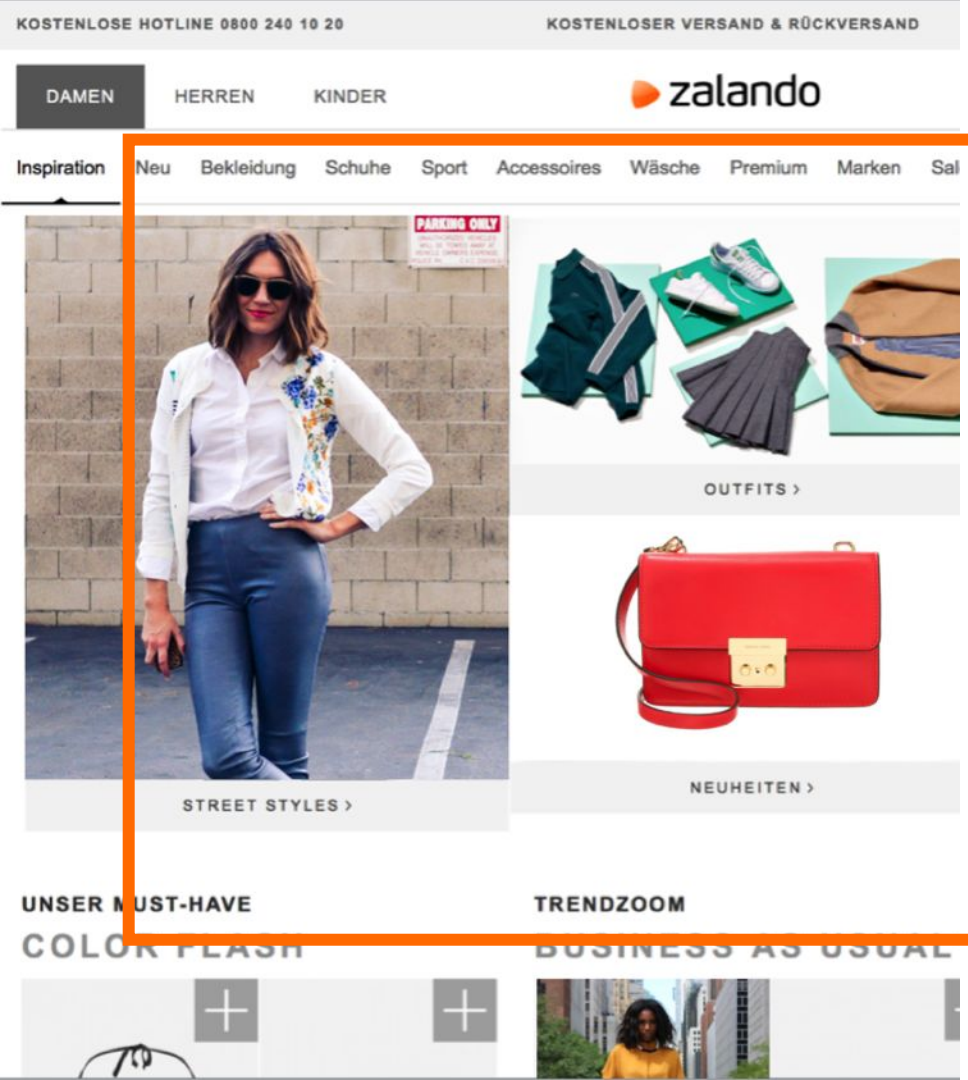

AWS
re:Invent

OPN211

How Zalando runs Kubernetes clusters at scale on AWS

Henning Jacobs
Senior Principal
Zalando SE



THE EUROPEAN ONLINE PLATFORM FOR FASHION

zalando

ZALANDO AT A GLANCE

~ 5.4 billion EUR
revenue 2018

> 300 million
visits per month

~ 14,000
employees in Europe

> 80%
of visits via mobile devices

> 28 million
active customers

> 400,000
product choices

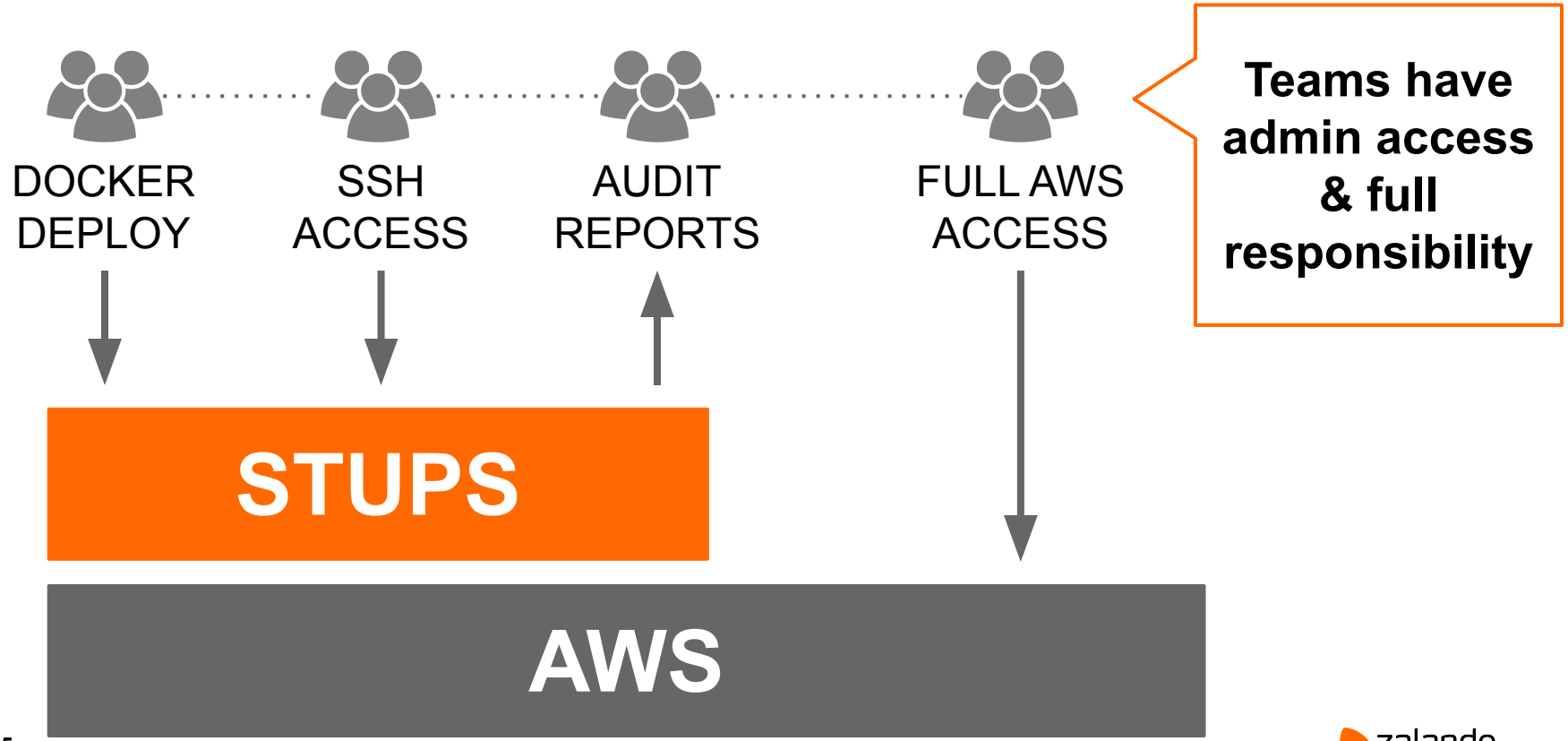
> 2,000
brands

17
countries

as of June 2019

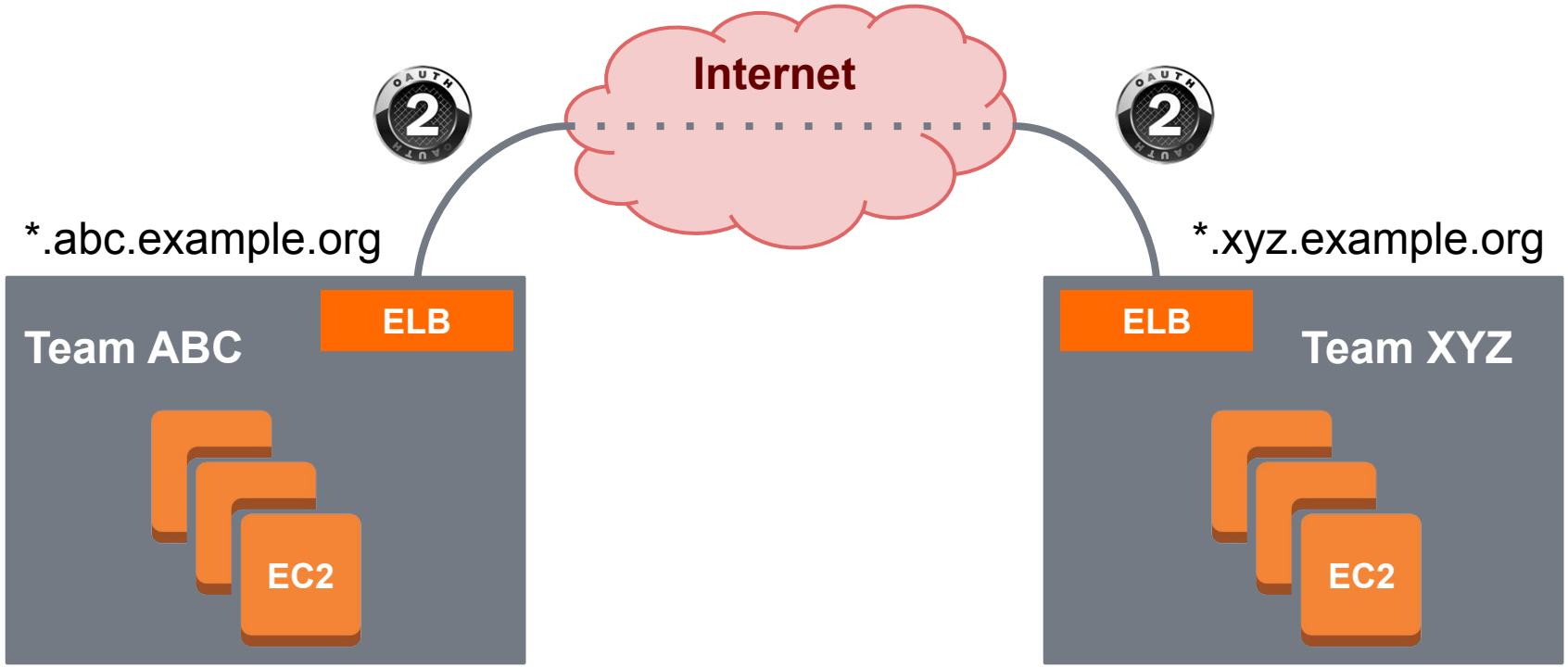
zalando

2015: JOURNEY INTO THE CLOUD



zalando

2015: ISOLATED AWS ACCOUNTS



zalando

INFRASTRUCTURE @ 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

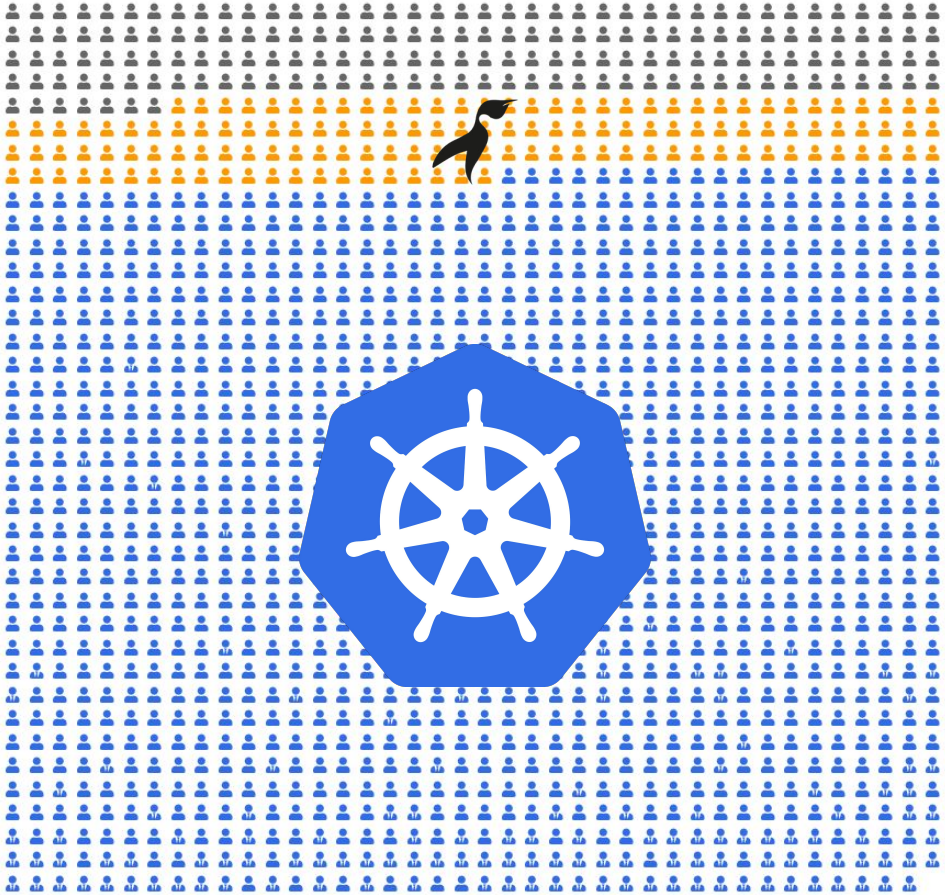
- Clusters per product (multiple teams).
- Instances are not managed by teams.
- Hands off approach.
- A lot of stuff out of the box.

2019: SCALE

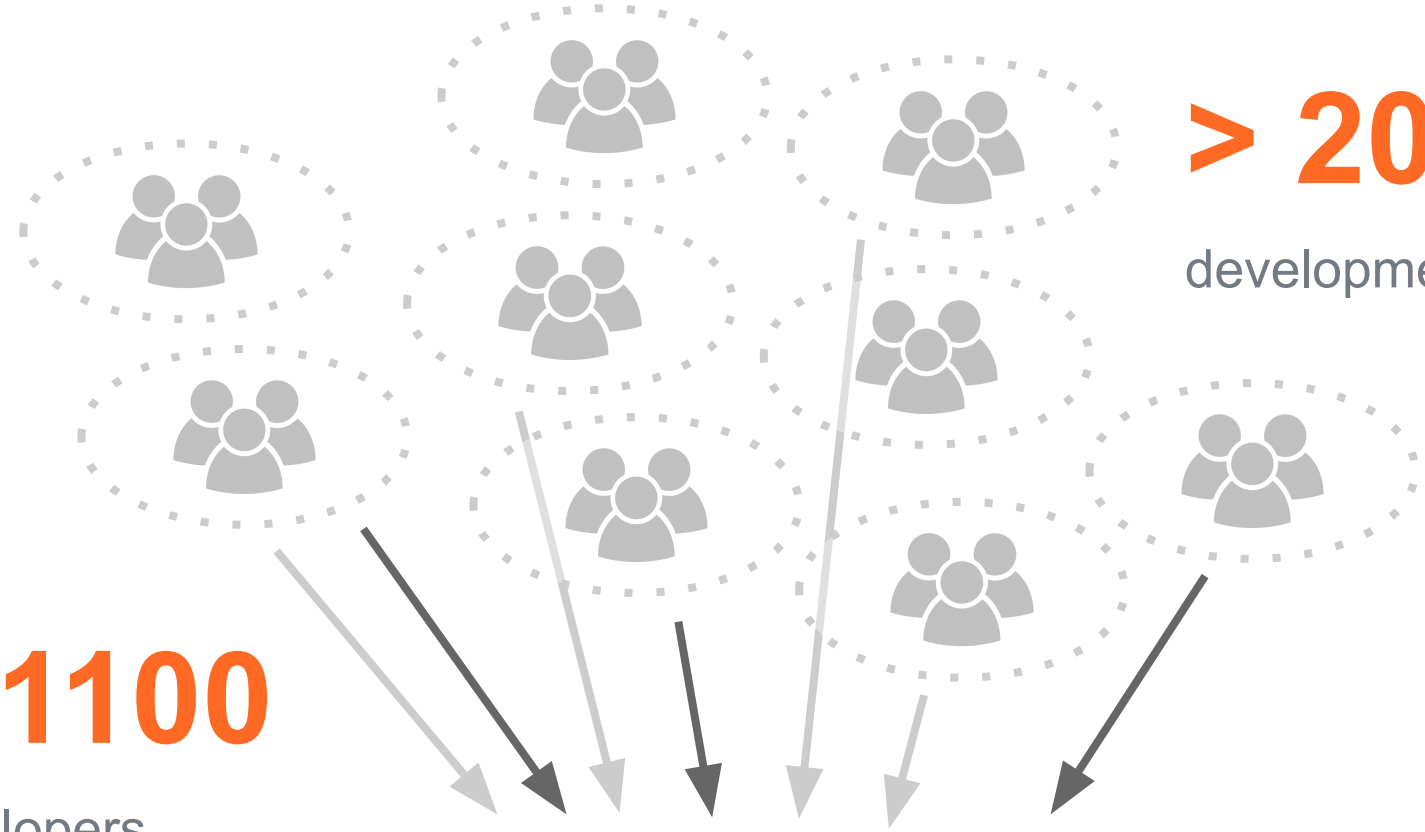
396 Accounts

140 Clusters

2019: DEVELOPERS USING KUBERNETES



zalando



> 200

development teams

> 1100

developers

Platform

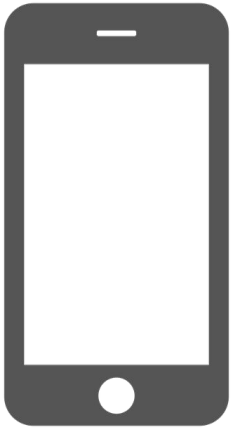
zalando

YOU BUILD IT, YOU RUN IT

The traditional model is that you take your software to the wall that separates development and operations, and throw it over and then forget about it. Not at Amazon. ***You build it, you run it****. This brings developers into contact with the day-to-day operation of their software. It also brings them into day-to-day contact with the customer.*

- A Conversation with Werner Vogels, ACM Queue, 2006

ON-CALL: YOU OWN IT, YOU RUN IT



When things are broken,
we want people with the best
context trying to fix things.

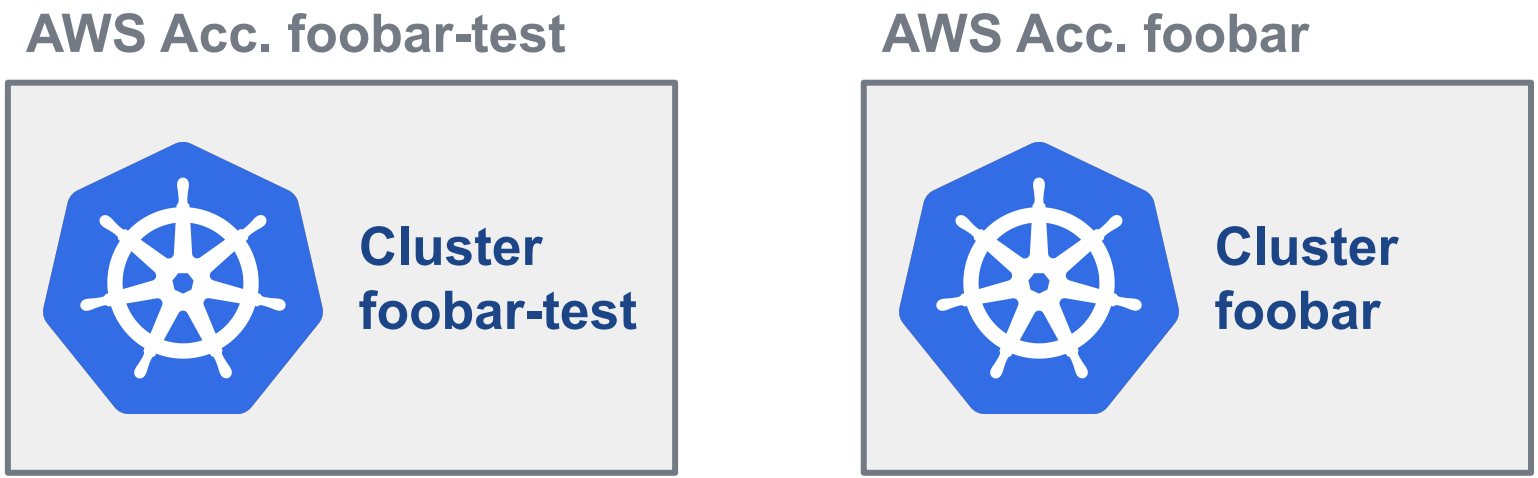
- Blake Scrivener, Netflix SRE Manager

GOALS

- No manual operations
- No pet clusters
- Reliability
- Autoscaling
- Latest Kubernetes
- Cost efficient

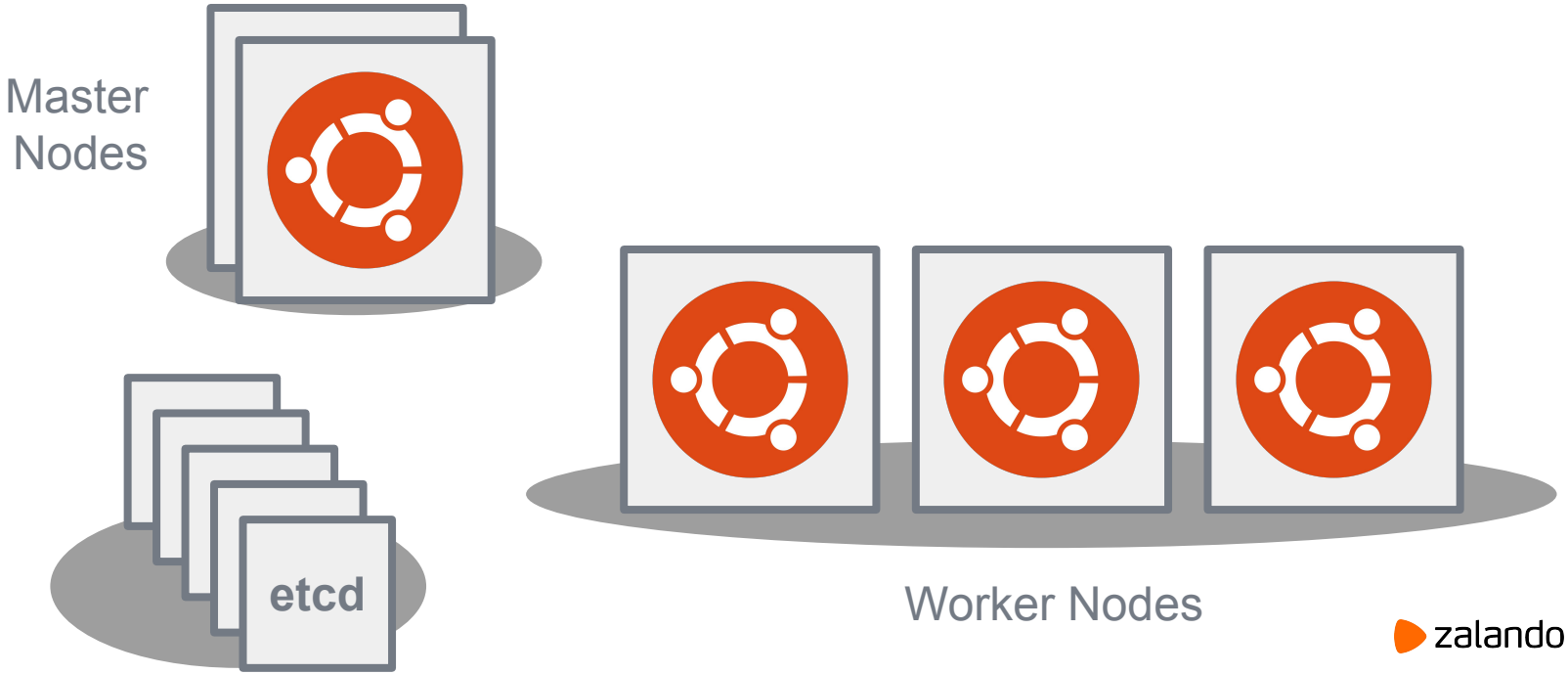
ARCHITECTURE

Pairs of clusters, each cluster in isolated account



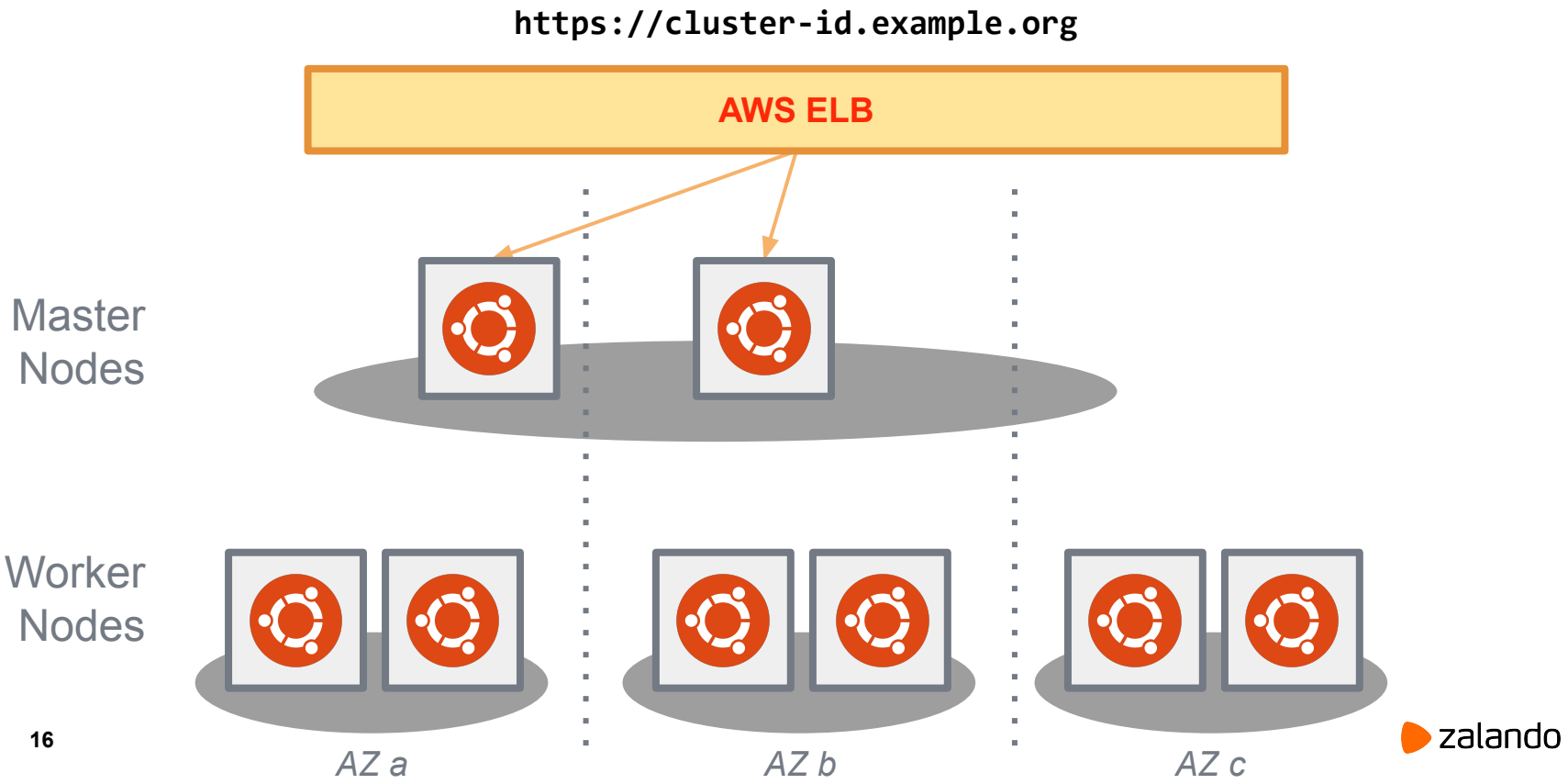
ARCHITECTURE

CloudFormation stacks, node pools w/ self-baked Ubuntu AMI



zalando

ARCHITECTURE



zalando

CLUSTER METADATA (CLUSTER-REGISTRY)

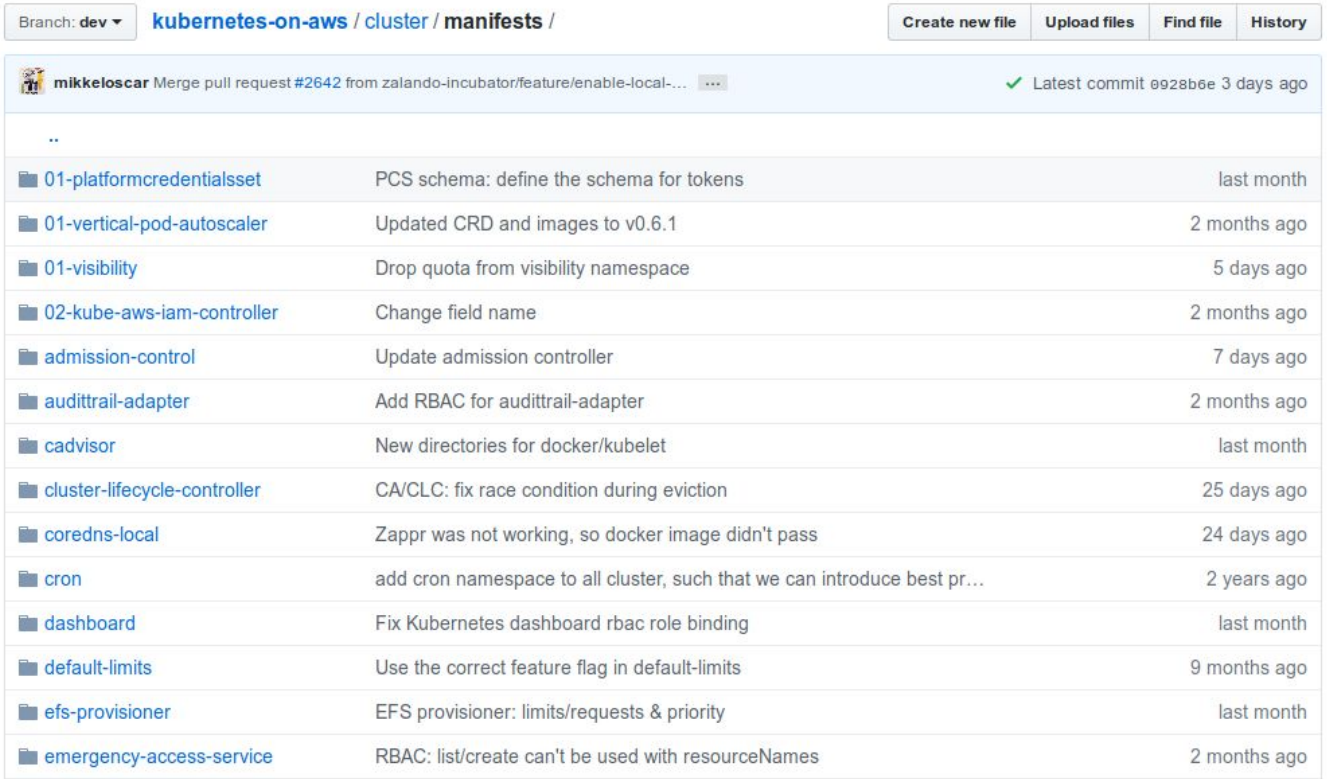
```
clusters:
- id: “cluster-id”
  api_server_url: “https://cluster-id.example.org”
  config_items:
    Key: “value”
  environment: “test”
  region: “eu-central-1”
  lifecycle_status: “ready”
  node_pools:
  - name: “worker-pool”
    instance_type: “m5.large”
    min_size: 3
    max_size: 20
```

CLUSTER CONFIGURATION

```
cluster
├── cluster.yaml       # Kubernetes cluster stack
├── etcd-cluster.yaml  # etcd cluster stack
├── manifests
│   ├── ...
└── node-pools         # master/worker nodes
    ├── ...
```

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

KUBERNETES CLUSTER MANIFESTS



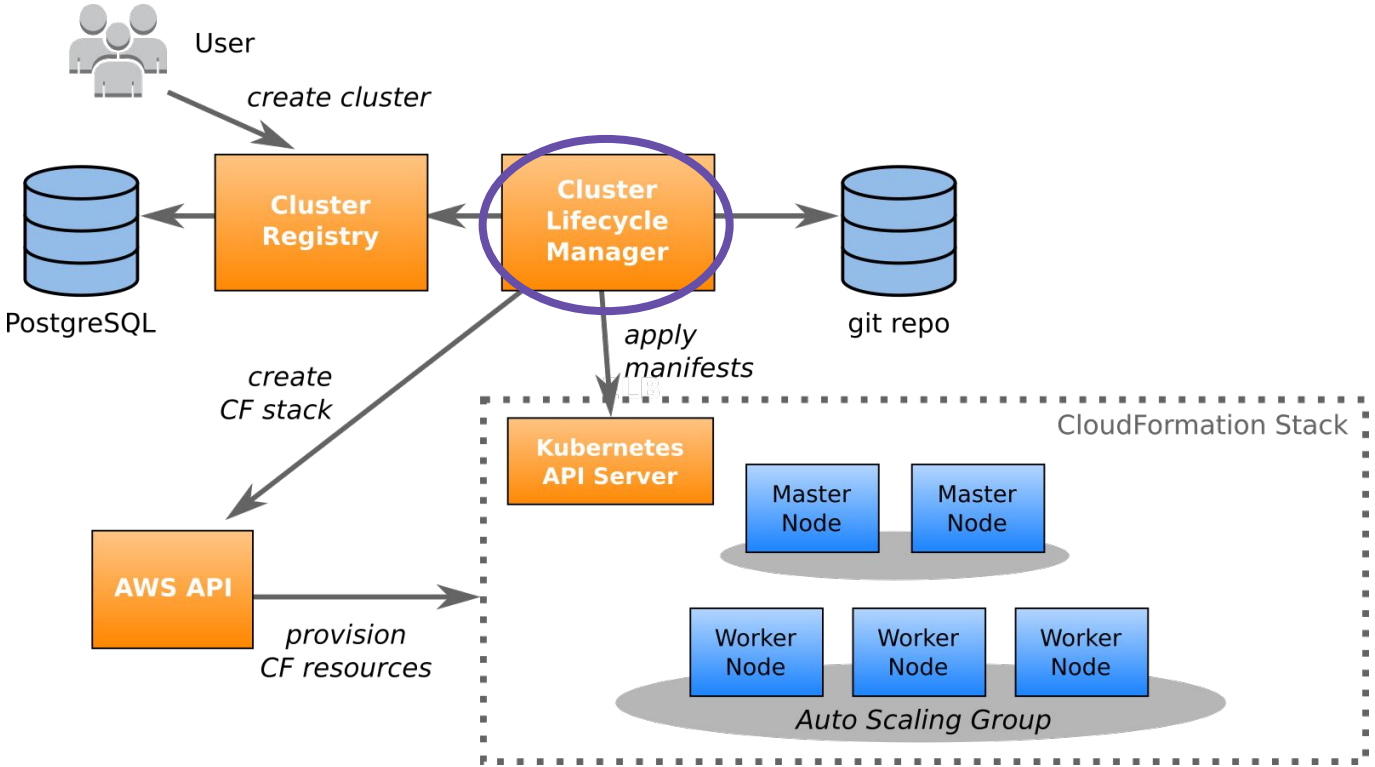
Branch: dev ▾ kubernetes-on-aws / cluster / manifests / — Create new file | Upload files | Find file | History

mikkeloscar Merge pull request #2642 from zalando-incubator/feature/enable-local-… — ✓ Latest commit 0928b6e 3 days ago

..		
01-platformcredentialsset	PCS schema: define the schema for tokens	last month
01-vertical-pod-autoscaler	Updated CRD and images to v0.6.1	2 months ago
01-visibility	Drop quota from visibility namespace	5 days ago
02-kube-aws-iam-controller	Change field name	2 months ago
admission-control	Update admission controller	7 days ago
audittrail-adapter	Add RBAC for audittrail-adapter	2 months ago
cadvisor	New directories for docker/kubelet	last month
cluster-lifecycle-controller	CA/CLC: fix race condition during eviction	25 days ago
coredns-local	Zappr was not working, so docker image didn't pass	24 days ago
cron	add cron namespace to all cluster, such that we can introduce best pr…	2 years ago
dashboard	Fix Kubernetes dashboard rbac role binding	last month
default-limits	Use the correct feature flag in default-limits	9 months ago
efs-provisioner	EFS provisioner: limits/requests & priority	last month
emergency-access-service	RBAC: list/create can't be used with resourceNames	2 months ago

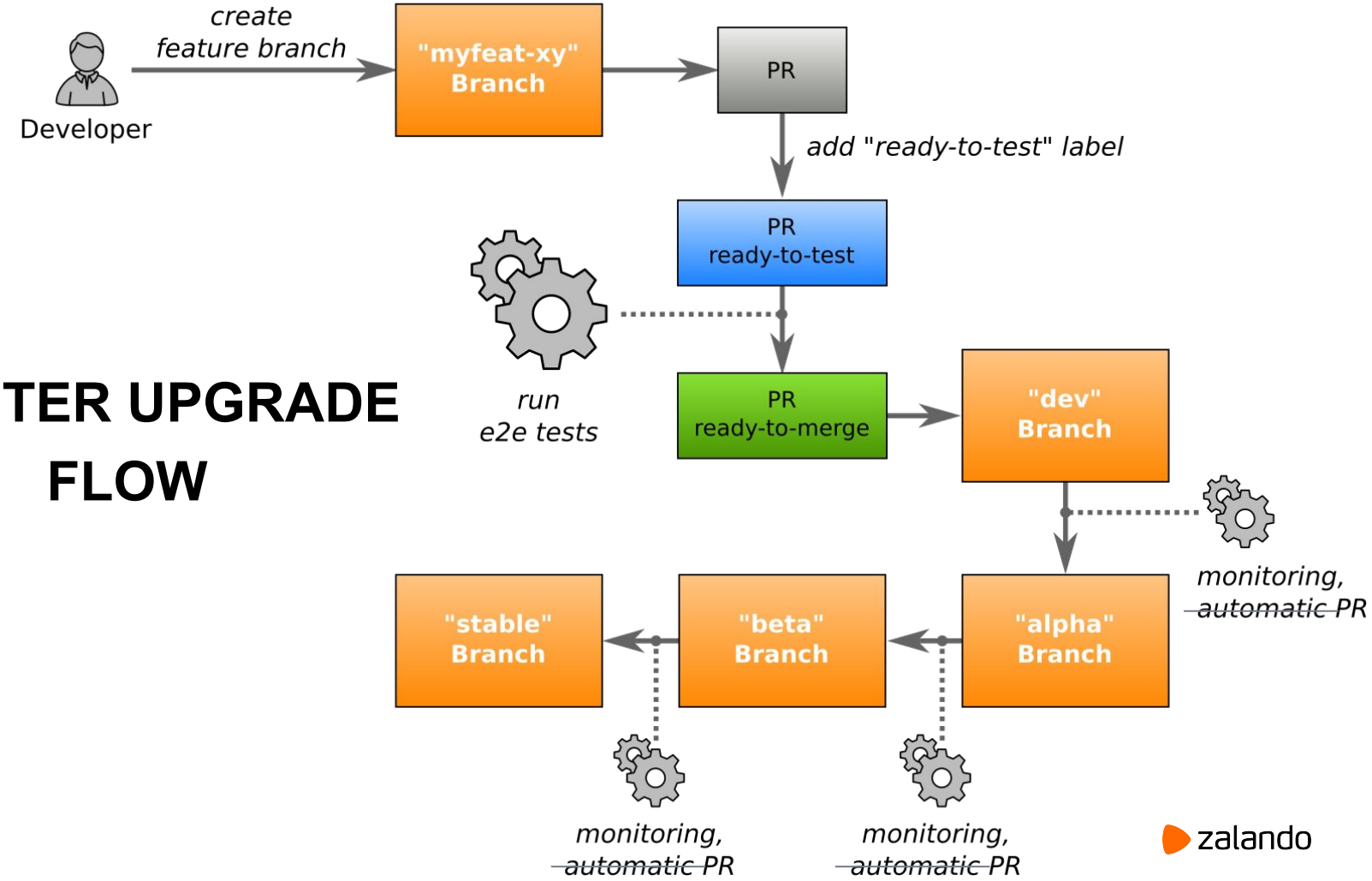
github.com/zalando-incubator/kubernetes-on-aws

CLUSTER LIFECYCLE MANAGER (CLM)



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

CLUSTER UPGRADE FLOW



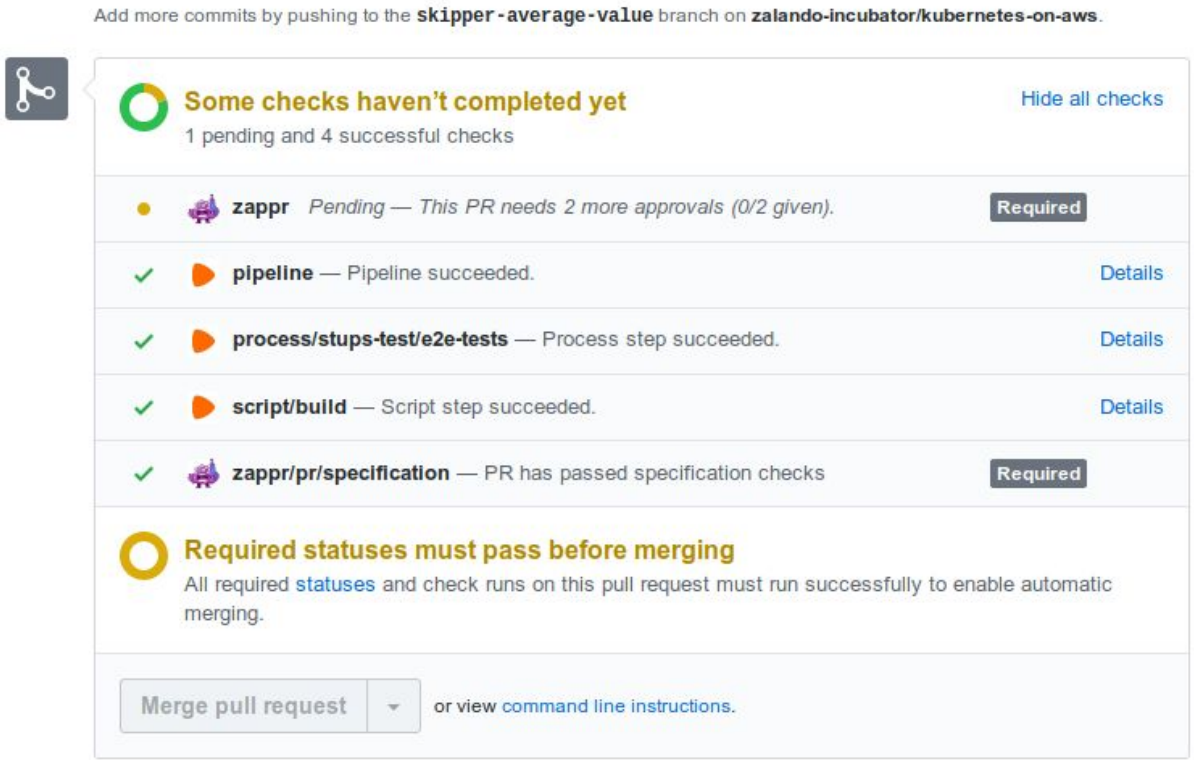
zalando

CLUSTER CHANNELS

Channel	Description	Clusters
`dev`	Development and playground clusters	**3**
`alpha`	Main infrastructure cluster (**important to us**)	**1**
`beta`	Non-prod clusters for the rest of the org	**65+**
`stable`	Production clusters.	**65+**

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

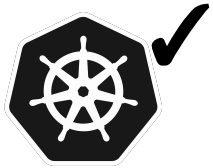
E2E TESTS ON EVERY PR



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

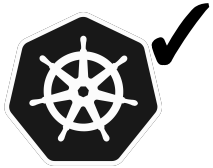
zalando

E2E TESTS



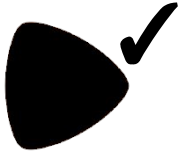
Conformance Tests
Upstream Kubernetes e2e conformance tests

159



StatefulSet Tests
Rolling update of stateful sets including volume mounting

2

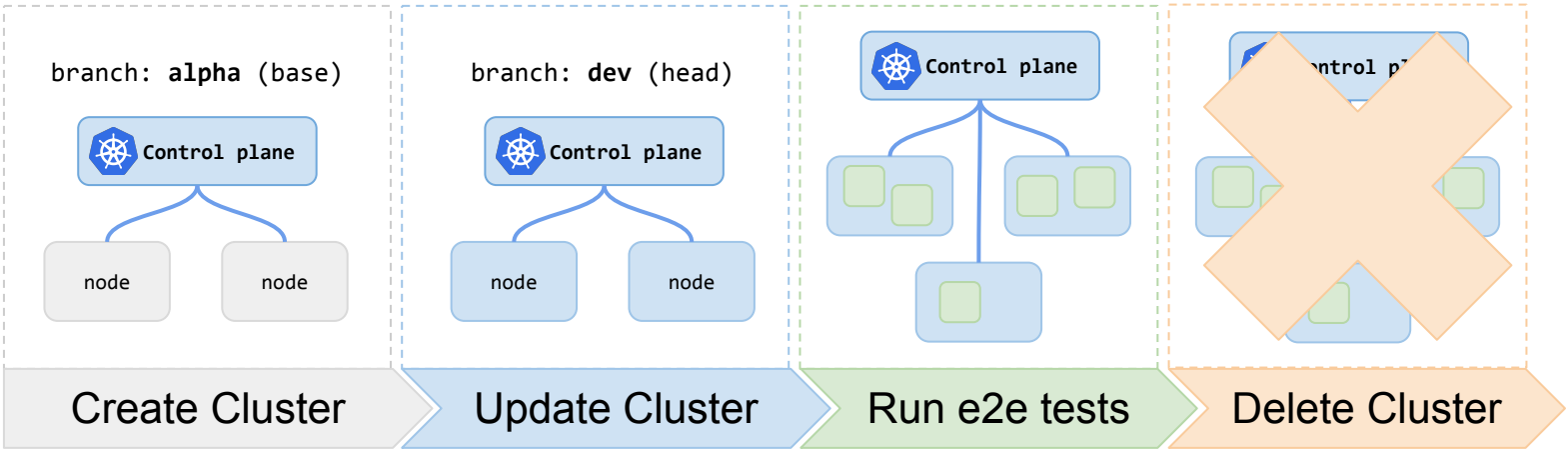


Zalando Tests (custom)
Custom tests for ingress, external-dns, PSP etc.

17

RUNNING E2E TESTS

Testing **dev** to **alpha** upgrade



zalando

UPGRADING NODES

zalando

NAÏVE NODE UPGRADE STRATEGY

zalando

NAÏVE NODE UPGRADE STRATEGY

Set ASG size to current + 1

zalando

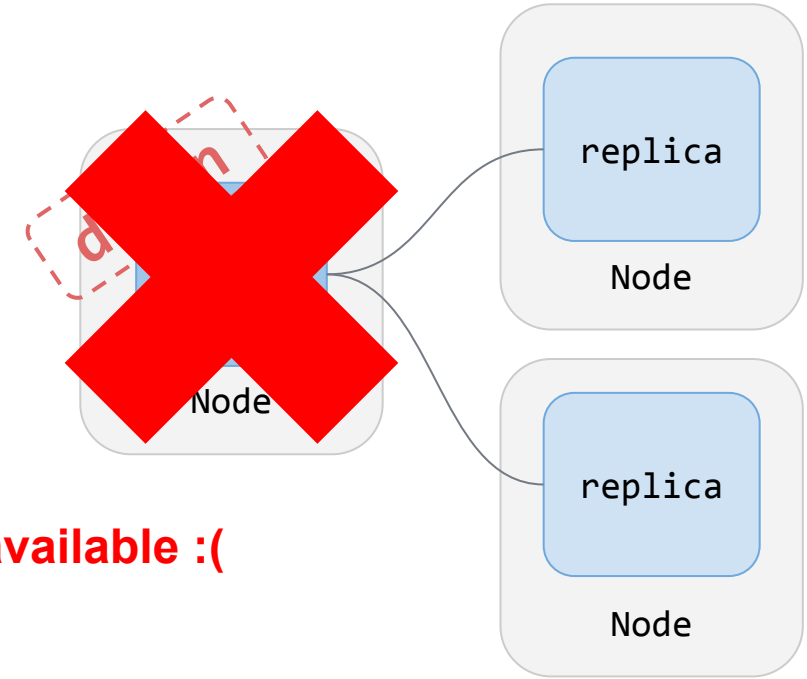
NAÏVE NODE UPGRADE STRATEGY

Get a new instance

zalando

PROBLEMS WITH THE NAÏVE STRATEGY

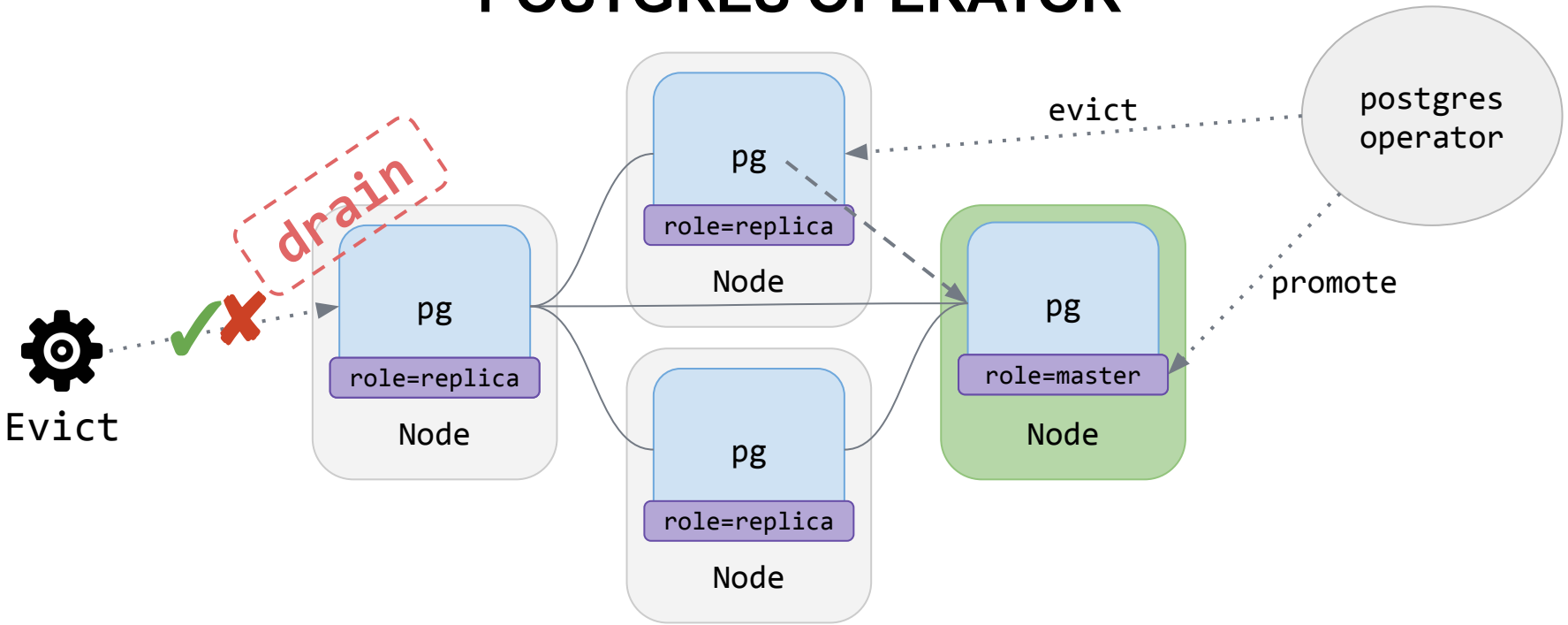
What about stateful applications like Postgres?



Postgres cluster unavailable :(

STATEFUL WORKLOADS (POSTGRES)

POSTGRES OPERATOR



github.com/zalando-incubator/postgres-operator

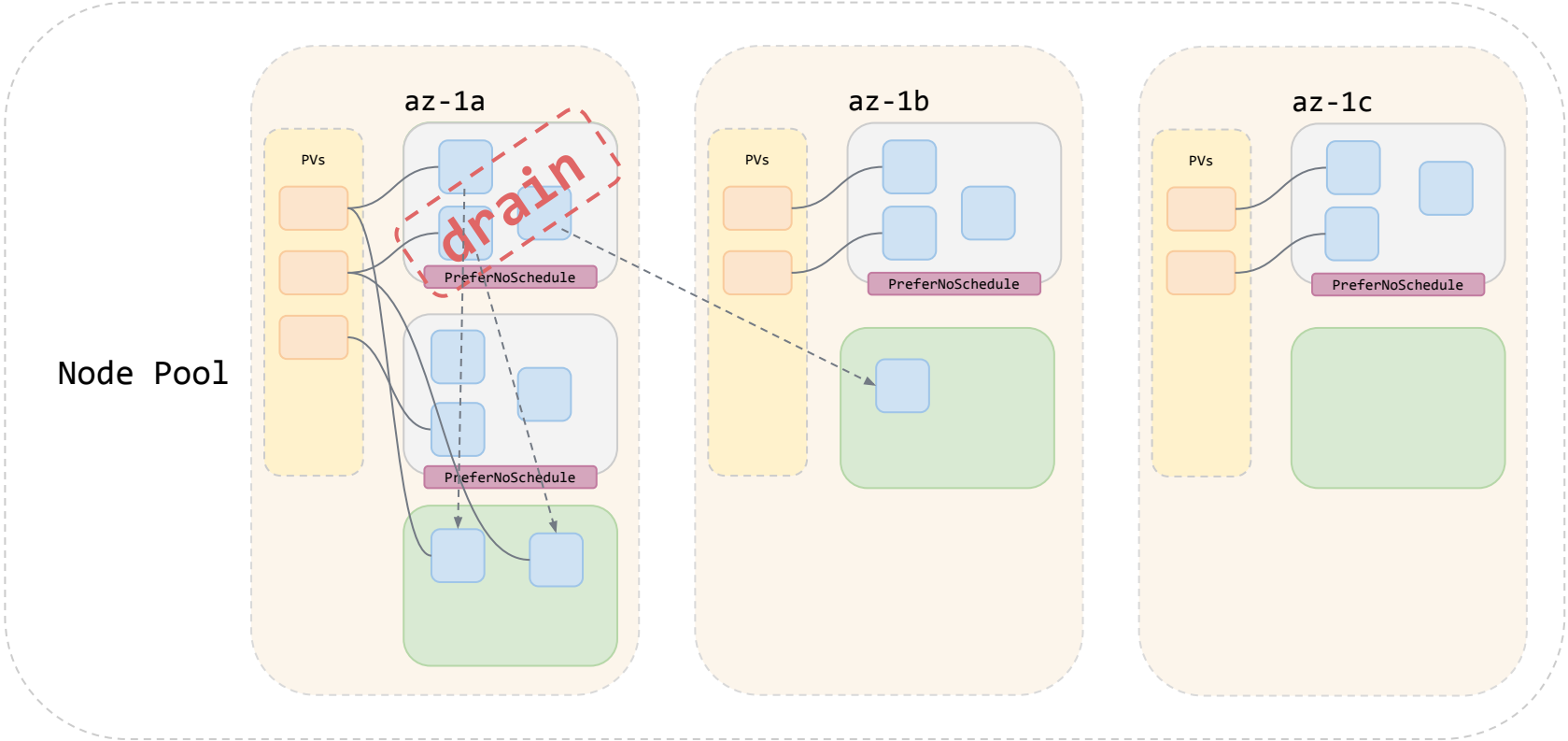
zalando

POSTGRES OPERATOR

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

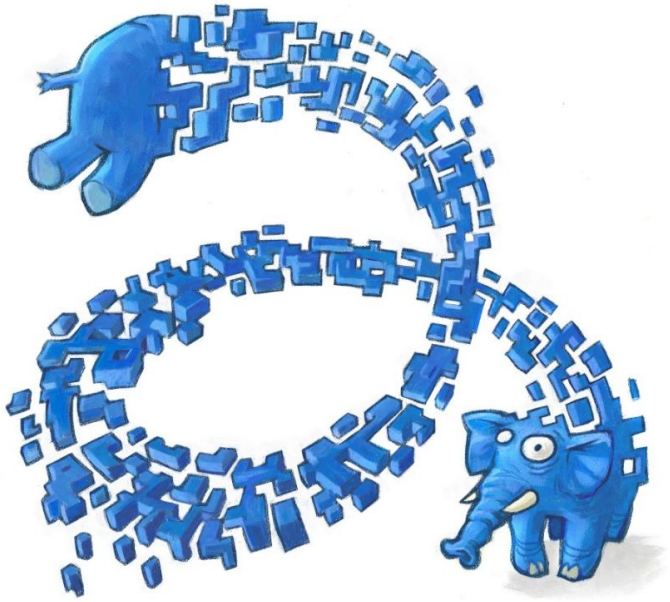
github.com/zalando-incubator/postgres-operator

ROLLING UPGRADE OF NODES



zalando

POSTGRES OPERATOR

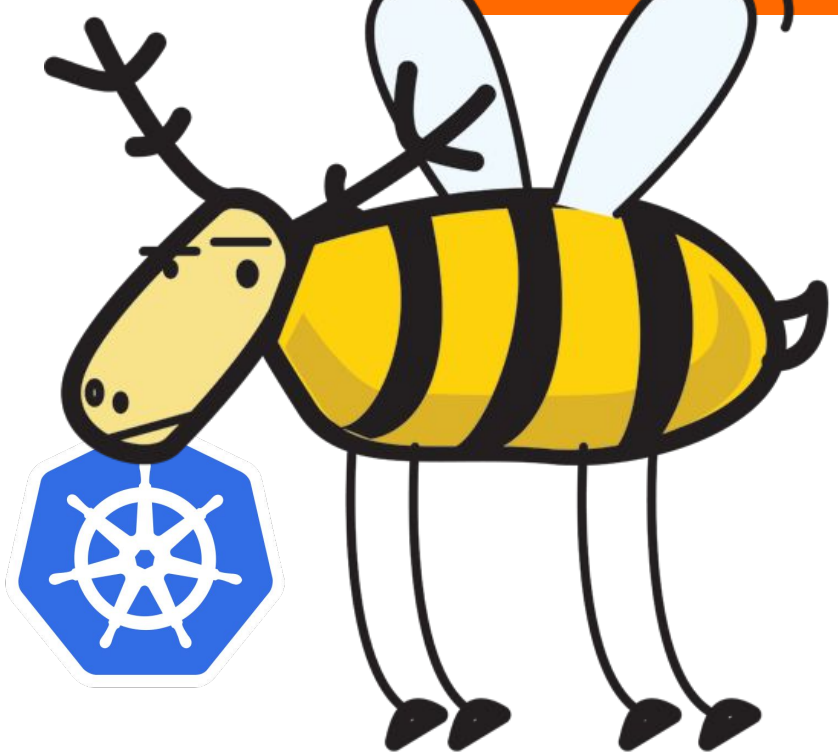


Application to manage PostgreSQL clusters on Kubernetes

>500

clusters running on Kubernetes

github.com/zalando/postgres-operator



Elasticsearch
2.500 vCPUs
1 TB RAM

Elasticsearch in Kubernetes

github.com/zalando-incubator/es-operator/

SLAS FOR CLUSTER UPDATES

- Respect **PodDisruptionBudgets**
- **Force-terminate Pods after 3 days** (or 8h on test)
- Cluster updates can be blocked anytime!

```
zkubectl cluster-update block [+ REASON]
```

DEPLOY & USER INTERFACE

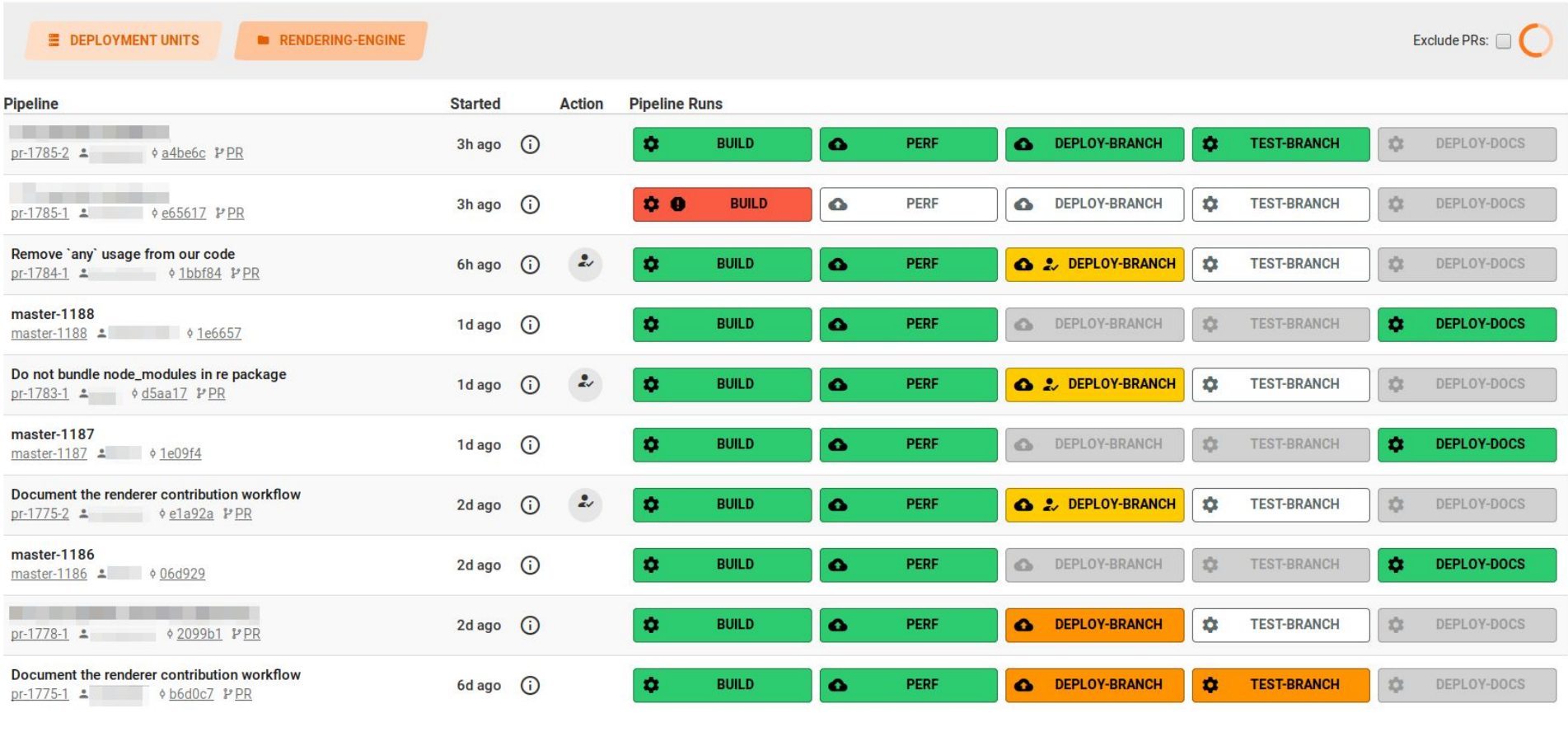
APP DEPLOYMENT CONFIGURATION

```
├── deploy/apply
│   ├── deployment.yaml
│   ├── credentials.yaml # Zalando IAM
│   ├── ingress.yaml
│   └── service.yaml
└── delivery.yaml        # Zalando CI/CD
```

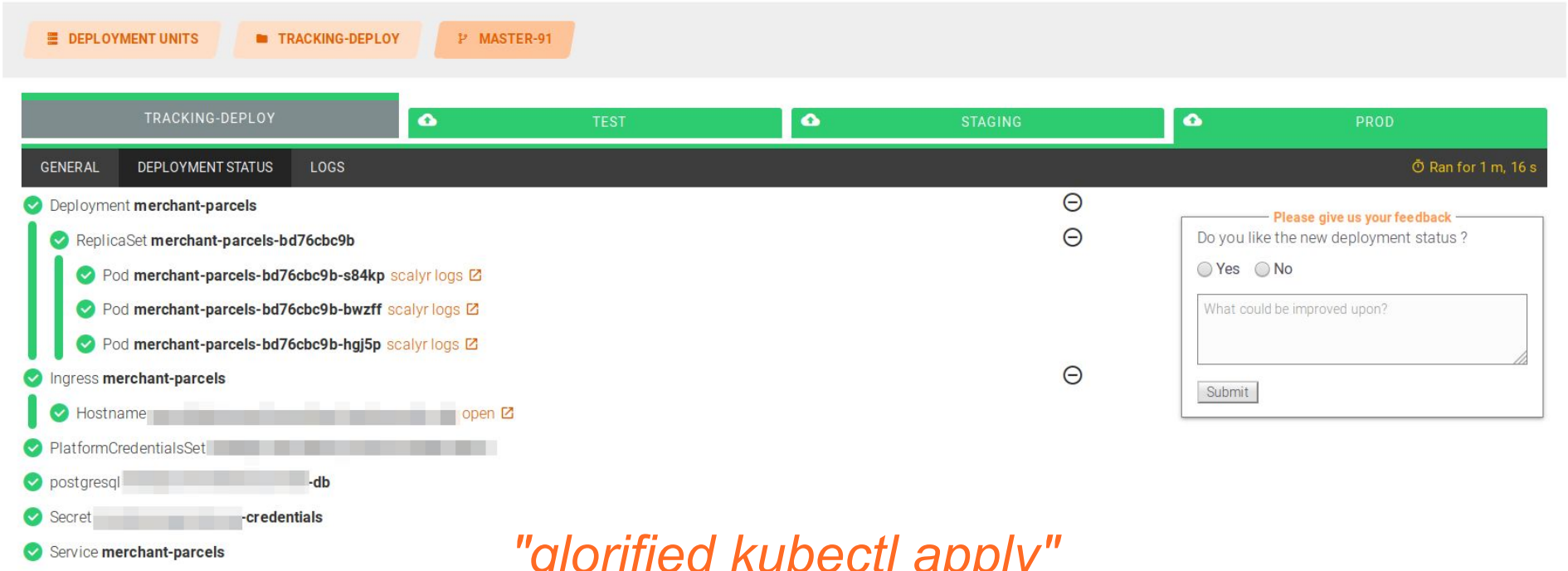
APP INGRESS.YAML

```yaml
kind: Ingress
metadata:
  name: "..."
spec:
  rules:
    # DNS name your application should be exposed on
    - host: "myapp.foo.example.org"
      http:
        paths:
          - backend:
              serviceName: "myapp"
              servicePort: 80
```

CONTINUOUS DELIVERY PLATFORM



CDP: DEPLOY



"glorified kubectl apply"

zalando

EMERGENCY ACCESS SERVICE

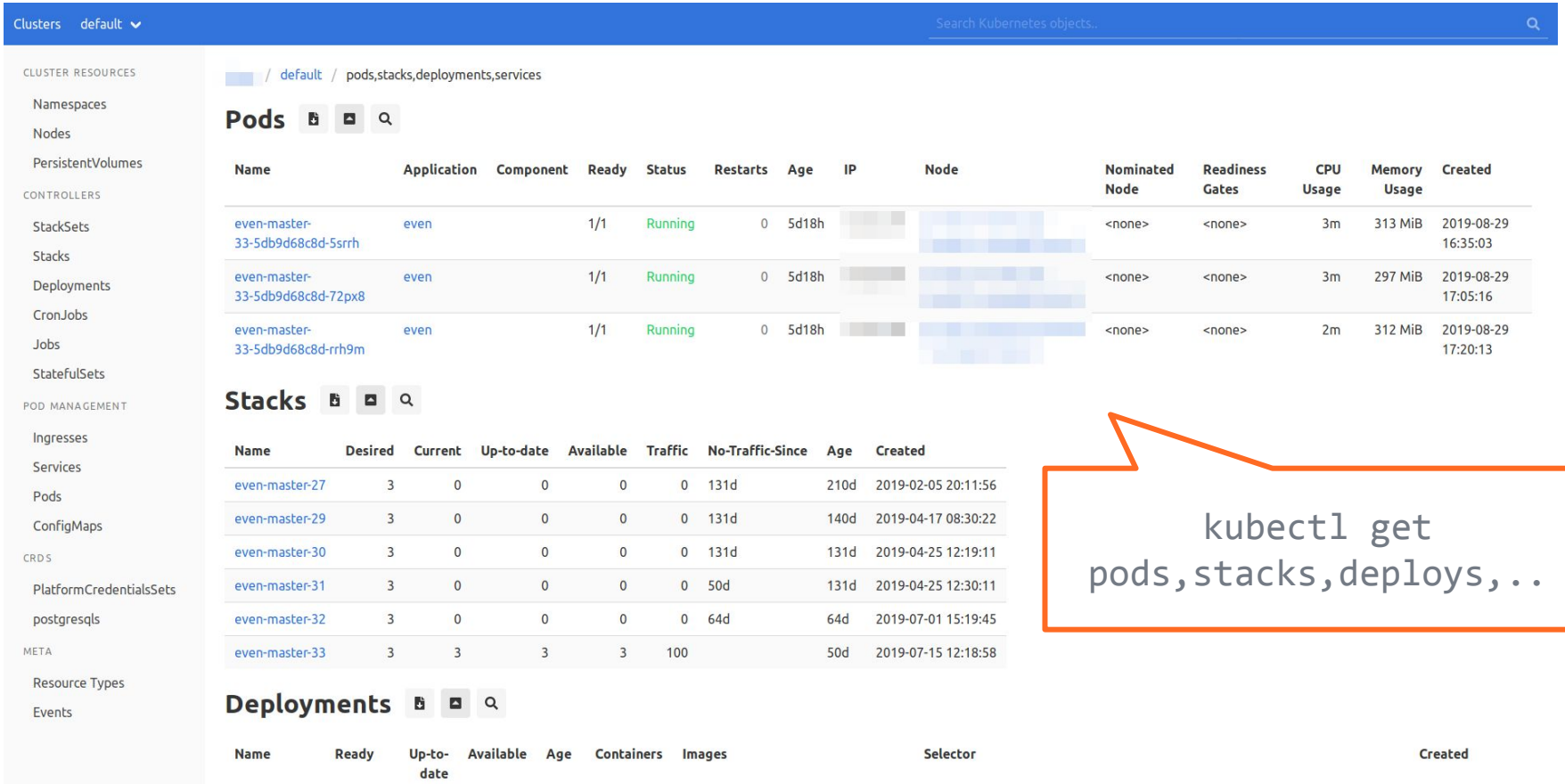
Emergency access by referencing Incident

```
zkubectl cluster-access request \
    --emergency -i INC REASON
```

Privileged production access via 4-eyes

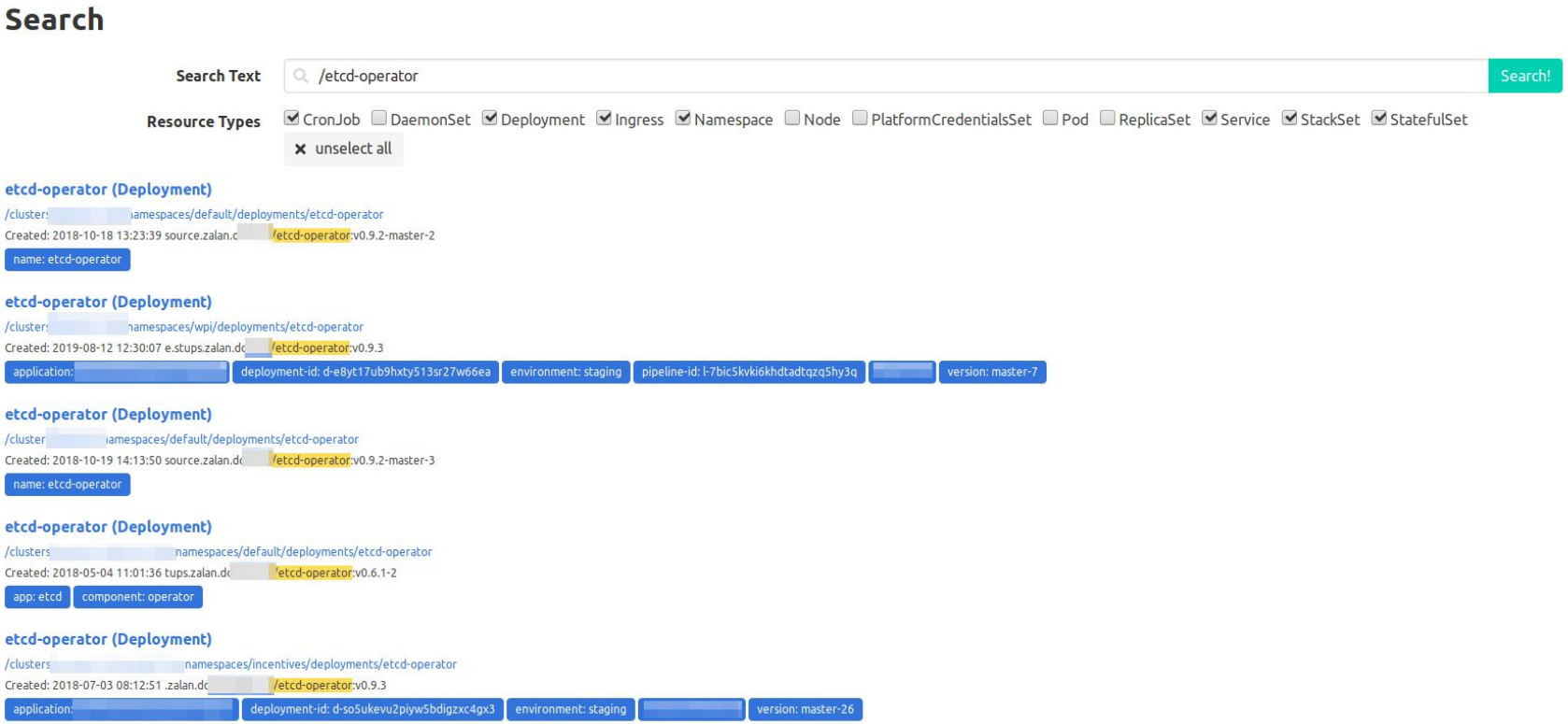
```
zkubectl cluster-access request REASON
zkubectl cluster-access approve USERNAME
```

KUBERNETES WEB VIEW



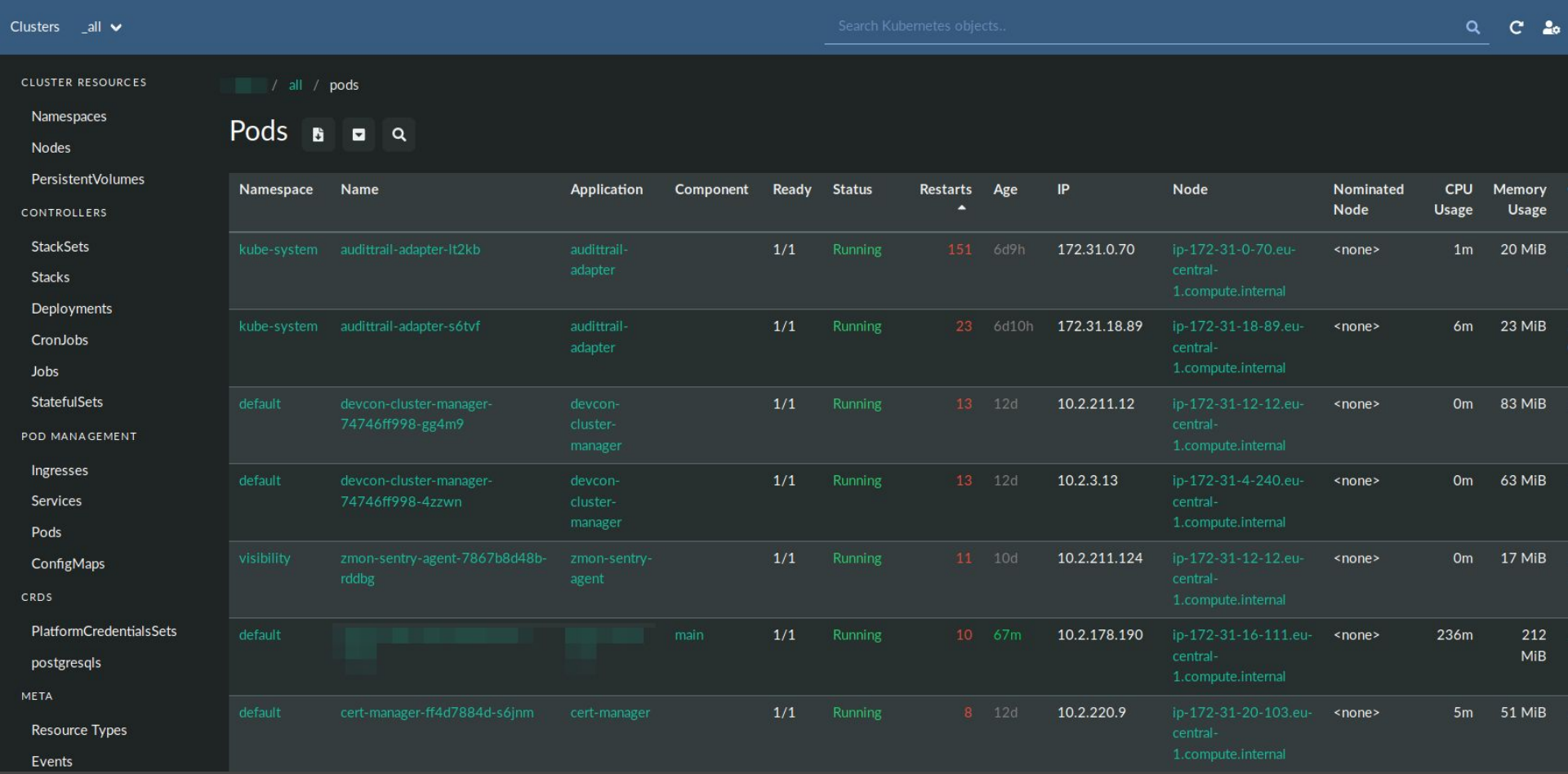
kubectl get pods,stacks,deploys,..

SEARCHING ACROSS 140+ CLUSTERS



codeberg.org/hjacobs/kube-web-view

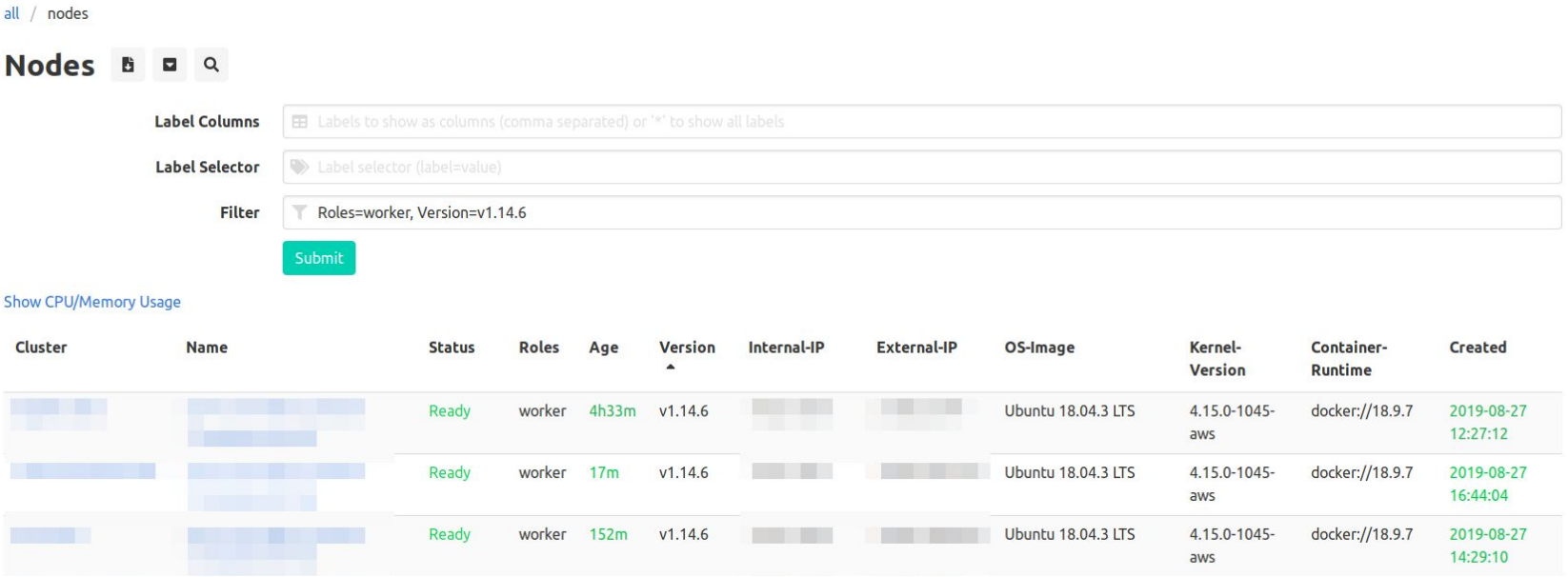
zalando



codeberg.org/hjacobs/kube-web-view

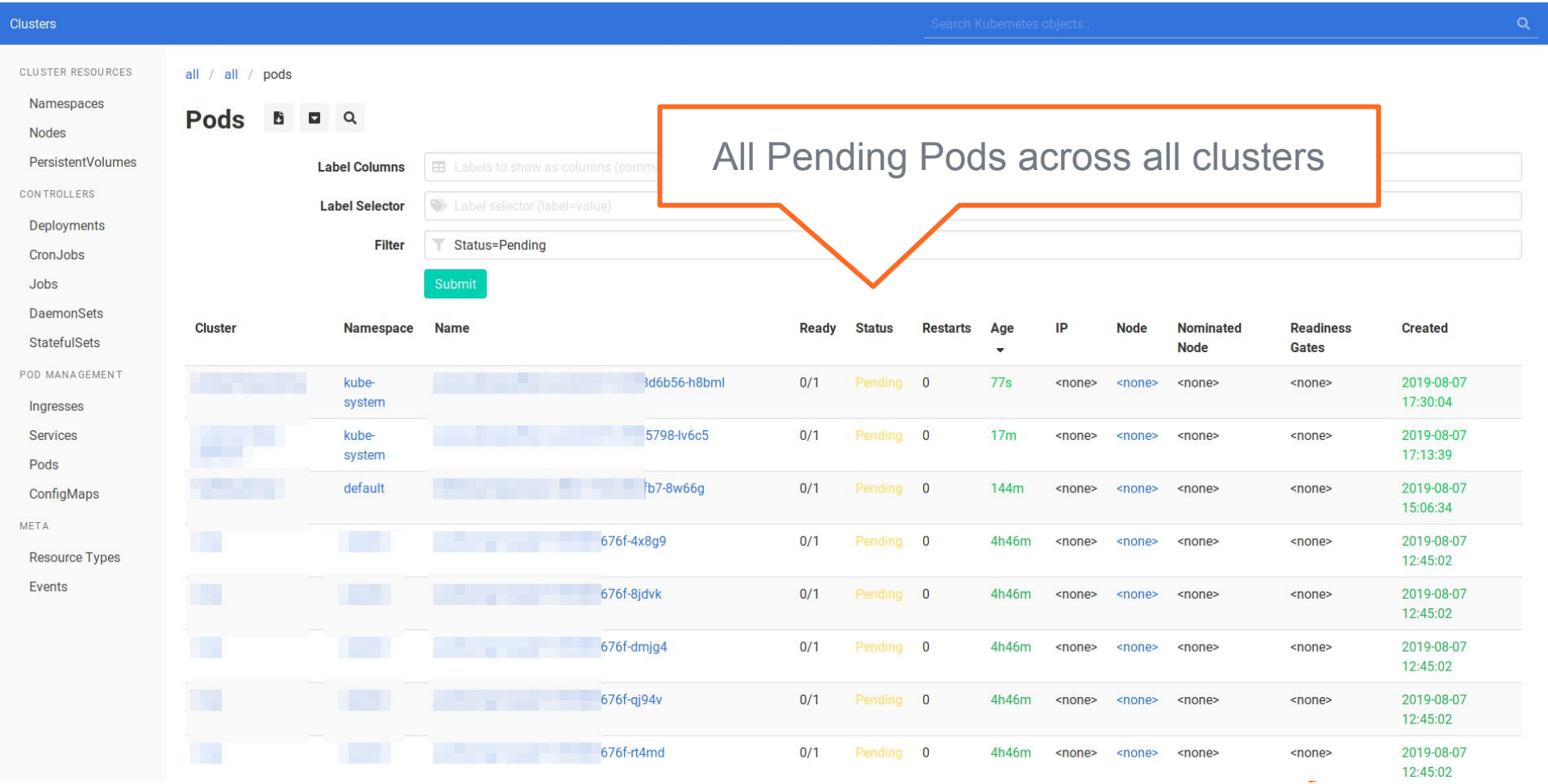
UPGRADE TO KUBERNETES 1.14

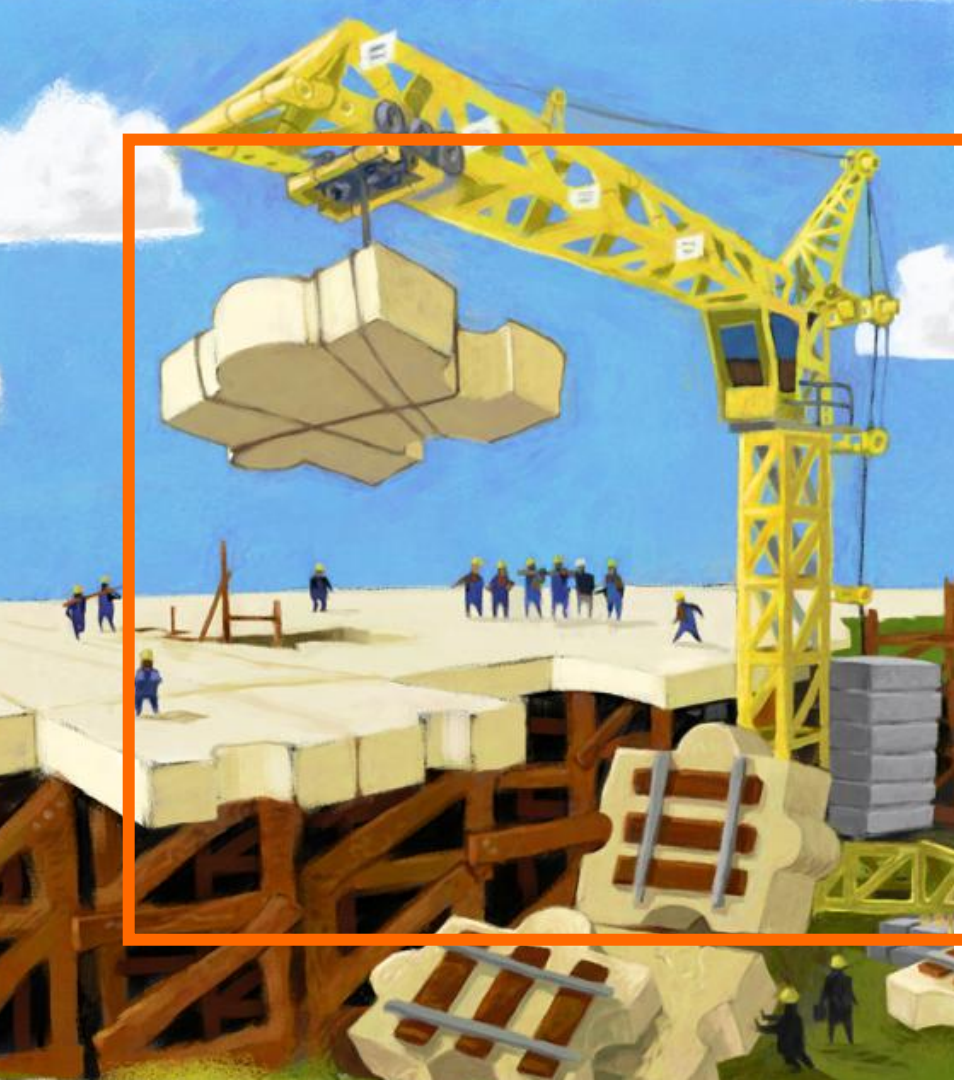
"Found 1223 rows for 1 resource type in 148 clusters in 3.301 seconds."



Cluster	Name	Status	Roles	Age	Version	Internal-IP	External-IP	OS-Image	Kernel-Version	Container-Runtime	Created
		Ready	worker	4h33m	v1.14.6			Ubuntu 18.04.3 LTS	4.15.0-1045-aws	docker://18.9.7	2019-08-27 12:27:12
		Ready	worker	17m	v1.14.6			Ubuntu 18.04.3 LTS	4.15.0-1045-aws	docker://18.9.7	2019-08-27 16:44:04
		Ready	worker	152m	v1.14.6			Ubuntu 18.04.3 LTS	4.15.0-1045-aws	docker://18.9.7	2019-08-27 14:29:10

zalando


Clusters
Search Kubernetes objects..
CLUSTER RESOURCES
Namespaces
Nodes
PersistentVolumes
CONTROLLERS
Deployments
CronJobs
Jobs
DaemonSets
StatefulSets
POD MANAGEMENT
Ingresses
Services
Pods
ConfigMaps
META
Resource Types
Events
all / all / pods
Pods
All Pending Pods across all clusters
Label Columns
Labels to show as columns (comm
Label Selector
Label selector (label=value)
Filter
Status=Pending
Submit
Cluster
Namespace
Name
Ready
Status
Restarts
Age
IP
Node
Nominated Node
Readiness Gates
Created
kube-system
3d6b56-h8bml
0/1
Pending
0
77s
<none>
<none>
<none>
<none>
2019-08-07 17:30:04
kube-system
5798-lv6c5
0/1
Pending
0
17m
<none>
<none>
<none>
<none>
2019-08-07 17:13:39
default
fb7-8w66g
0/1
Pending
0
144m
<none>
<none>
<none>
<none>
2019-08-07 15:06:34
676f-4x8g9
0/1
Pending
0
4h46m
<none>
<none>
<none>
<none>
2019-08-07 12:45:02
676f-8jdvk
0/1
Pending
0
4h46m
<none>
<none>
<none>
<none>
2019-08-07 12:45:02
676f-dmjg4
0/1
Pending
0
4h46m
<none>
<none>
<none>
<none>
2019-08-07 12:45:02
676f-qj94v
0/1
Pending
0
4h46m
<none>
<none>
<none>
<none>
2019-08-07 12:45:02
676f-rt4md
0/1
Pending
0
4h46m
<none>
<none>
<none>
<none>
2019-08-07 12:45:02



AVOIDING CONFIGURATION DRIFT

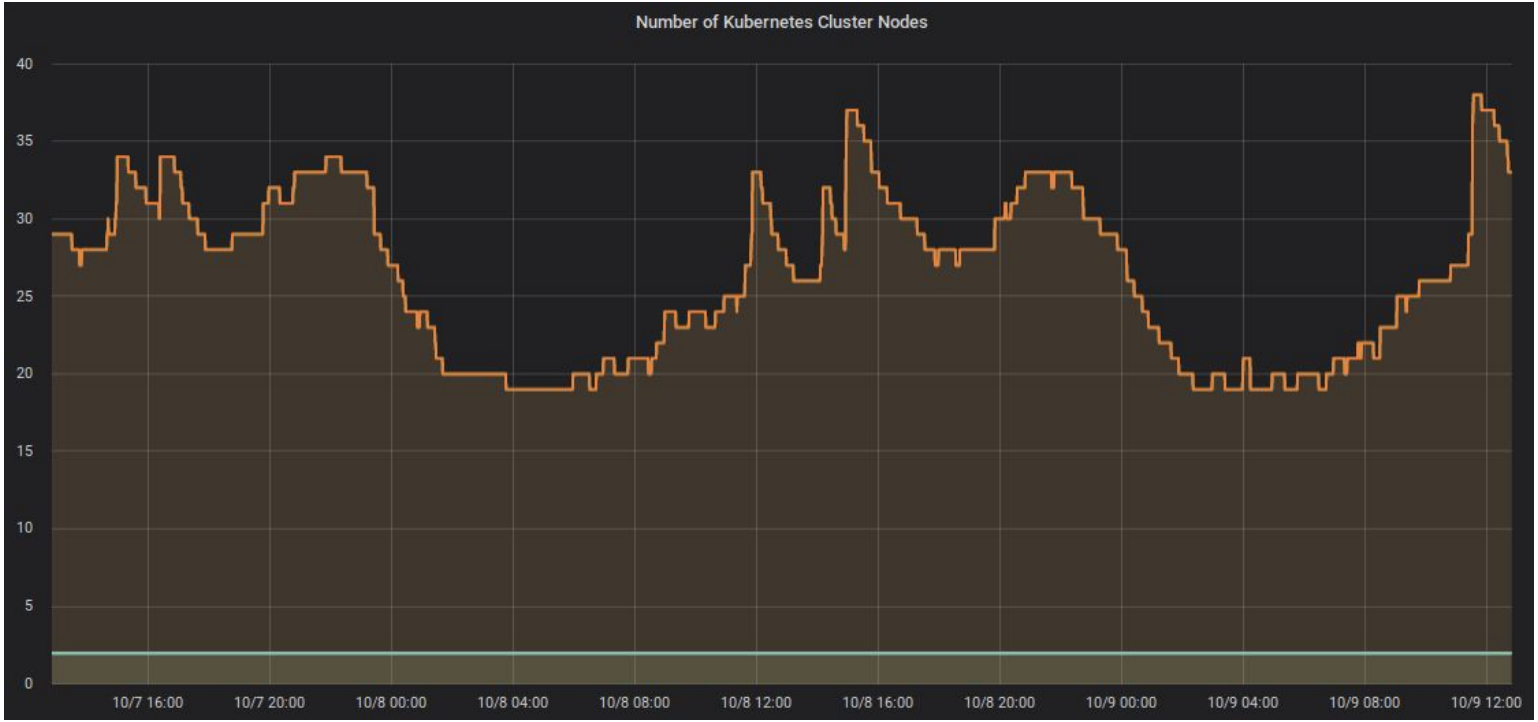
CLUSTER CONFIGURATION

Clusters look mostly the same, except:

- secrets, e.g. credentials for external logging provider
- node pools and their instance sizes

Cluster-specific config items are stored in Cluster Registry

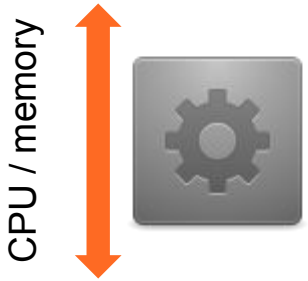
CLUSTER AUTOSCALER



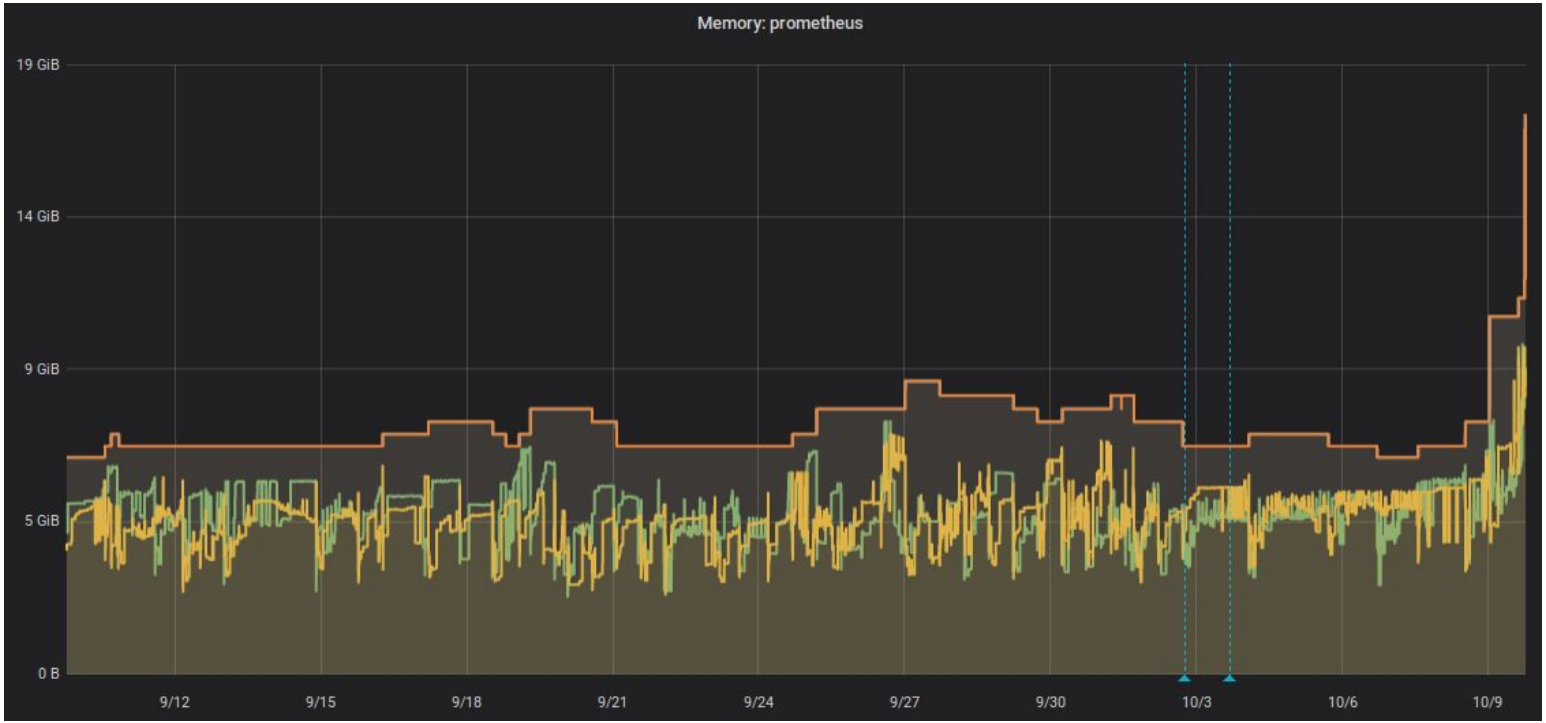
zalando

VERTICAL POD AUTOSCALER

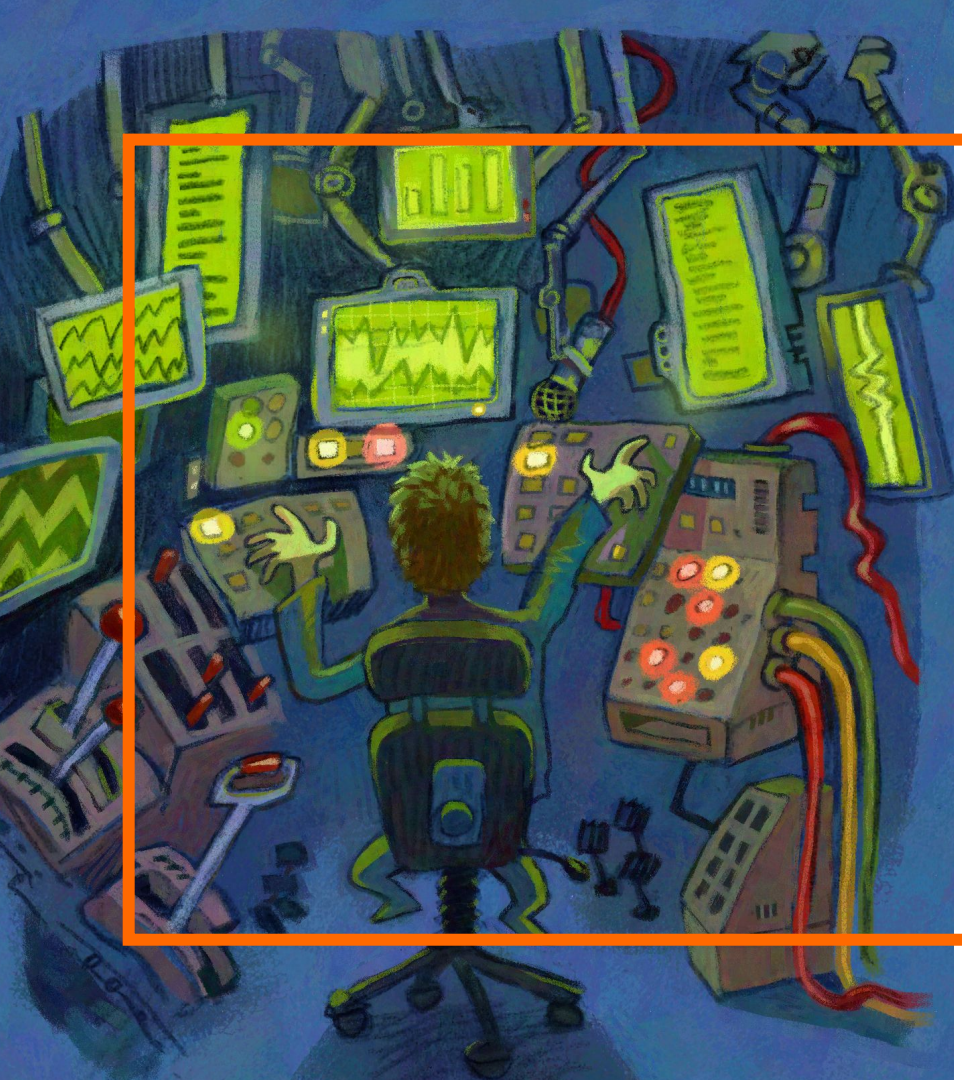
- Prometheus
- External DNS
- Heapster / Metrics Server
- our ALB Ingress Controller



VERTICAL POD AUTOSCALER



zalando



MONITORING & COST EFFICIENCY

MONITORING SYSTEM - ZMON

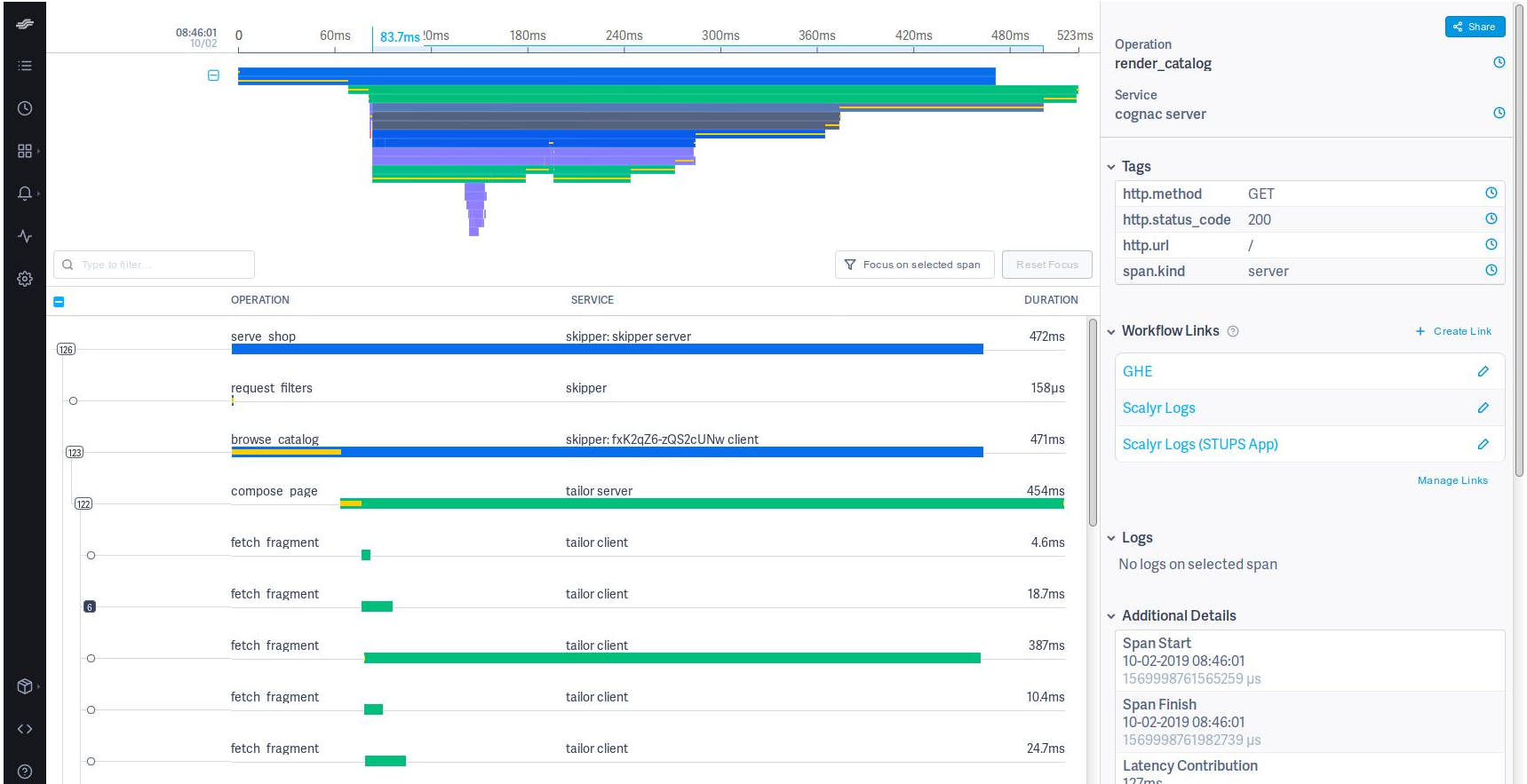
- Dynamic entity registration
  (clusters, pods, ..)
- Generic checks on entity attributes,
  e.g. for all production clusters
  "Less than 60% of worker nodes are ready"
- OpsGenie alerts

OpsGenie

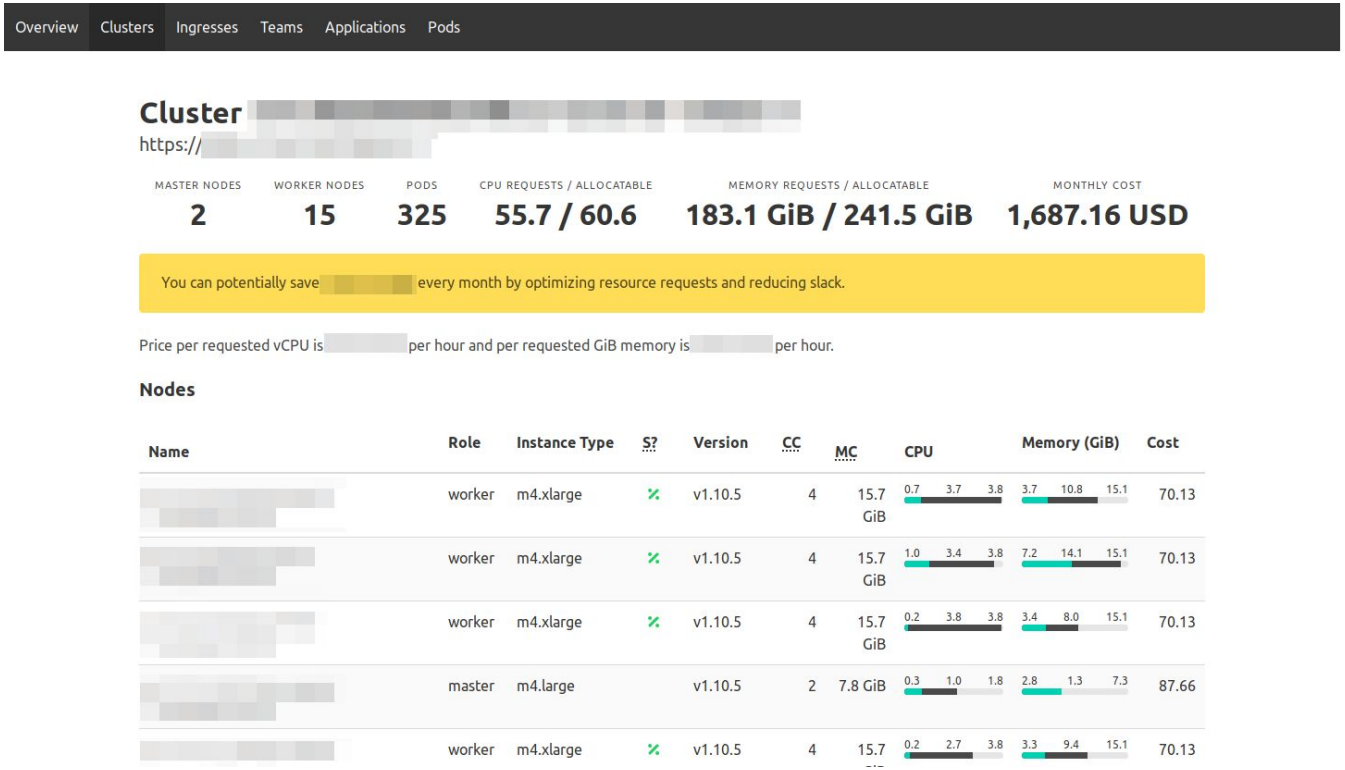
zalando

OPENTRACING

LIGHTSTEP



KUBERNETES RESOURCE REPORT



github.com/hjacobs/kube-resource-report

zalando

RESOURCE REPORT: TEAMS

ID	C	A	P	CR	MR	CPU	Memory (MiB)	Cost	Slack Cost
[redacted]	3	14	114	457.9	1.7 TiB	69.01 / 457.9	1,074,024 / 1,780,420	27,558.50	13,327.47
[redacted]	1	9	251	428.95	426.2 GiB	137.99 / 428.95	164,613 / 436,384	19,406.82	13,111.24

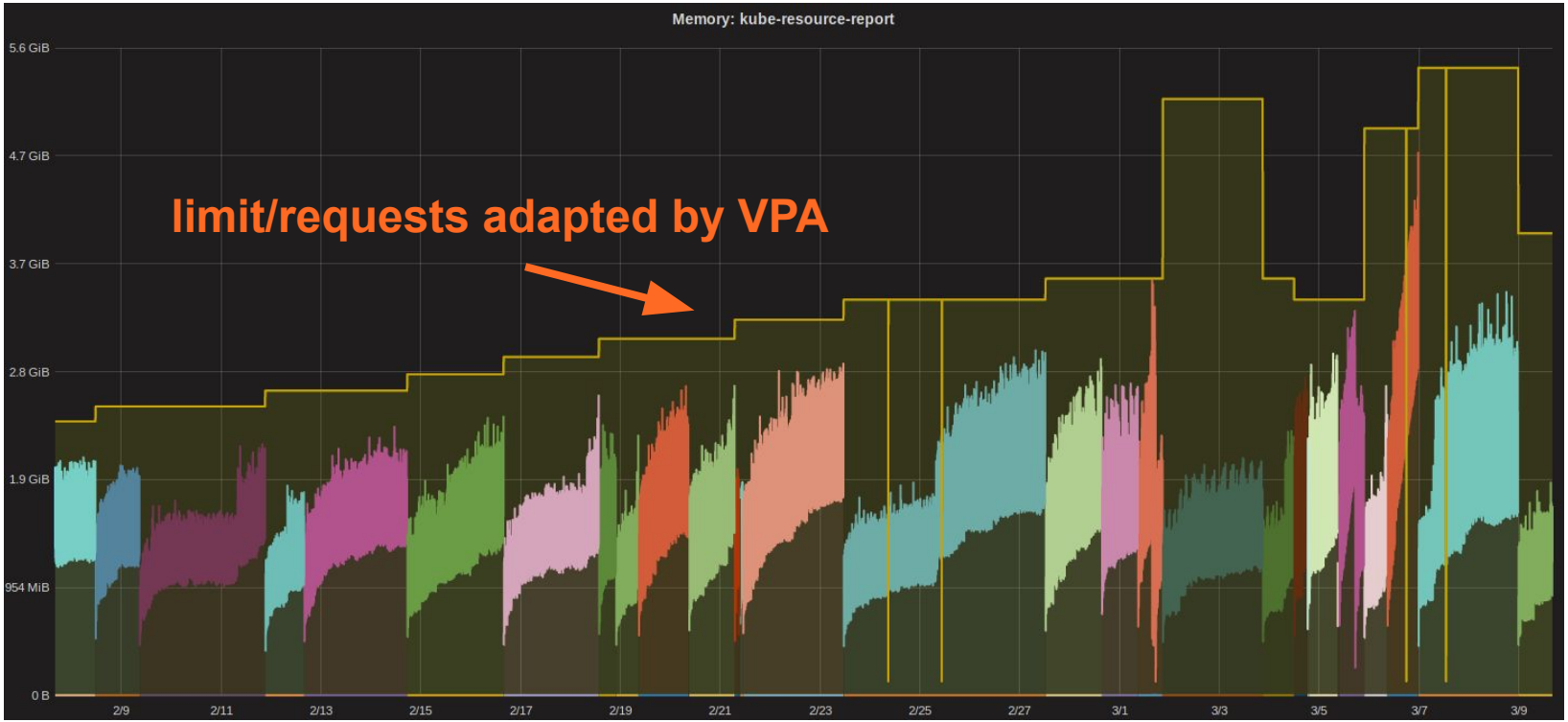
Sorting teams by Slack Costs

github.com/hjacobs/kube-resource-report

KUBERNETES APPLICATION DASHBOARD

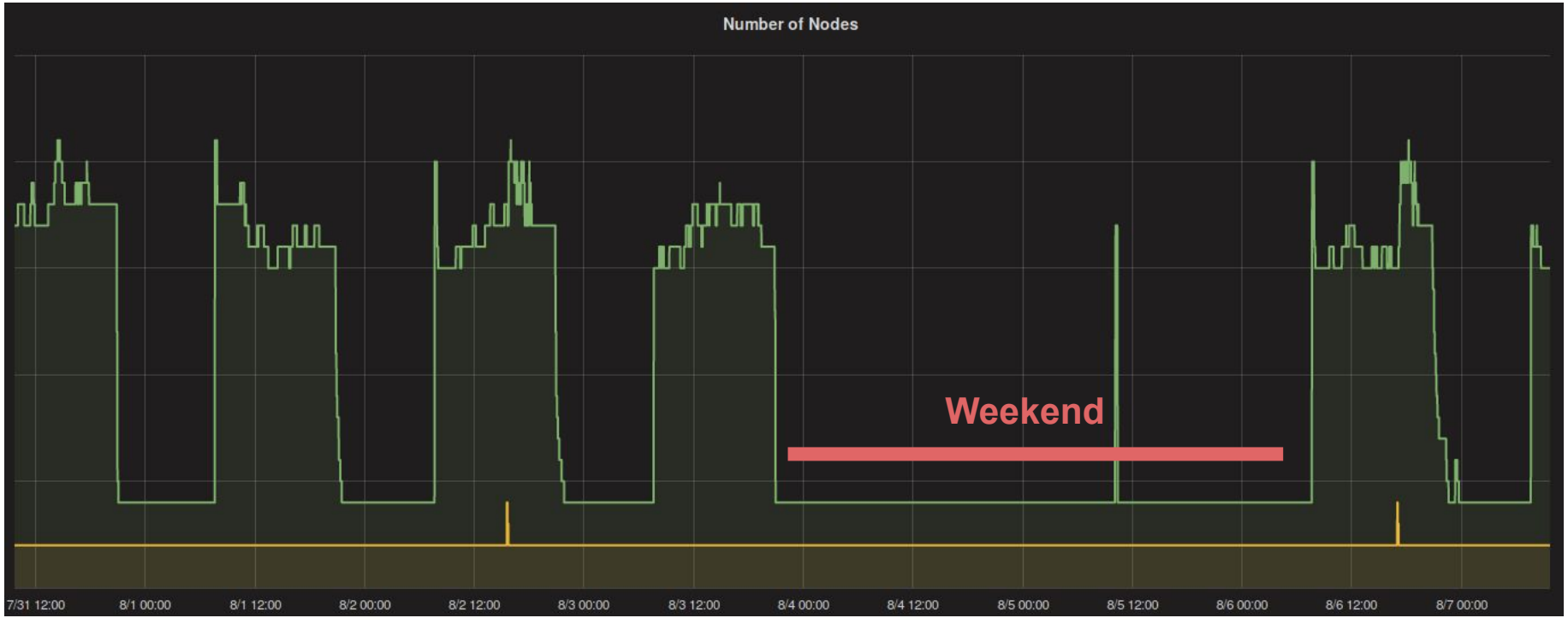
zalando

VERTICAL POD AUTOSCALER



zalando

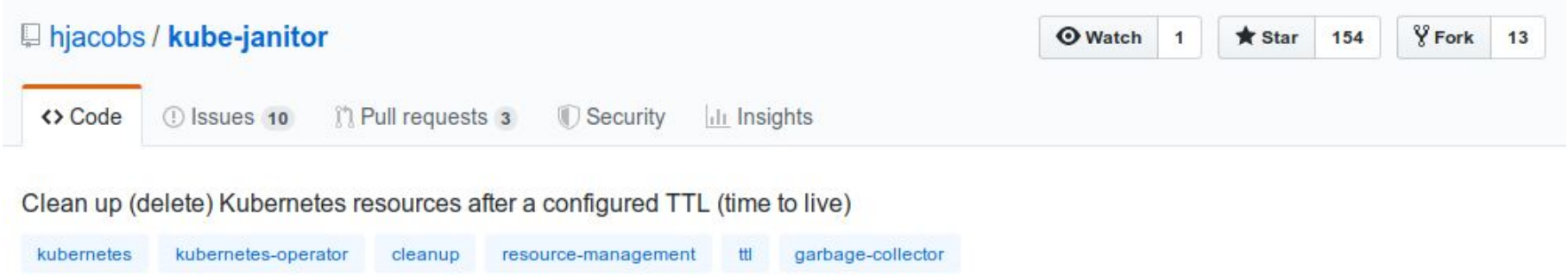
DOWNSCALING DURING OFF-HOURS



github.com/hjacobs/kube-downscaler

zalando

KUBERNETES JANITOR



- **TTL** and **expiry date** annotations, e.g.
  - set time-to-live for your test deployment
- **Custom rules**, e.g.
  - delete everything without "app" label after 7 days

github.com/hjacobs/kube-janitor

EC2 SPOT NODES

Role	Instance Type	S?	Version	CC	MC	CPU	Memory (GiB)	Cost
worker	m4.2xlarge		v1.12.5-custom.master-1	8	31.4 GiB	1.6 6.9 7.8	8.8 14.3 30.9	350.64
worker	m4.4xlarge	%	v1.12.5-custom.master-1	16	62.9 GiB	4.1 9.0 15.8	51.2 62.3 62.4	193.73

72% savings

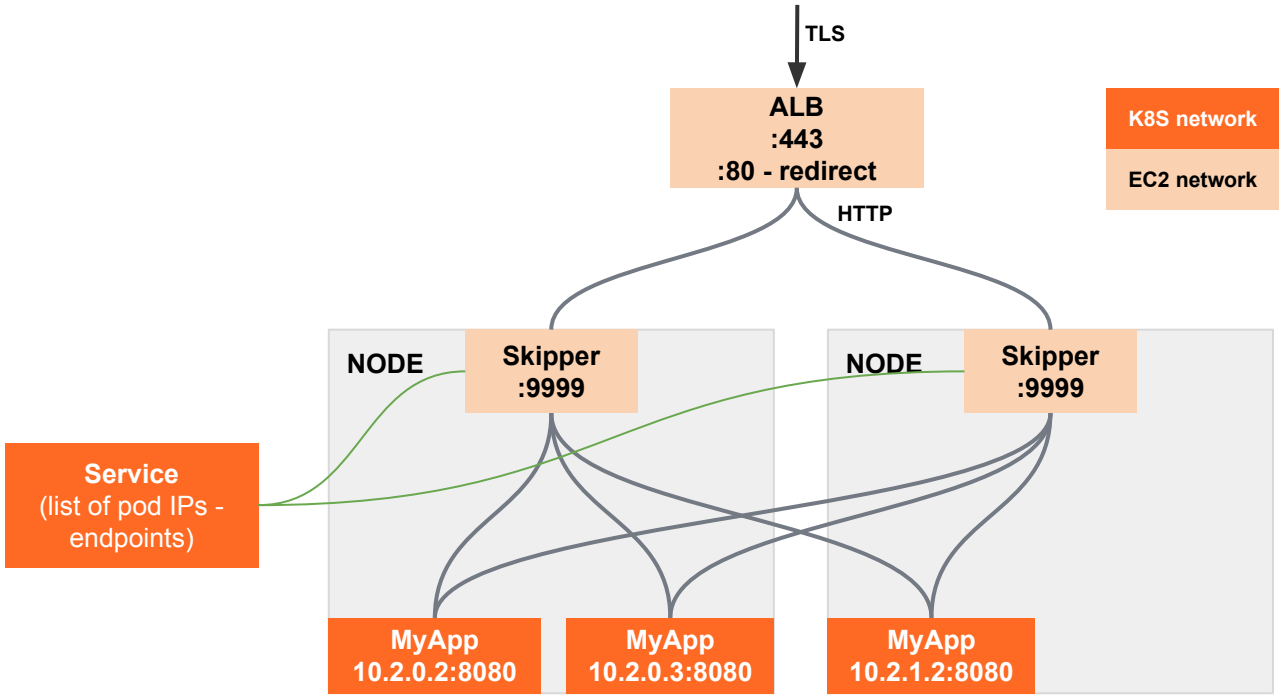
OUR SETUP VS VANILLA KUBERNETES

zalando

HOW MUCH DO WE DIVERGE?

- API access via Zalando OAuth
- CPU throttling disabled via Kubelet flag
- No memory overcommit (requests == limits)
- Ingress: External DNS, Skipper, AWS ALB
- Custom CRDs: Zalando OAuth, Postgres, StackSet
- Kubernetes Downscaler
- DNS setup (CoreDNS DaemonSet, ndots: 2)

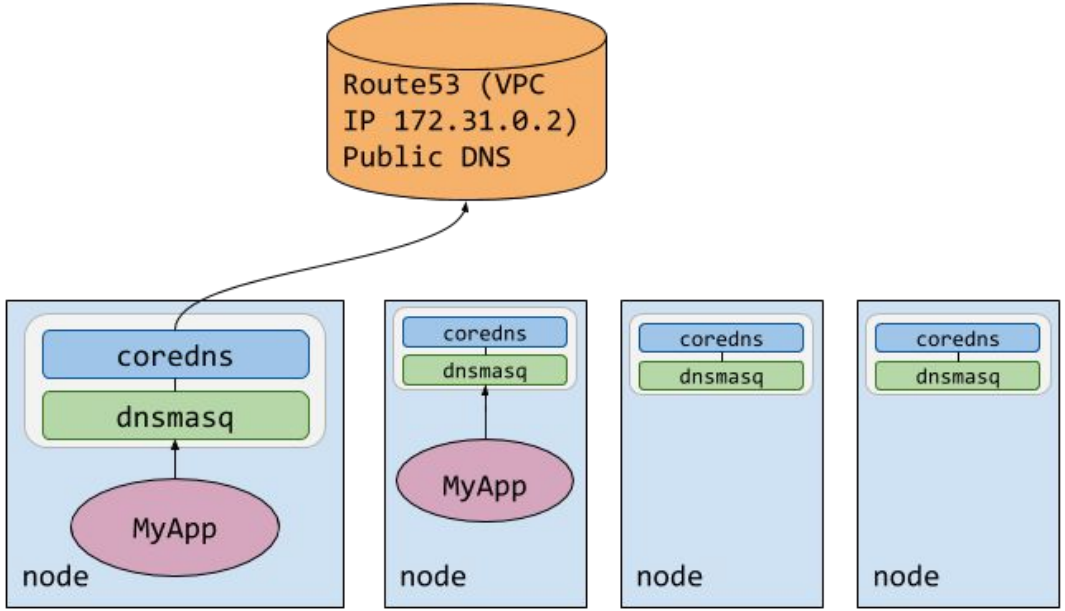
INGRESS: ALB + SKIPPER



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

zalando

DNS: COREDNS AS DAEMONSET



github.com/zalando-incubator/kubernetes-on-aws/blob/dev/docs/postmortems/jan-2019-dns-outage.md

NON-PROD VS PROD

- Non-production similar to plain hosted Kubernetes
- Production:
  - No write access (only via CI/CD)
  - Compliance webhooks
  - Require production-ready Docker images

COMPLIANCE FOR PRODUCTION

- Pods require `application` label pointing to application registry

  ⇒ establishes link to owning team

- Docker images must be built from master via CDP

NOTE: teams can freely choose their namespace(s)

HOWTOS

Home

Plan

Setup

Design

Code

Build

Test

Deploy

Autoscale Your Application

Configure deployments from PRs

Setting Resource Requests and Limits

Expose Application to Internet

Give Your Application Access to AWS

Give Your Application Access to Kubernetes

Migrate a STUPS Application

Run Applications in Production

Run Stateful Applications

Store Passwords in Deployment Configurations

Troubleshoot a Failing Deployment

Setting Resource Requests and Limits

Resource requests are specified per container in the Pod spec of your Kubernetes deployment, for example in the file `deployment.yaml`. Resource requests are used by the Kubernetes scheduler to decide if a Pod can fit on a node or not. It's important to choose the right amount of resources as it will provide better resource utilization in the clusters and prevent running with too much slack capacity which translates to wasted resources.

Correctly Configure Resource Requests and Limits

Existing Application

If you have an existing application running you can check that it's configured with sensible resource requests and limits by looking at the Application Dashboard. This dashboard will show the current CPU and Memory usage along with the specified requests and limits. Additionally it will indicate if the application is over committing on memory meaning `limits > requests`.

In this example the application has the `limits` set 10 times higher than the `requests`.

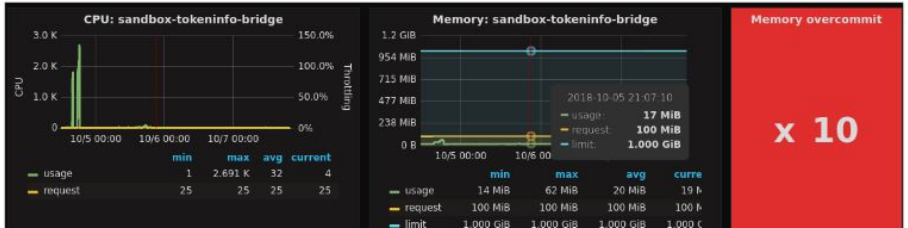


Table of contents

Correctly Configure Resource Requests and Limits

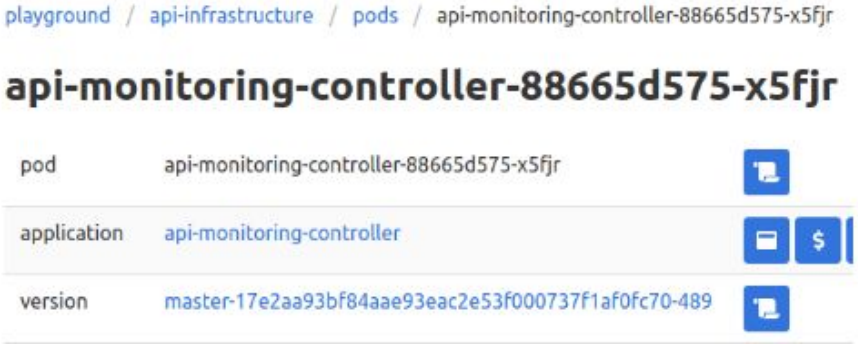
Existing Application

New Application

MONTHLY DEVELOPER NEWSLETTER

KUBERNETES

- **Priority support for node pools**: For Cyberweek we added support for having fallback node pools with different instance types and priority in order to provide capacity even when AWS runs out of the desired instance types.
  We will keep this setup going forward allowing better instance availability especially for scared instance types. You can request custom fallback pools! See the docs.
- **zkubectl tunnel** allows you to access non-public resources by setting up a series of port forwards through your Kubernetes cluster. For instance, you can use it to access protected ElastiCache and RDS databases from your local machine, e.g. to dump and restore their data. See the docs for other use cases and how to setup a tunnel.
- **Kube-Web-View**: You can now reach Scalyr, Lightstep, GitHub, CDP, Pier One, Postgres, the API Portal and more from Kube Web View:



zalando

SUMMARY

- Seamless updates
- Avoid pet clusters
- Small disruptions are normal
- Automated cluster e2e tests
- Documentation & communication

FUTURE

- API version updates (1.16+)
- Improved Autoscaling
- Improved StackSet, Gradual Rollout
- Migrations
- Cost efficiency
- Looking at VPC CNI, AWS IAM, EKS, ...

KUBERNETES FAILURE STORIES

- Zalando's Failure Stories - KubeCon EU 2019
- Build Errors of Continuous Delivery Platform
- Total DNS outage in Kubernetes cluster

https://k8s.af

COMMON PITFALLS

- Insufficient e2e tests
- Readiness & Liveness Probes
- Resource Requests & Limits
- DNS

OPEN SOURCE & MORE

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

Skipper HTTP Router & Ingress controller
github.com/zalando/skipper

Ingress Controller for AWS
github.com/zalando-incubator/kube-ingress-aws-controller

Kubernetes Web View
codeberg.org/hjacobs/kube-web-view

More Zalando Tech Talks
github.com/zalando/public-presentations

zalando

Thank you!

Henning Jacobs
@try_except_