



Cloud Native Infrastructure @Zalando

Cloud Native Aarhus

November 6th 2018



meetup

MIKKEL LARSEN

[@mikkeloscar](https://twitter.com/mikkeloscar)

2018-11-06



WE BRING FASHION TO PEOPLE IN 17 COUNTRIES

17 markets

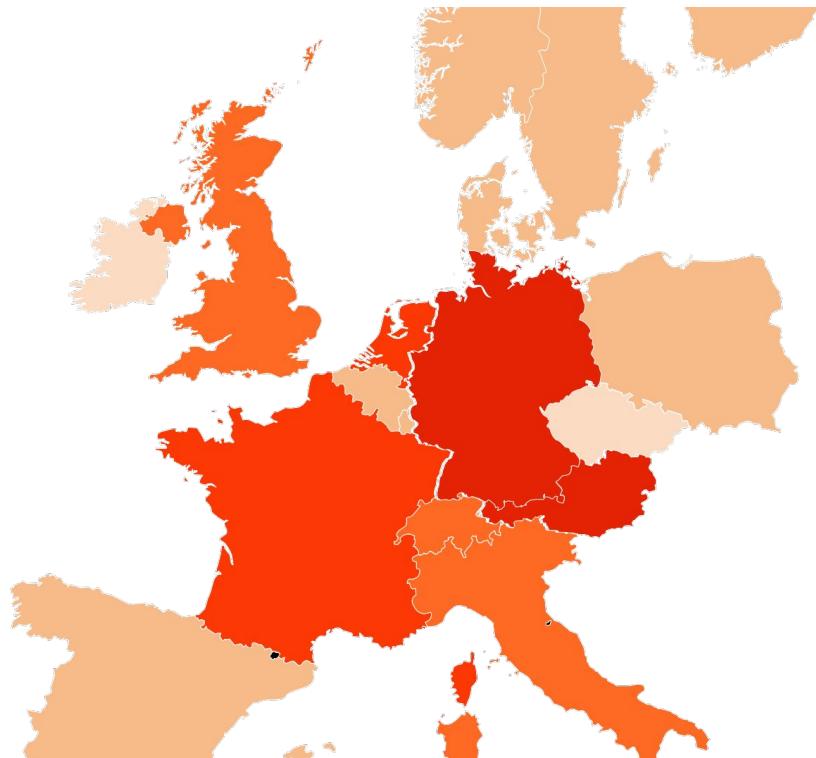
7 fulfillment centers

23 million active customers

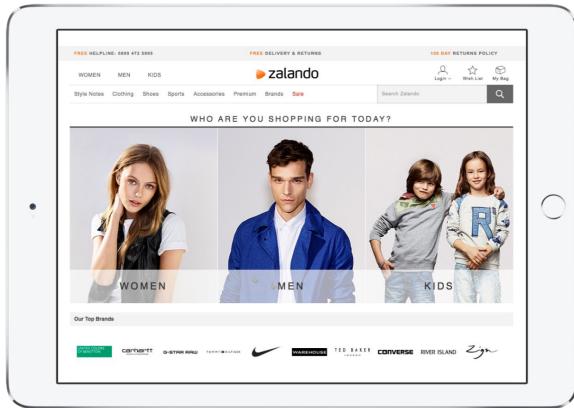
4.5 billion € net sales 2017

200 million visits per month

15,000 employees in Europe



ZALANDO TECH



~ 2.000
Employees in Tech

~ 300
Delivery teams

SCALE

373
Accounts



101
Clusters

INFRASTRUCTURE HISTORY @ ZALANDO



STUPS (toolset around AWS)

AWS accounts per team.

All instances must run the same AMI.

PowerUser access to Production.

You build it, you run EVERYTHING.



Kubernetes

Clusters per product (multiple teams).

Instances are not managed by teams.

Hands off approach.

A lot of stuff out of the box.

KUBERNETES CLUSTER SETUP



Master



Master



Config



Core OS



EC2
Instances



CloudFormation
Stacks



Worker



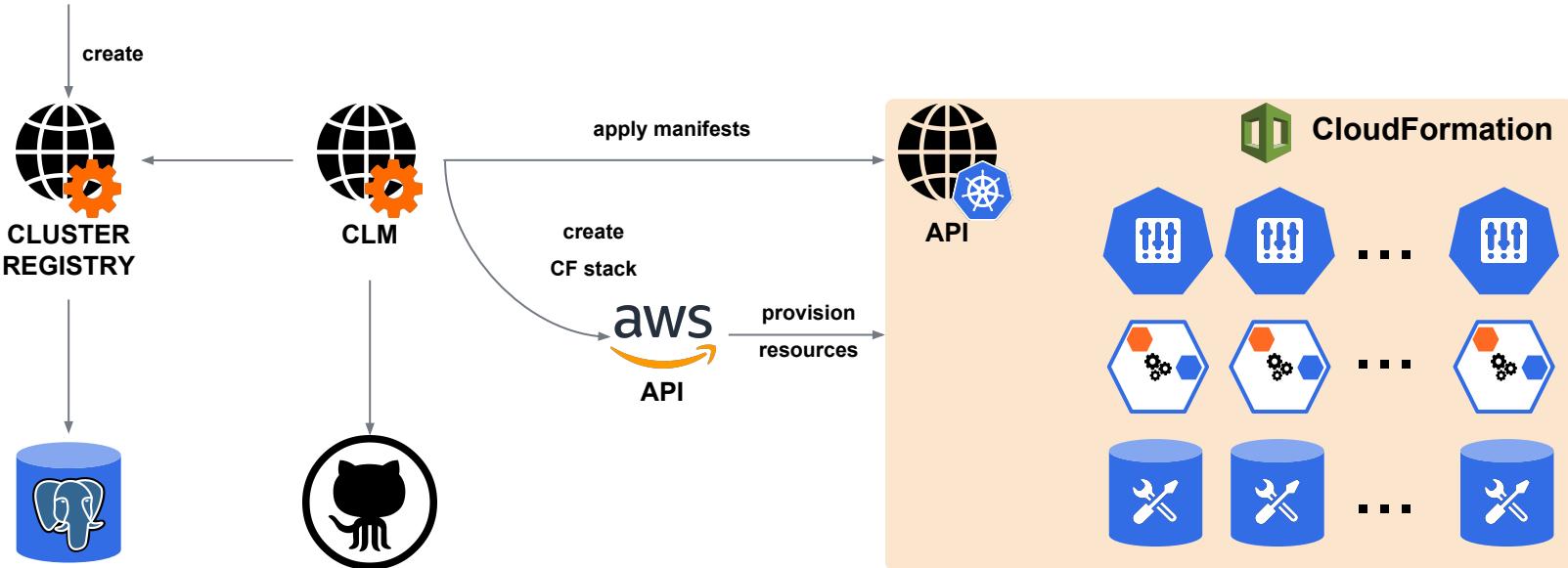
github.com/zalando-incubator/kubernetes-on-aws

CLUSTER PROVISIONING

CLUSTER LIFECYCLE MANAGER (CLM)



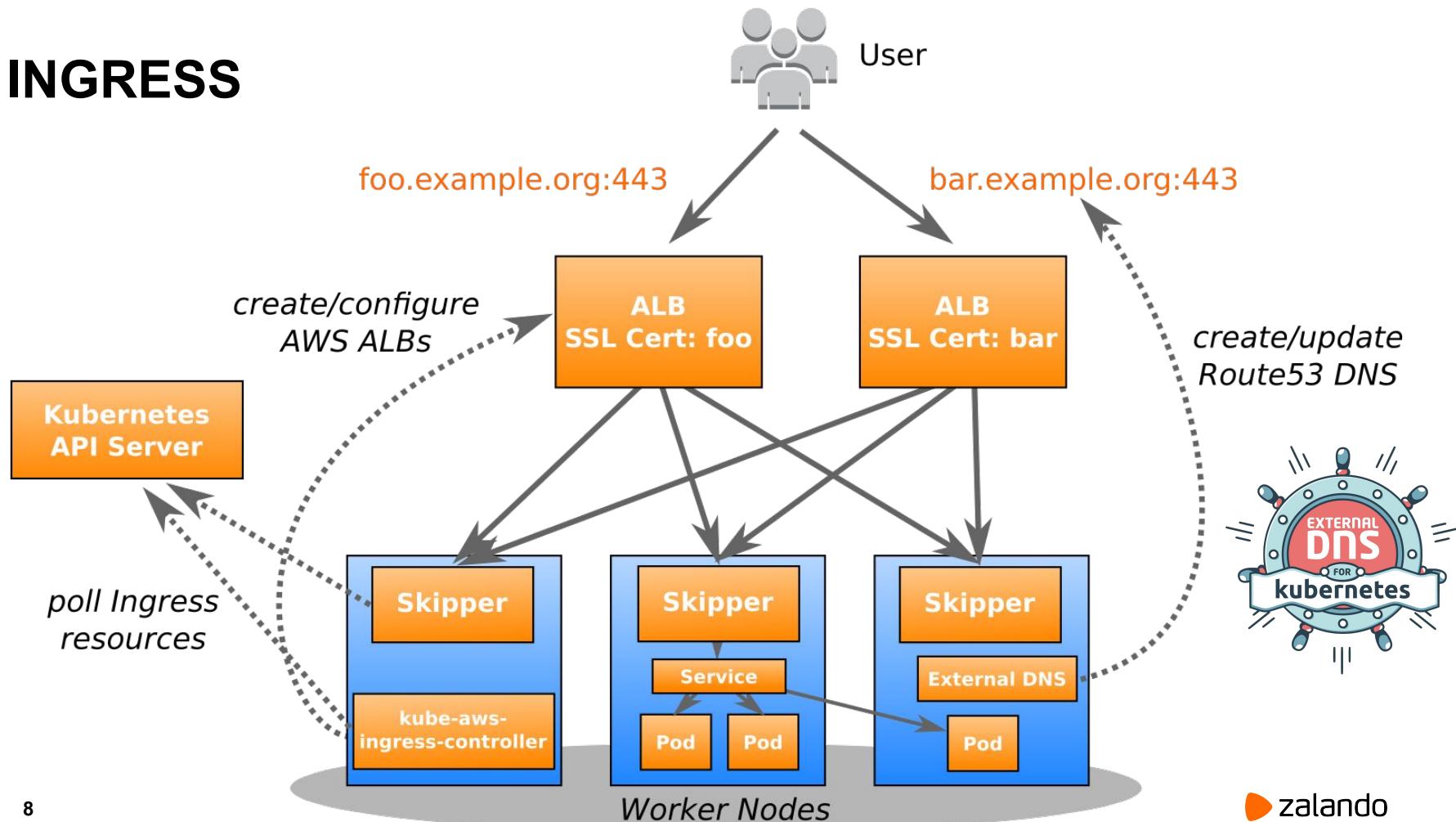
ADMIN



github.com/zalando-incubator/cluster-lifecycle-manager

github.com/zalando-incubator/kubernetes-on-aws

INGRESS



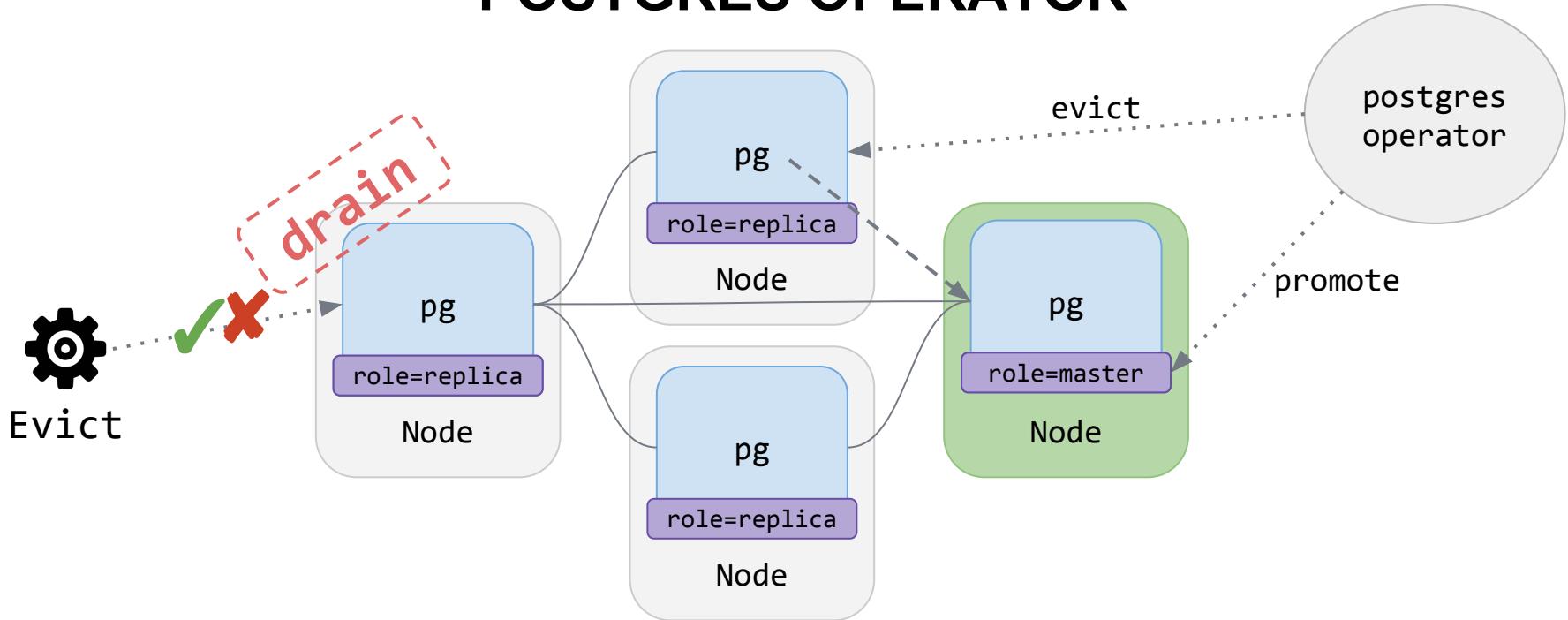


STATEFUL WORKLOADS (POSTGRES)

POSTGRES AS A SERVICE

```
apiVersion: "acid.zalan.do/v1"
kind: "postgresql"
metadata:
  name: "test-cluster"
  namespace: "default"
spec:
  teamId: "teapot"
  postgresql:
    version: "10"
  numberOfInstances: 3
  volume:
    size: "10Gi"
```

POSTGRES OPERATOR

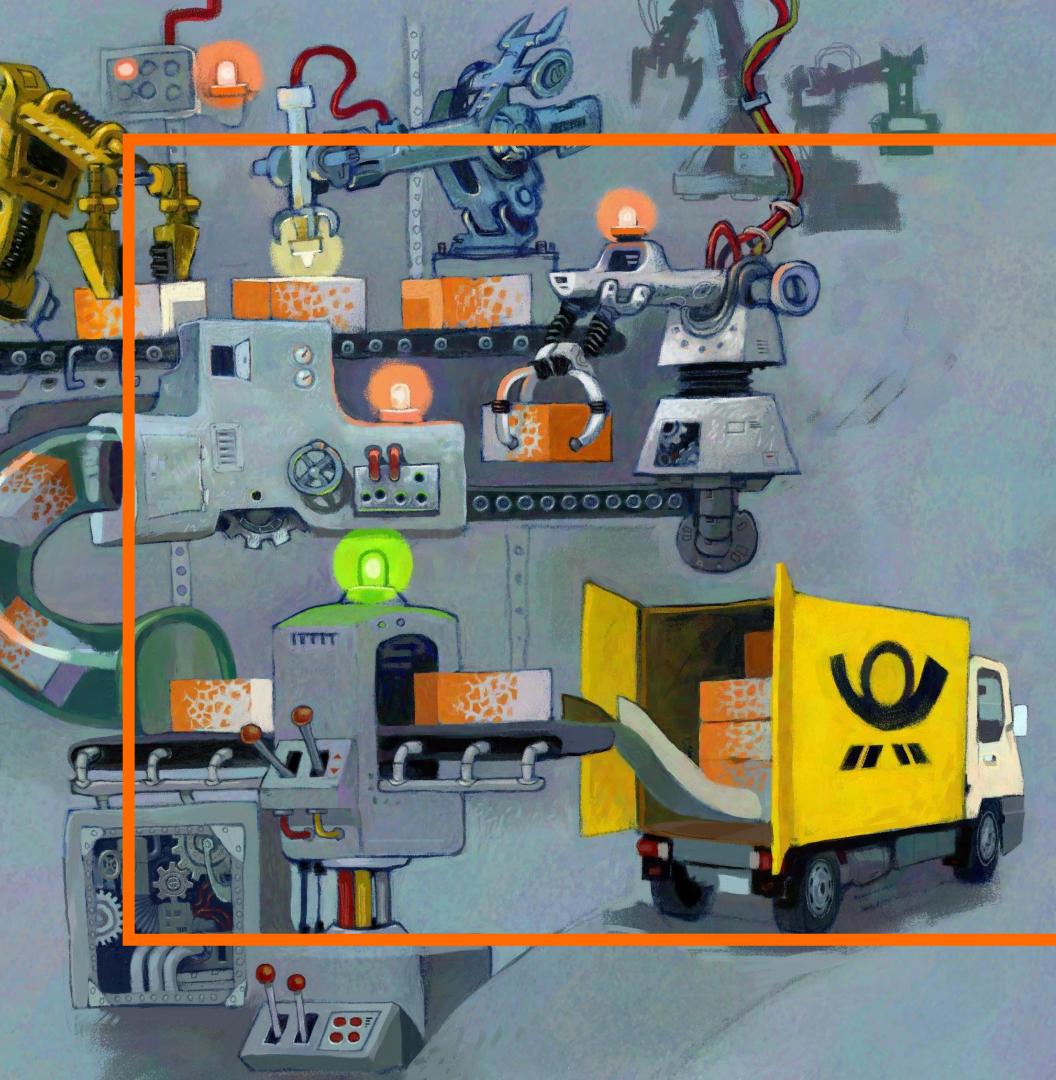


github.com/zalando-incubator/postgres-operator

POSTGRES OPERATOR

```
apiVersion: policy/v1beta1
kind: PodDisruptionBudget
metadata:
  name: "postgres-cluster"
spec:
  minAvailable: 1
  selector:
    matchLabels:
      application: "postgres-cluster"
      role: "master"
```

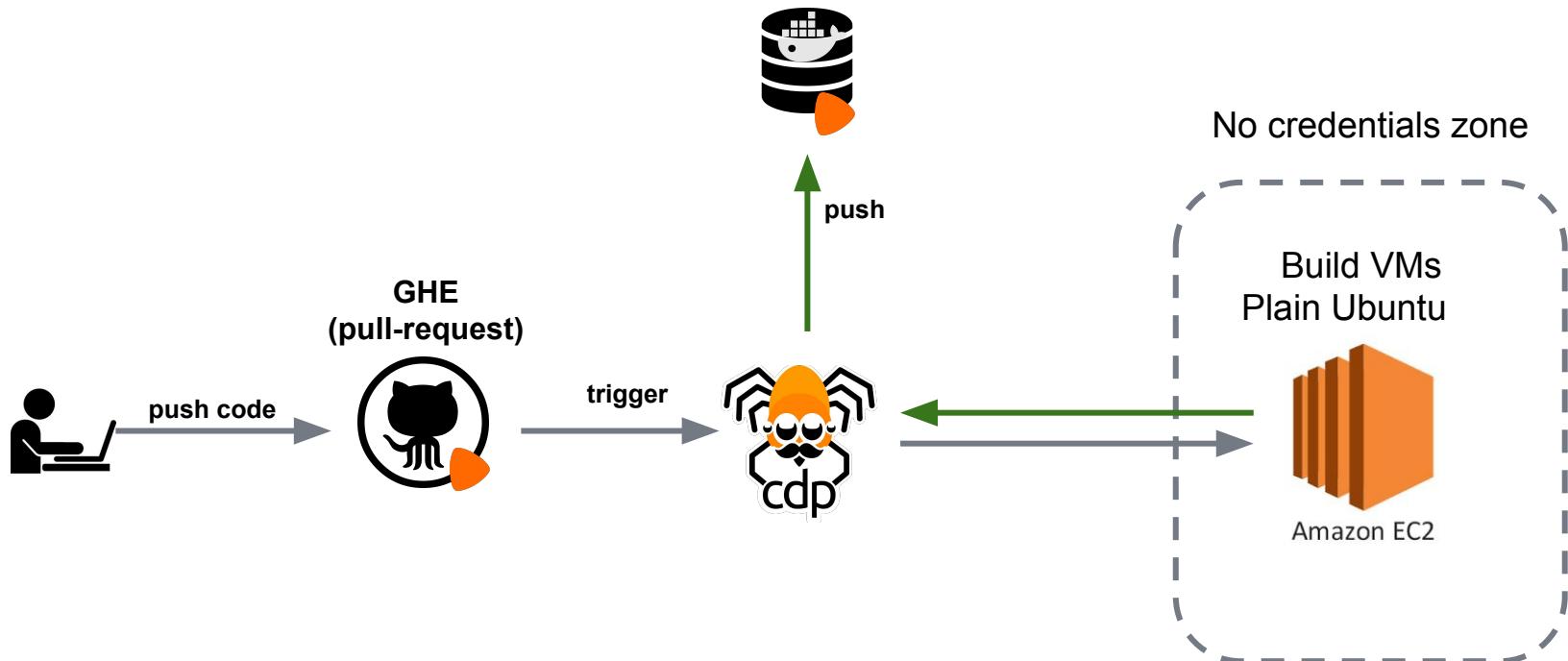
github.com/zalando-incubator/postgres-operator



CI/CD

CONTINUOUS DELIVERY PLATFORM (CDP)

BUILD PIPELINE



CONTINUOUS DELIVERY PLATFORM (CDP)

BUILD DEFINITION

The screenshot shows a developer console interface for a Continuous Delivery Platform (CDP). The top navigation bar includes 'Developer Console', a search bar, and links for 'Feedback' and 'Support'. On the left, a sidebar lists 'APPLICATIONS' (Pipelines) and 'TOOLS' (Repositories, Clusters). The main area displays a 'DEPLOYMENT UNITS' section with tabs for 'PIERTWO' (selected), 'PIERTWO', and 'PR-1-8'. Below this is a detailed view of the 'PIERTWO' build configuration, divided into sections: GENERAL, WARNING, ERROR, and DELIVERY CONFIGURATION (selected). The DELIVERY CONFIGURATION section contains a JSON-like tree structure representing the build steps:

```
root: {} 2 keys
  version: "2017-09-20"
  pipeline: [] 1 item
    0: {} 5 keys
      id: "build"
      env: {} 1 key
        LEIN_ROOT: "true"
      type: "script"
      overlay: "ci/clojure-for-cdp"
      commands: [] 2 items
        0: {} 2 keys
          cmd: "docker run -d -p 5432:5432 postgres lein test"
          desc: "Run unit tests"
        1: {} 2 keys
          cmd: "lein set-version "${CDP_BUILD_VERSION}" lein do clean, uberjar IMAGE="pierone.stups.zalan.do/automata/pierone:${CDP_BUILD_VERSION}" docker build -t "$IMAGE" . if [ -z "$CDP_PULL_REQUEST_NUMBER" ]; then docker push "$IMAGE" fi TEST_IMAGE="pierone.stups.zalan.do/automata/pierone-test:${CDP_BUILD_VERSION}" docker tag "$IMAGE" "$TEST_IMAGE" docker push "$TEST_IMAGE"
          desc: "Build and push docker image"
```

CONTINUOUS DELIVERY PLATFORM (CDP)

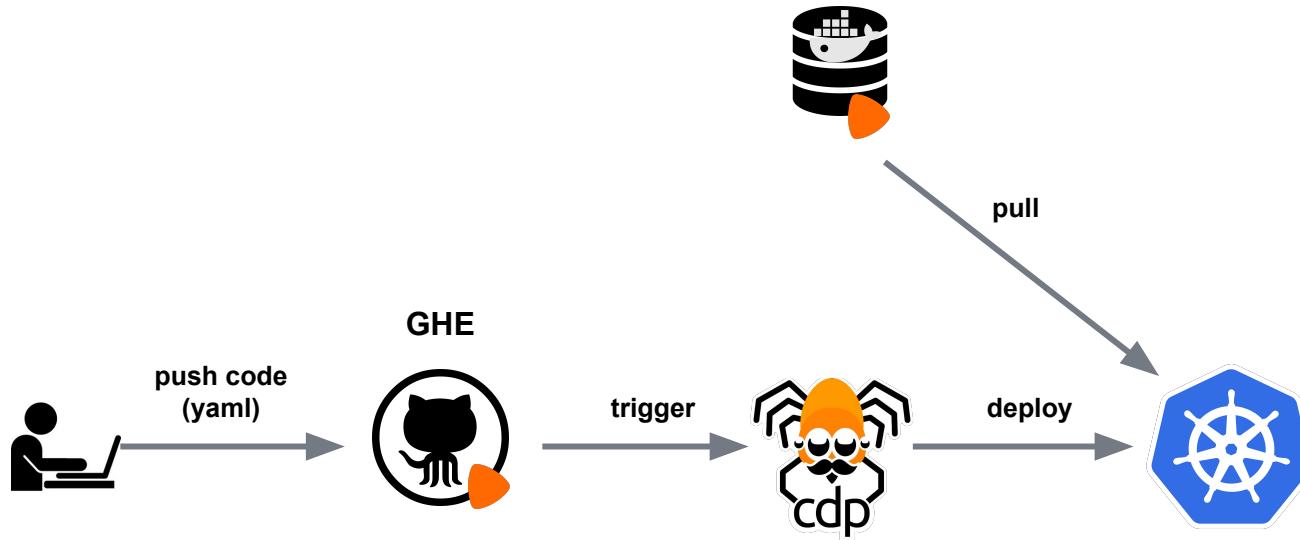
BUILD OUTPUT

The screenshot shows a web-based Continuous Delivery Platform (CDP) interface. At the top, there's a navigation bar with a menu icon, 'Developer Console' (highlighted in blue), a search bar, and links for 'Feedback' and 'Support'. On the left, a sidebar has sections for 'APPLICATIONS' (Pipelines) and 'TOOLS' (Repositories, Clusters). The main area shows 'DEPLOYMENT UNITS' (PIERTWO, PR-1-8). The 'PIERTWO' tab is selected, showing a green header with 'PIERTWO', a gear icon, and 'BUILD'. Below is a log table with columns: GENERAL, WARNING, ERROR, LOGS (which is active and highlighted in blue). A status message 'Ran for 4 m, 22 s' is shown. The log content is as follows:

```
00:14:07 67885e448177: Preparing
00:14:07 ec75999a0cb1: Preparing
00:14:07 65bdd50ee76a: Preparing
00:14:07 059ad60bcacf: Waiting
00:14:07 8db5f072feec: Waiting
00:14:07 67885e448177: Waiting
00:14:07 ec75999a0cb1: Waiting
00:14:07 65bdd50ee76a: Waiting
00:14:07 e367b8804fd0: Layer already exists
00:14:07 3766357ab54b: Layer already exists
00:14:07 059ad60bcacf: Layer already exists
00:14:07 8db5f072feec: Layer already exists
00:14:07 67885e448177: Layer already exists
00:14:07 ec75999a0cb1: Layer already exists
00:14:07 65bdd50ee76a: Layer already exists
00:14:07 0169d778678b: Pushed
00:14:07 545dfec09f52: Pushed
00:14:15 ba10d51f56c: Pushed
00:14:16 pr-1-8: digest: sha256:4386c7a86c78f31514a9551172a37d0e1e082545f3ee741b46a5e5862c354429 size: 2407
00:14:16 Step 'Build and push docker image' finished successfully.
00:14:16 Build Successful.
00:14:16 Closing the connection to the build host
```

CONTINUOUS DELIVERY PLATFORM (CDP)

DEPLOYMENT PIPELINE



CONTINUOUS DELIVERY PLATFORM (CDP)

DEPLOYMENT DEFINITION

The screenshot shows the Zalando Developer Console interface. The left sidebar has sections for APPLICATIONS (Pipelines) and TOOLS (Repositories, Clusters). The main area displays a deployment configuration for 'CLUSTER-LIFECYCLE-MANAGER-DEPL...'. The top navigation bar includes a search bar, 'Feedback' and 'Support' links, and tabs for 'DEPLOYMENT UNITS', 'CLUSTER-LIFECYCLE-MANAGER-DEPLOY', and 'MASTER-140'. Below these are tabs for 'DEPLOY-STAGING' and 'DEPLOY-PROD'. The central content area shows a JSON-like delivery configuration:

```
CLUSTER-LIFECYCLE-MANAGER-DEPL...
GENERAL WARNING ERROR DELIVERY CONFIGURATION

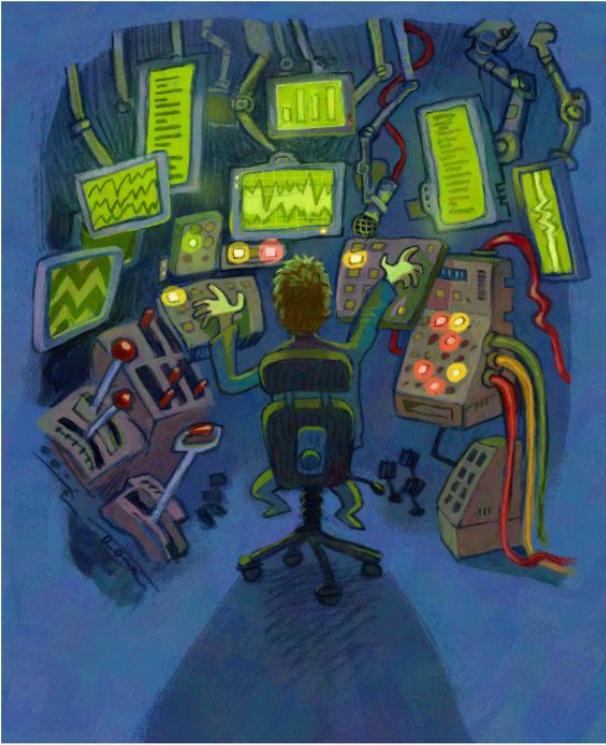
{
  "1": {
    "keys": 6,
    "id": "deploy-prod",
    "desc": "Deploy to production (stups)",
    "type": "process"
  },
  "config": {
    "keys": 1
  },
  "apply_permanent_resources": {
    "keys": 1
  },
  "env": [
    {
      "0": {
        "keys": 2,
        "name": "APPLICATION",
        "value": "cluster-lifecycle-manager"
      },
      "1": {
        "keys": 2,
        "name": "IMAGE",
        "value": "registry.opensource.zalan.do/teapot/cluster-lifecycle-manager"
      },
      "2": {
        "keys": 2,
        "name": "VERSION",
        "value": "32f9efa"
      },
      "3": {
        "keys": 2,
        "name": "AWS_ACCOUNT",
        "value": "786011980701"
      }
    }
  ]
}
```

CONTINUOUS DELIVERY PLATFORM (CDP)

DEPLOYMENT OUTPUT

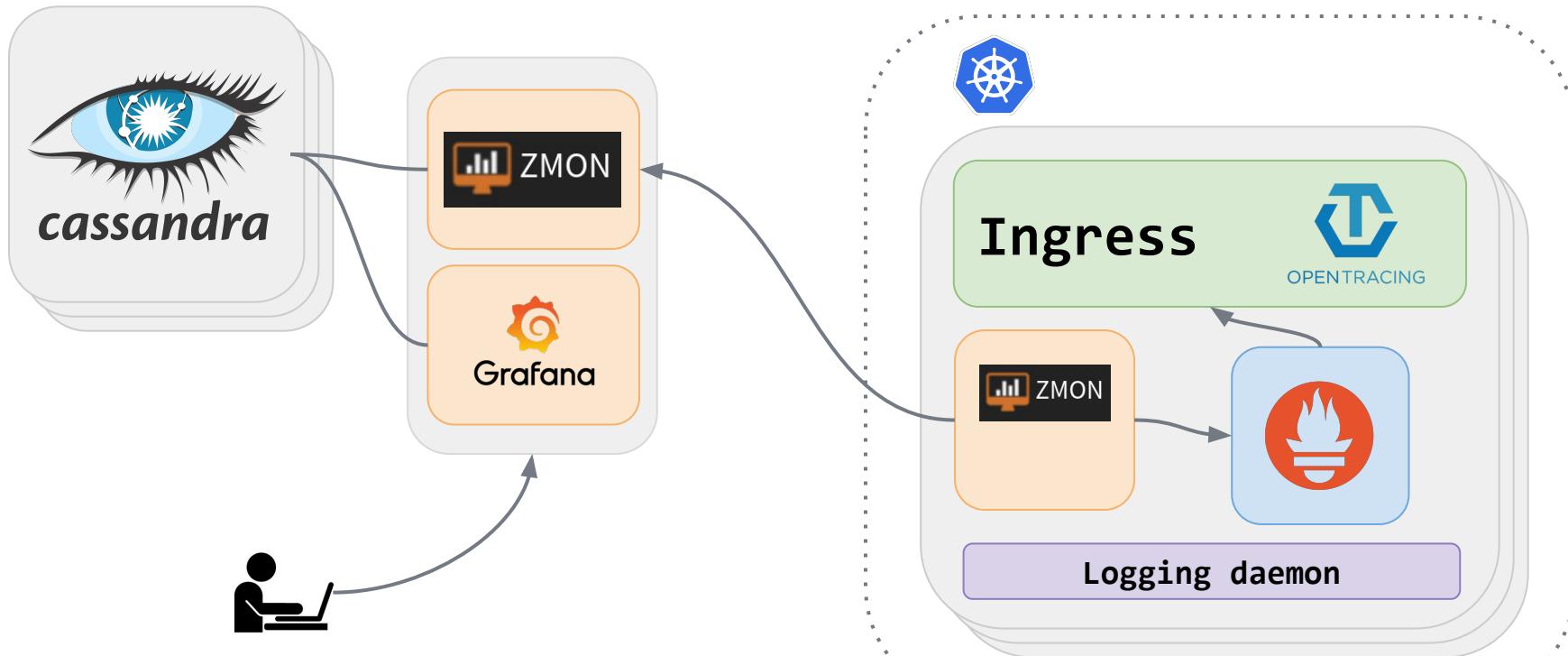
The screenshot shows the Zalando Developer Console interface. On the left, there's a sidebar with sections for APPLICATIONS (Pipelines) and TOOLS (Repositories, Clusters, Kube Resource Report, Lightstep Manager). The main area has a search bar at the top right and tabs for DEPLOYMENT UNITS, PIERONE-DEPLOY, and MASTER-9. The PIERONE-DEPLOY tab is active, showing a green header bar with the name and a cloud icon. Below it is a table with two columns: GENERAL and LOGS. The LOGS column displays log entries from a CloudFormation stack:

LOGS
3:58:55 PM INFO[0006] CloudFormation Stack is ready id="arn:aws:cloudformation:eu-central-1:786011 3:58:55 PM INFO[0006] Waiting for Ingress to become ready... name=pierone namespace=default prober=ingress 3:58:55 PM INFO[0006] Waiting for DNS record to become ready... dns=pierone.stups.zalan.do ip= lb=aws-7860-1v9y5j1w2c9ac-17 3:58:55 PM INFO[0006] Resolved IPs for desired DNS record and its ELB don't match dns=pierone.stups.zalan.do ip= lb=aws-7860-1v9y5j1w2c9ac-17 3:58:55 PM INFO[0006] Waiting for DNS record to become ready... dns=pierone-k8s.stups.zalan.do ip= lb=aws-7860-1v9y5j1w2c9ac-17 3:58:55 PM INFO[0007] DNS record is ready dns=pierone-k8s.stups.zalan.do ip= lb=aws-7860-lb-1v9y5j1w2c9ac-17 3:58:55 PM INFO[0007] Ingress is ready name=pierone namespace=default prober=ingress 3:58:55 PM INFO[0007] Waiting for Service to become ready... name=pierone namespace=default prober=service 3:58:55 PM INFO[0007] Waiting for DNS record to become ready... dns=pierone-elb.stups.zalan.do ip= lb=adc7482d2ddd11e880c20648805 3:58:55 PM INFO[0007] DNS record is ready dns=pierone-elb.stups.zalan.do ip= lb=adc7482d2ddd11e880c20648805 3:58:55 PM INFO[0007] Service is ready name=pierone namespace=default prober=service 3:58:55 PM === [main] apply_permanent_resources finished with exitCode: '0', status: 'terminated', reason: ''



OBSERVABILITY

OBSERVABILITY



ZMON CHECKS

The screenshot shows the ZMON web interface. At the top, there's a navigation bar with links for 'Dashboards', 'Cloud', 'Entities', 'Check defs' (which is highlighted in white), 'Alert defs', and 'Trial Run'. Below the navigation is a status bar with the text '0 in queue, 1/1 active, 0.00/s' and icons for notifications, help, search, and user 'mlarsen'. To the right of the status bar are buttons for 'Edit', 'Trial Run', and 'Changes'.

Check: Application Health

Description: Example ZMON check definition which returns a HTTP status code

```
def check():
    try:
        return http('/.well-known/health', timeout=5).code()
    except:
        return 999
```

Interval: 1m
Entities: type = kio_application

github.com/zalando/zmon

ZMON ALERTS

ZMON Dashboards Cloud Entities Check defs Alert defs Trial Run

0 in queue, 1/1 active, 0.00/s 🔔 ? 📈 🔎 mlarsen

ID: 12846, Status: , Team: Teapot

[Edit](#) [Clone](#) [Inherit](#) [Delete](#) [Changes](#) [Trial Run](#) [Evaluate](#)
[Cleanup](#) [Comments \(0\)](#)

Alert: even: Application unhealthy ↑

Description	Checking if we can reach the HTTP endpoint
Condition	<pre>def alert(): return alert_series(lambda v: v >= 500, 3)</pre>
Responsible Team	Teapot

github.com/zalando/zmon

ZMON DASHBOARD

 ZMON Dashboards Cloud Entities Check defs Alert defs Trial Run

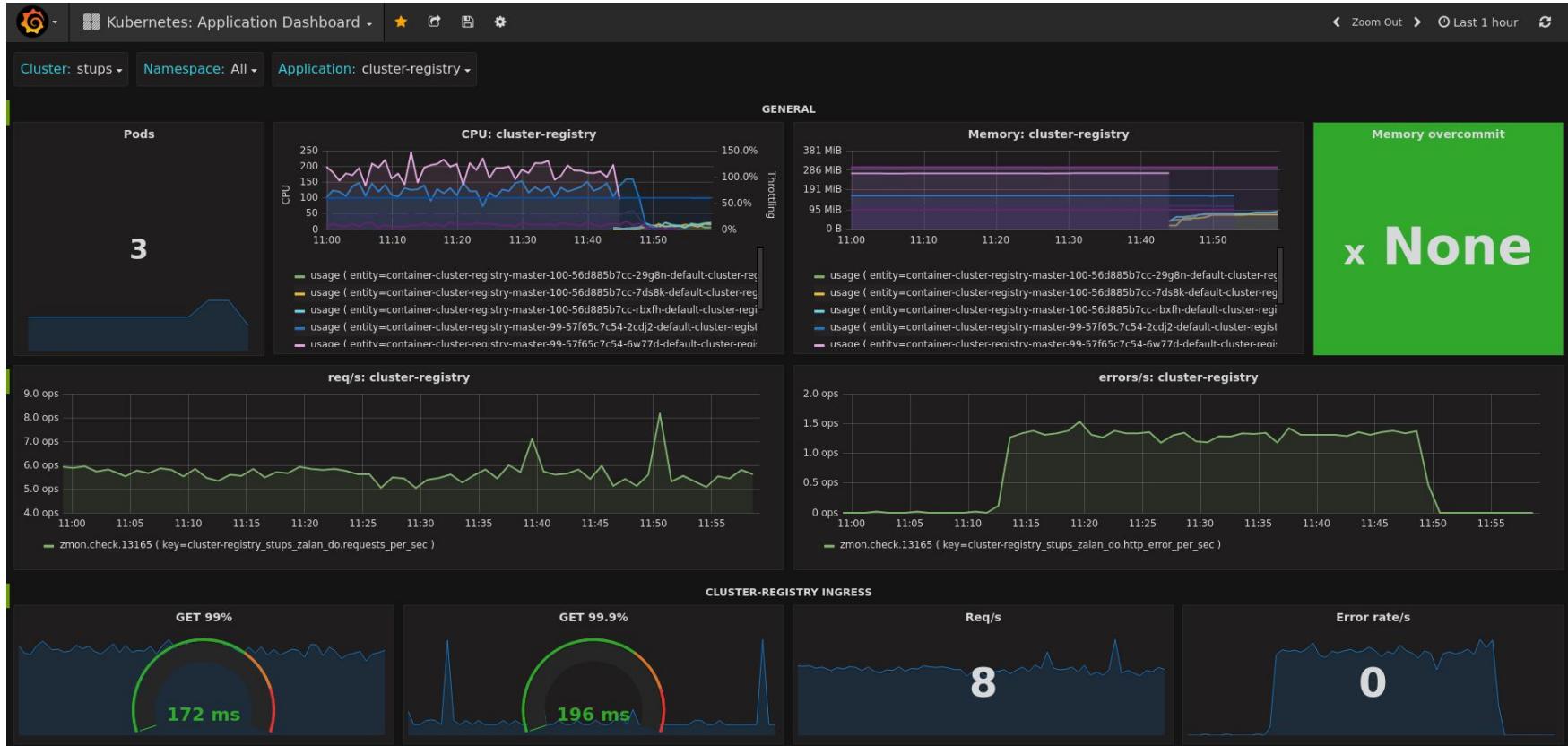
0 in queue, 1/1 active, 0.00/s 11:39 🔔 ? 📈 🔍 mlarsen

🔍 Search for alerts ⚡ 3 Hide Widgets ⌂ ⚙️

 Kubernetes: CLM clusters with problems: {cluster} (27)	3m	etcd: snapshots too old "governance: no backups"	29m
 AWS limits insufficient {info} (23)	2m	Cluster Registry success rate SLO breach	17m
Kubernetes DNS healthy	4h	Cluster ASG scaling issues (prod): "aws:343040485429 (0..0 instances, 1..1 desired)"	8h

github.com/zalando/zmon

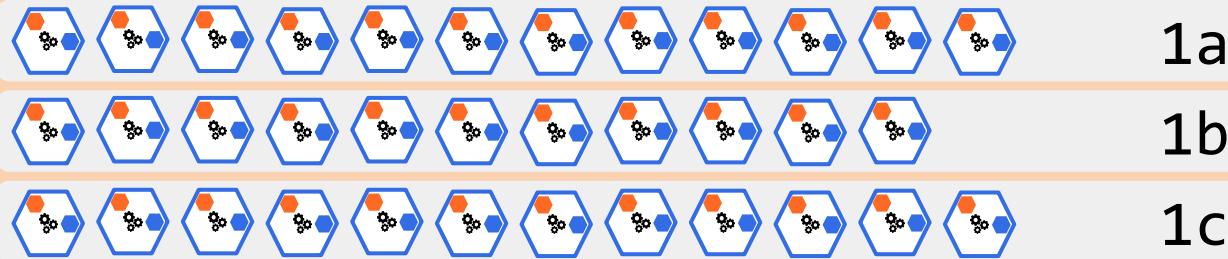
GRAFANA APPLICATION DASHBOARD





CLUSTER AUTOSCALER

Node Pool: General purpose (m5.2xlarge)



Node Pool: GPU (p3.2xlarge)

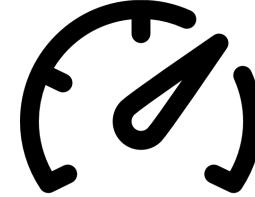


HORIZONTAL POD AUTOSCALING (CUSTOM METRICS)



amazon
SQS

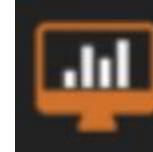
Queue Length



Ingress Req/s



Prometheus Query

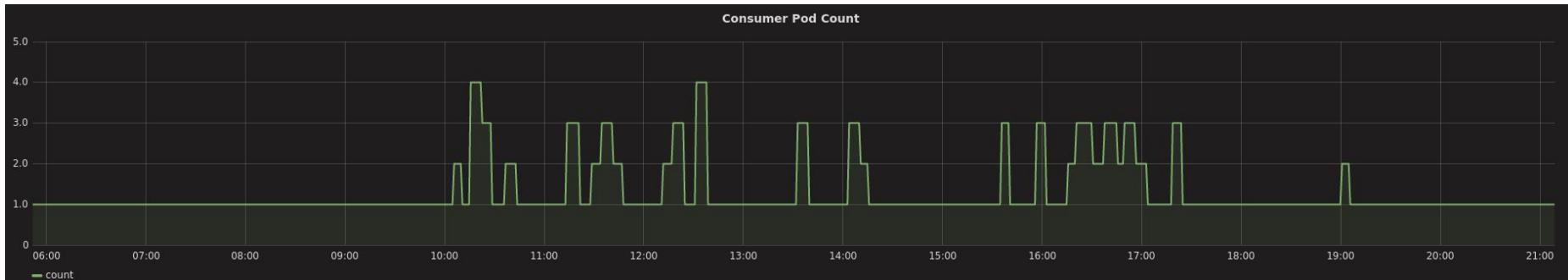
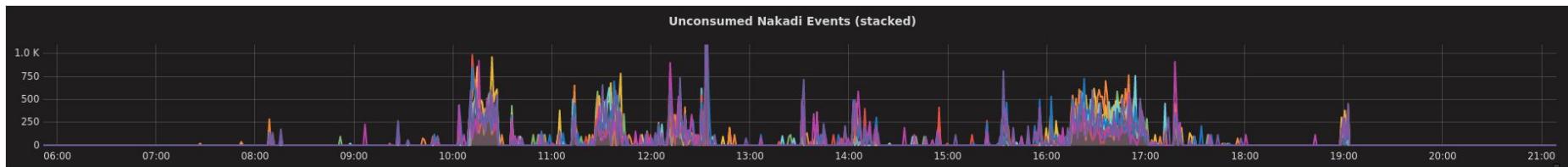


ZMON Check

github.com/zalando-incubator/kube-metrics-adapter

HORIZONTAL POD AUTOSCALING (CUSTOM METRICS)

JOB SCALING TO CONSUME EVENTS (ZMON CHECK)



OPEN SOURCE

Cluster Lifecycle Manager

github.com/zalando-incubator/cluster-lifecycle-manager

Kubernetes on AWS

github.com/zalando-incubator/kubernetes-on-aws

AWS ALB Ingress controller

github.com/zalando-incubator/kube-ingress-aws-controller

Skipper HTTP Router & Ingress controller

github.com/zalando/skipper

External DNS

github.com/kubernetes-incubator/external-dns

Postgres Operator

github.com/zalando-incubator/postgres-operator

Kube Metrics Adapter

github.com/zalando-incubator/kube-metrics-adapter





MIKKEL LARSEN

SENIOR SOFTWARE ENGINEER
PLATFORM INFRASTRUCTURE

mikkel.larsen@zalando.de

[@mikkeloscar](https://twitter.com/mikkeloscar)

Illustrations by [@01k](#), [@kcgrenn](#), [@ntakayama](#)

2018-07-12

