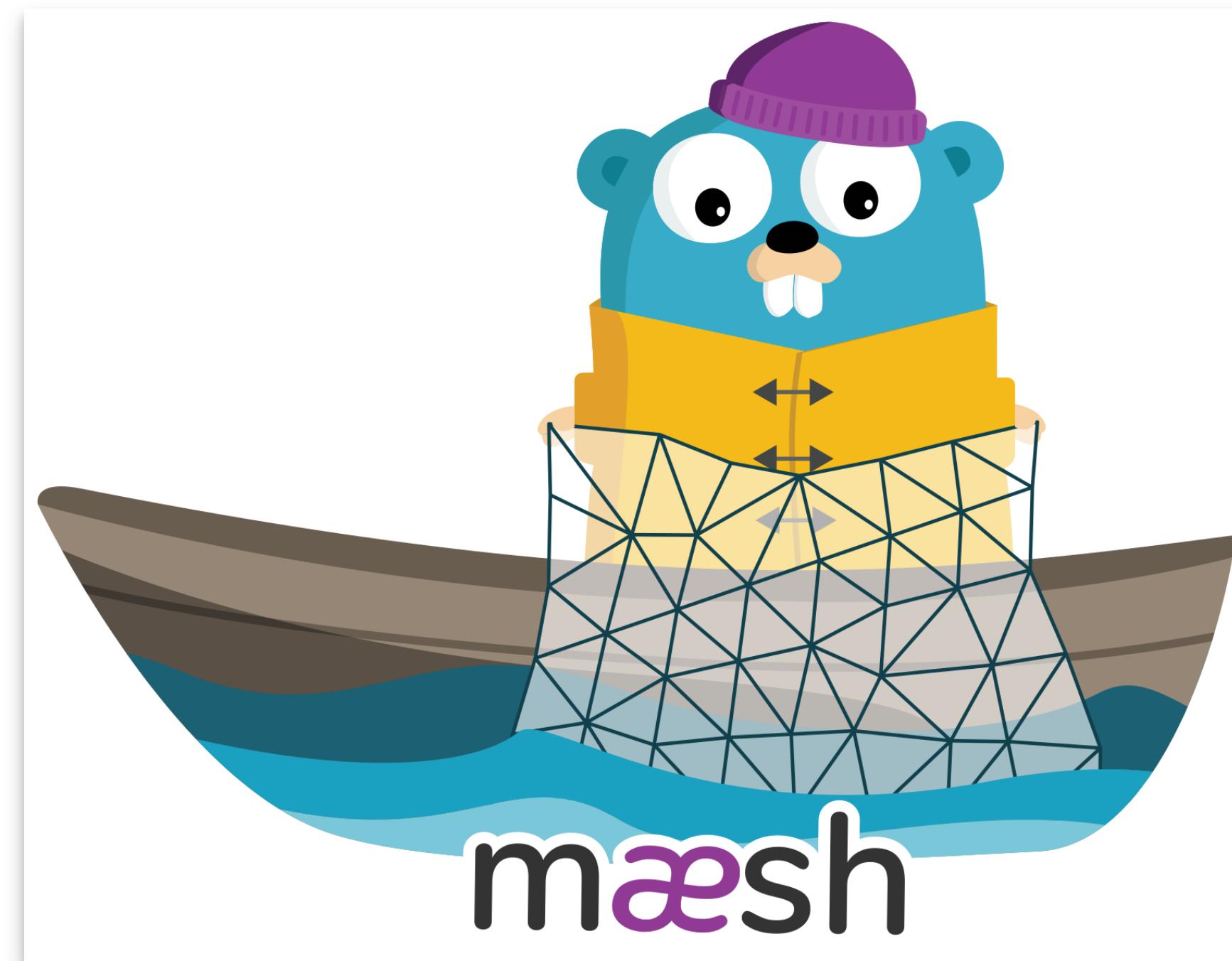
Maesh

East / West Traefik

What about routing traffic from service to services?



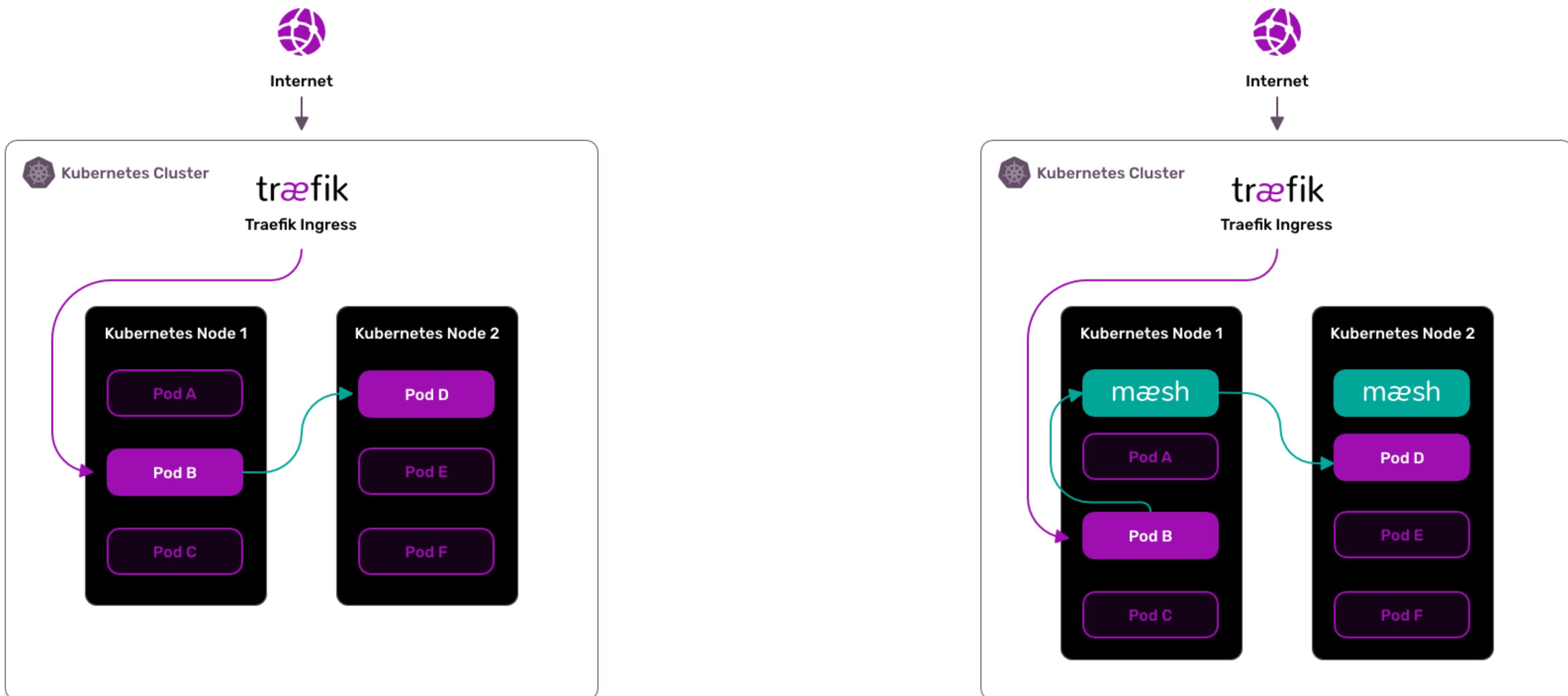
Say Hello To Maesh



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
```

Demo Time (3)!

Maesh in Konvoy

That's All Folks!

Thank You!

 @DamienDuportal

 dduportal



- Slides (HTML): <https://containous.github.io/slides/d2iq-virtual-event>
- Slides (PDF): <https://containous.github.io/slides/d2iq-virtual-event/slides.pdf>
- Source on : <https://github.com/containous/slides/tree/d2iq-virtual-event>