Resurs Bank

From Zero to Hero with OpenShift

Red Hat Forum 2019-10-03, Stockholm

Jimmy Falkbjer, Principal Architect Software

Resurs Bank

Introduction

- Jonas Björk, Operations
- Kristian Ejvind, Operations
- Mathias Åhsberg, System Developer

- Pernilla Lundqvist, Project Manager
- Niclas Tegnér, System Developer

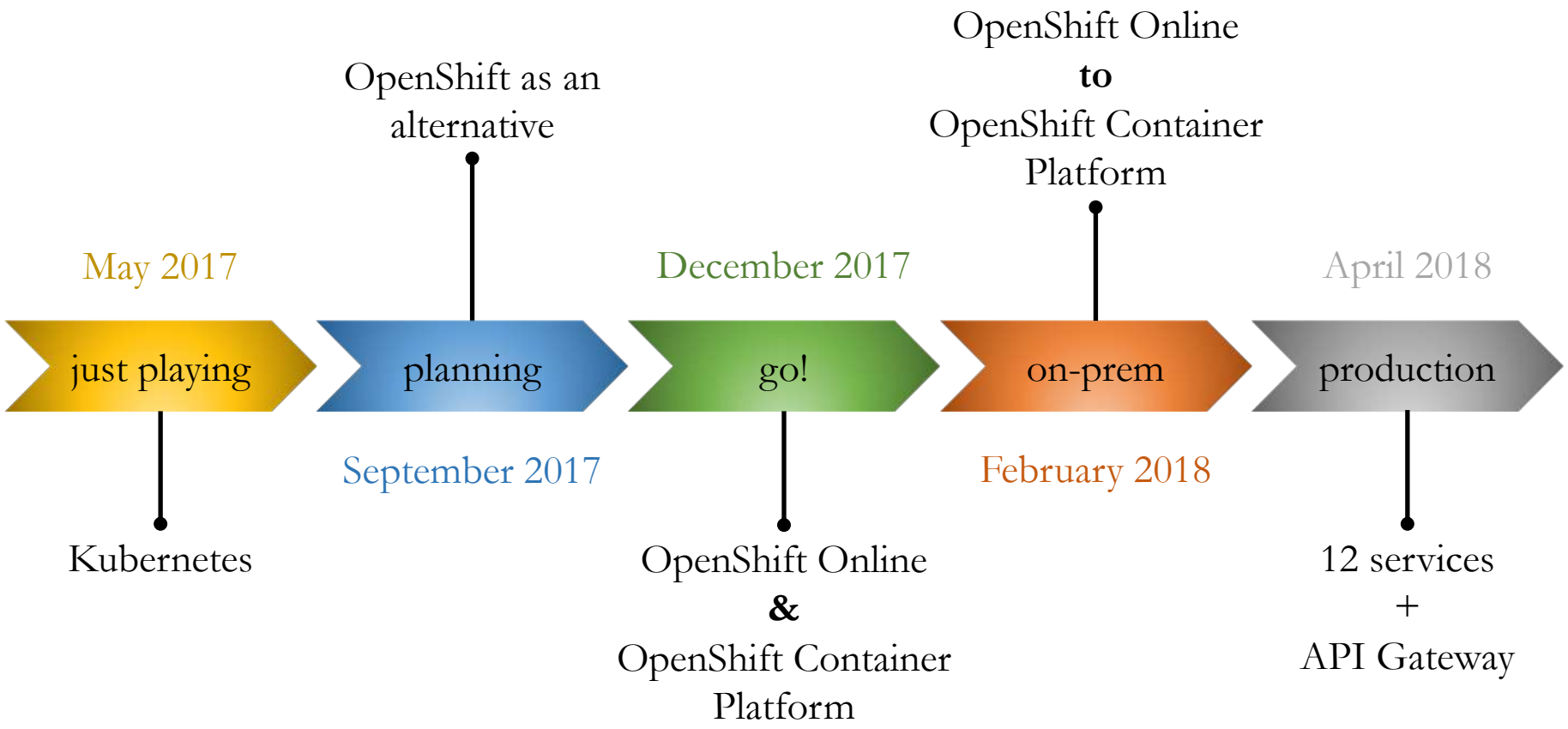
Resurs Bank

Goals for the new Environment

- Containers, of course
- Secure
  - Services
  - Data
- Clear separation of concerns
  - Secrets
  - Keep up software/frameworks upgrades
- No more monoliths
- Automate as much as possible
  - Build and deploy pipeline
  - Deploy to production during working hours

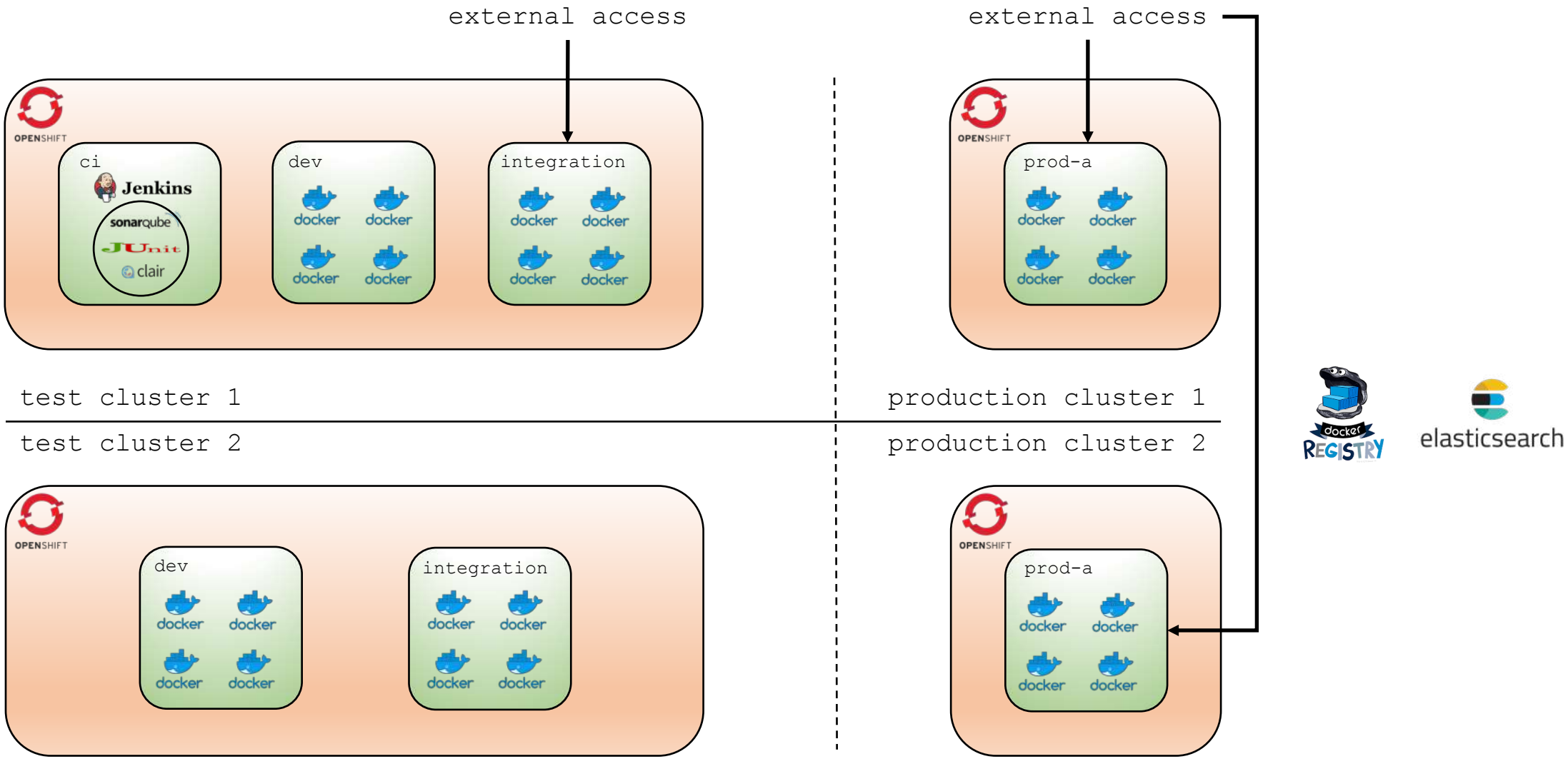
Resurs Bank

Road to production



Resurs Bank

What did we have



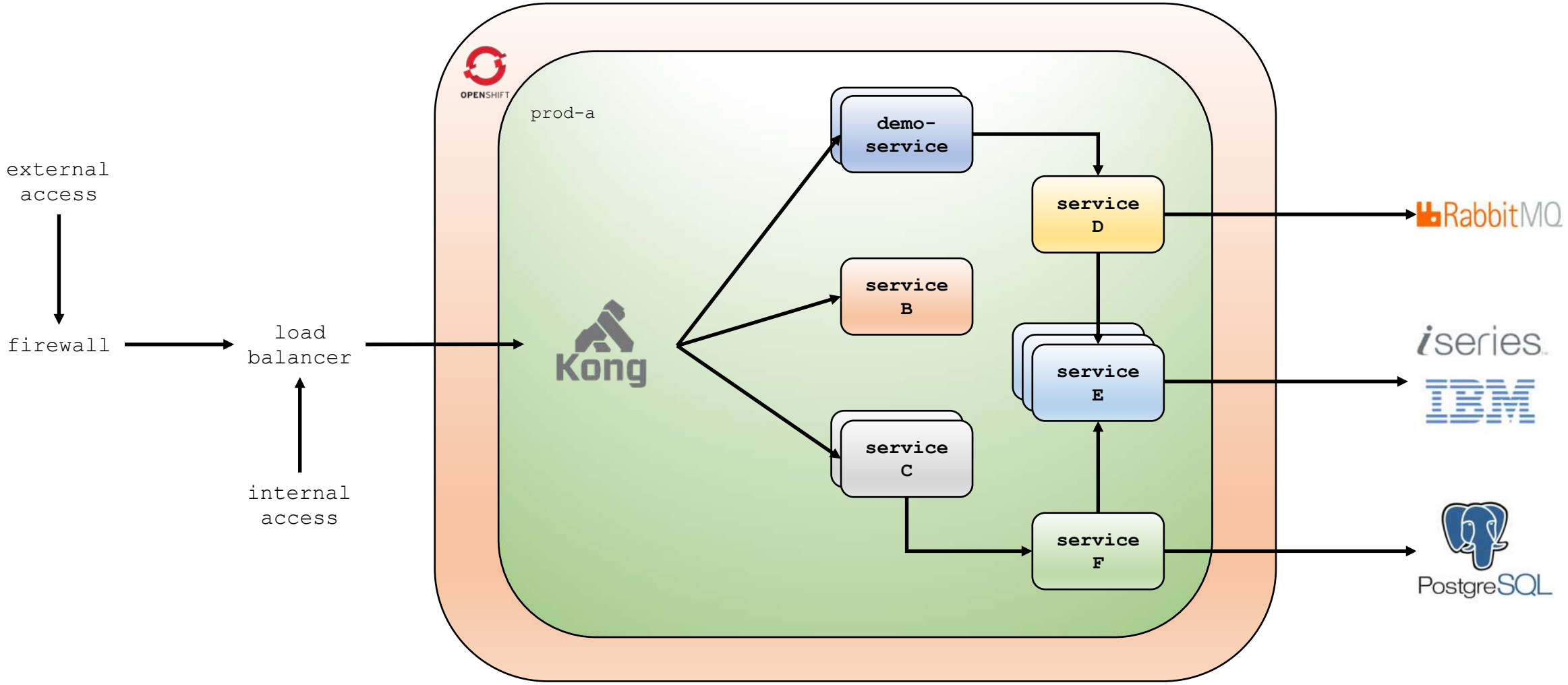
Resurs Bank

Environment

- OpenShift 3.7 on VMware, with two compute/storage tiers per cluster
- Persistent storage using vSphere cloud provider
- One common Elasticsearch instance for all clusters
- One common Openshift Standalone Registry for all clusters

Resurs Bank

Production



Resurs Bank

Kong

- Øredev, 2016
  - Jeremy Seitz - Domain-Driven Desire: API architecture, cross-functional teams, and war stories around the topic *https://vimeo.com/191051851*
  - *. . . based on NGINX, it is Open Source . . .*
- Plugin based
  - Key Authentication; API key for consumers
  - IP Restriction; configure service for internal use
  - Prometheus; expose metrics
  - Zipkin; distributed tracing
  - Pre/Post-function; exchange client certificate
  - and some more . . .

Resurs Bank

Kong configuration

- Configuration via curl commands

```
curl -X POST "http://localhost:8001/services/" \
  -d "name=demo-service" \
  -d "url=http://demo-service:8080/api"

curl -X POST "http://localhost:8001/services/demo-service/routes" \
  -d "paths[]=/api/demo_service" \
  -d "strip_path=true"

curl -X POST "http://localhost:8001/services/demo-service/plugins" \
  -d "name=prometheus"
```

- From version 0.12.2 to 1.3.0, via 9 upgrades, with no downtime

Resurs Bank

Kong future, perhaps

- Remove curl commands
- Use declarative configuration
  - All configuration in yaml format
  - `oc apply/delete`
- Replace HAProxy with Kong Ingress Controller

Resurs Bank

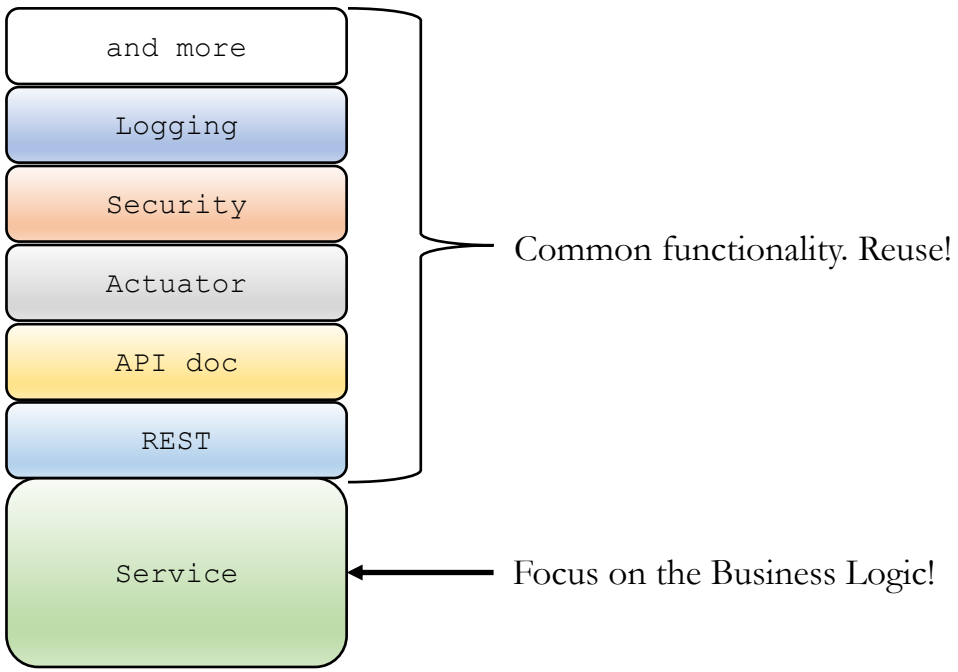
Service definition

- Endpoints under /api
- OpenAPI spec under /api_docs
- Distributed tracing conforming to OpenTracing standard
- Aggregated application logging to Resurs Bank central log in UTC timestamps
- Validation of JWT authorization header when applicable
- Check required authorities for endpoints when applicable
- Health endpoint (200 OK)
- Metrics endpoint (prometheus format)
- Secrets configured by Operations in Production
- Follow endpoint/naming/coding conventions
- Common error response structure with traceId
- README.md
- Jenkinsfile
- Dockerfile

Resurs Bank

Spring Boot Service with Starters

- REST
  - /api
- API documentation
  - /api_docs
- Actuator
  - /actuator/*
- Security
  - JWT
  - Authorities
- Logging
- Error handling
- and more . . .



Resurs Bank

Base images by Operations

- RHEL base image
  - resurs-minimal:latest

```
FROM registry.access.redhat.com/rhel7-minimal:latest
RUN yum -y upgrade
COPY Resurs-RootCA.crt /etc/pki/ca-trust/source/anchors
```

- Java base image
  - resurs-minimal-java11:latest

```
FROM base/resurs-minimal:latest
RUN microdnf update && \
    microdnf --enablerepo=rhel-7-server-rpms install java-11-openjdk-headless --nodocs && \
    microdnf clean all
```

Resurs Bank

Building service image

- demo-service:b141d0b

```
FROM base/resurs-minimal-java11:latest
ADD demo-service/build/libs/demo-service.jar /app.jar

RUN sh -c 'touch /app.jar'

EXPOSE 8080

ENV JAVA_OPTS="-Djava.security.egd=file:/dev/./urandom"
ENV JAVA_MEM_OPTS=""

ENTRYPOINT ["sh", "-c", "java $JAVA_OPTS $JAVA_MEM_OPTS -jar /app.jar"]
```

Resurs Bank

Configuration, take 1

- All configuration in git, including secrets
- Developers create required .yaml files

Resurs Bank

application.yml, in the source project

```yaml
spring.application.name: demo-service
client:
  connect-timeout: 3000
  read-timeout: 15000

---
spring.profiles: openshift

another-service.url: http://another-service:8080/api
datasource.host: postgres
resurs.cloud.secret.directories: /etc/demo-service/secrets
```

Resurs Bank

postgres.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: postgres
  namespace: prod-a
spec:
  type: ExternalName
  externalName: postgres.resurs.loc
```

Resurs Bank

configmap.yaml

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: demo-service
  namespace: prod-a
  labels:
    app: demo-service
data:
  application.yaml: |
    spring.profiles: openshift
    client:
      connect-timeout: 2000
      read-timeout: 10000
```

Resurs Bank

deployment.yaml

```yaml
apiVersion: apps/v1beta1
kind: Deployment
      - env:
        - name: SPRING_PROFILES_ACTIVE
          value: openshift
        - name: SPRING_CONFIG_LOCATION
          value: classpath:application.yml,file:/etc/demo-service/config/application.yaml
        volumeMounts:
        - mountPath: /etc/demo-service/config
          name: demo-service
      volumes:
      - configMap:
          items:
          - key: application.yaml
            path: application.yaml
          name: demo-service
        name: demo-service
```

Resurs Bank

service.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: demo-service
  namespace: prod-a
  labels:
    app: demo-service
spec:
  selector:
    app: demo-service
  ports:
    - name: http
      protocol: TCP
      port: 8080
      targetPort: 8080
  type: ClusterIP
```

Resurs Bank

Secret

- Operations
  - Can see and reveal Secret
- Developers
  - Cannot see Secret

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: demo-service-secrets
  namespace: prod-a
data:
  db.username: UmVhbGx5Pz8/Cg==
  db.password: Tm90aGluZ0hlcmUuCg==
```

- But we wanted version control of our secrets!

Resurs Bank

SealedSecret

- SealedSecrets by Bitnami
  - Kubernetes controller
  - Anyone can encrypt
  - Only the controller can decrypt
- Operations
  - Create SealedSecret
  - Add SealedSecret to Git
- Developers
  - Can see the SealedSecret

```yaml
apiVersion: bitnami.com/v1alpha1
kind: SealedSecret
metadata:
  name: demo-service-secrets
  namespace: prod-a
spec:
  encryptedData:
    db.username: AgBE7kmxlfs0gbXZZ…
    db.password: AgAZ8Qz3J2krU2V8H…
```

- We have version control of our secrets!

Resurs Bank

Configuration, take 2

```
demo-service
└── kubernetes
    └── openshift
        ├── application-dev.yaml
        ├── application-integration.yaml
        ├── application-prod.yaml
        └── deployment-config.json
```

Resurs Bank

deployment-config.json

```
{
    "template": "springboot",
    "environments": {
        "base": {
            "labels": {
                "team": "softwaresolutions"
            },
            "kong": {
                "expose": "public"
            },
            "secrets": [{
                "name": "demo-service-secrets",
                "mountPath": "/etc/demo-service/secrets"
            }]
        },
        "dev": {
            "replicas": 1,
            "cpuLimit": 1
        },
        "integration": {},
        "prod": {}
    }
}
```

Resurs Bank

generated-deployment-list.yaml

- ConfigMap
  - application-<environment>.yaml
- Service
  - labels
  - selector
  - namespace
  - type
  - Ports
- *API Gateway configuration*
  - *curl commands*
- Deployment
  - labels
  - namespace
  - affinity
  - image
  - env
  - ports
  - livenessProbe/readinessProbe
  - resources
  - volumeMounts

Resurs Bank

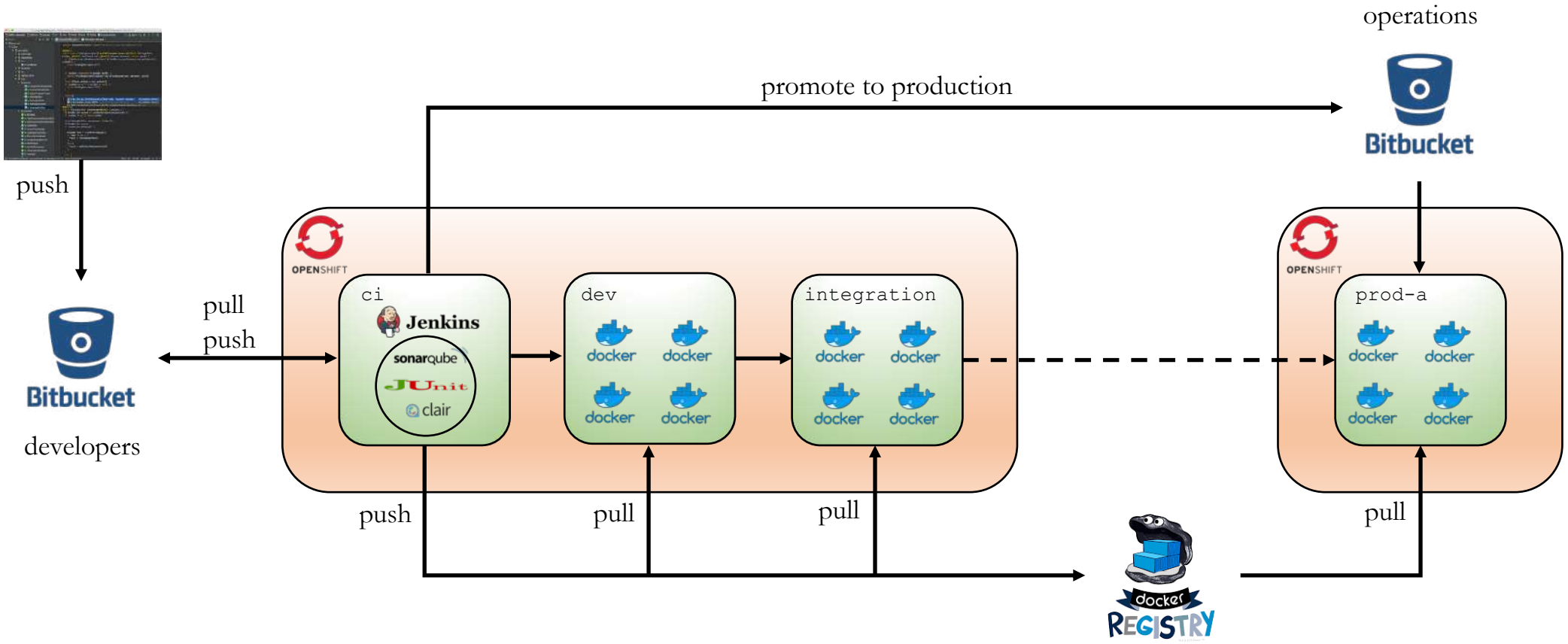
Generated configuration

```
<namespace>
└── apps
    └── demo-service
        ├── demo-service-secrets.yaml
        ├── generated-deployment-list.yaml
        └── kong-configuration.txt
```

- apiVersion: apps/**v1beta1** → apps/**v1beta2** → apps/**v1**

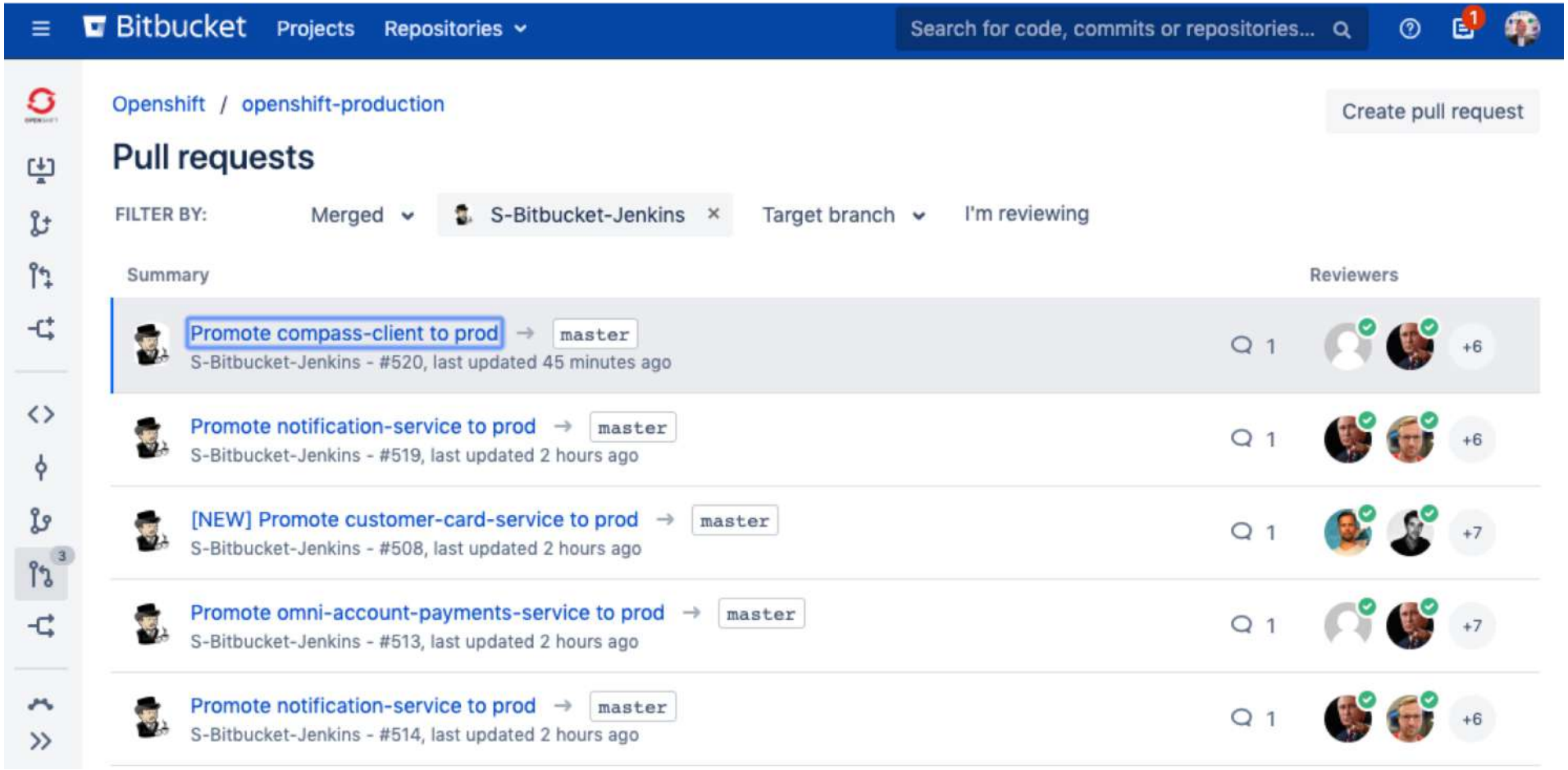
Resurs Bank

Pipeline



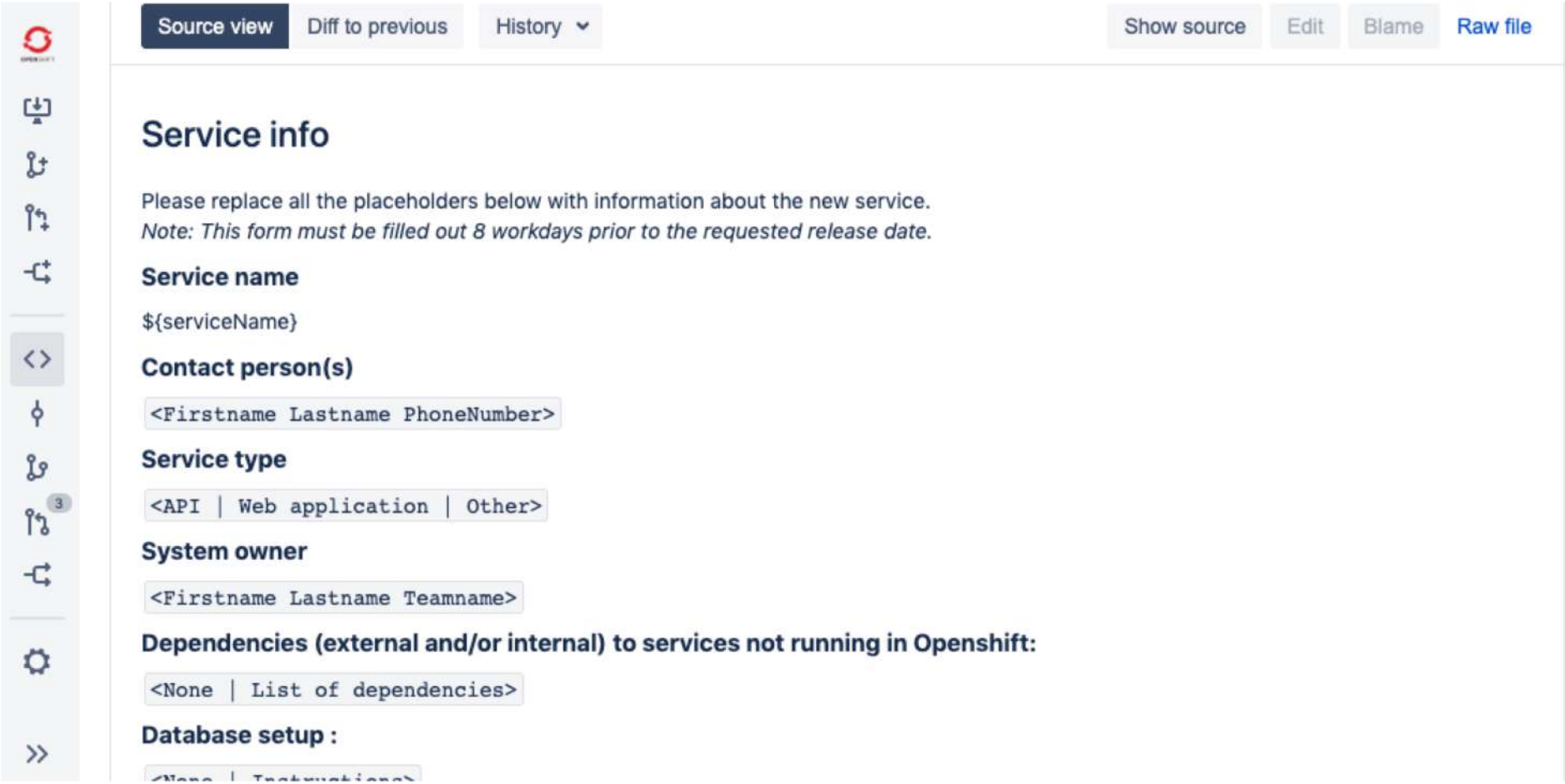
Resurs Bank

Jenkins create PR to Operations



Resurs Bank

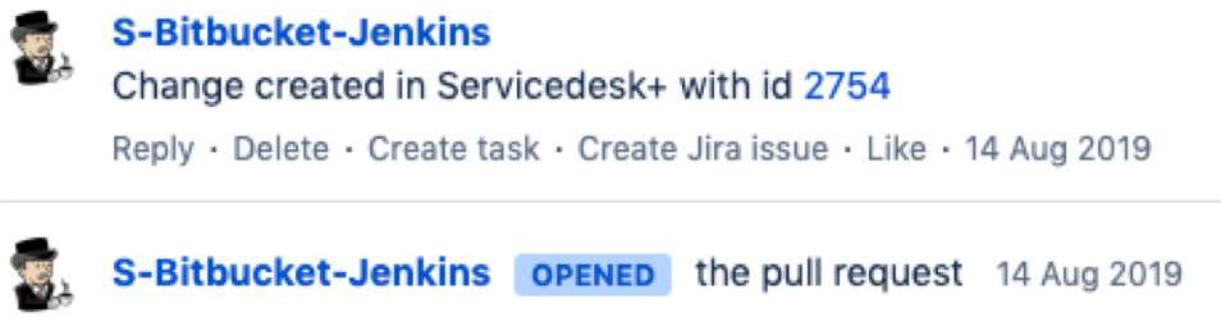
New service with PR template



Resurs Bank

ServiceDesk Plus integration

- Standard change
- Change management gets what they want
- Deploy to Production, any time, almost . . .



Resurs Bank

Faster deploy to production

- Legacy environment
  - Announce intent to deploy on Thursday, deployed on Tuesday 06.00
- OpenShift
  - Approve your PR by 09.00 on Tue/Thu, deployed immediately

- ~800 deployments to Production in 18 months
  - < 10 of them have been rolled back

Resurs Bank

Deploy all services

- `oc apply -recursive -f .`
- Unfortunately startup order of services matters
  - Configure `PriorityClass`
  - Use `priorityClassName` in Deployment for service
- Flux*
  - The state in the cluster matches the configuration in git, automatically

*https://fluxcd.io/

Resurs Bank

Upgrading OpenShift

- 3.7 → 3.10 was ok, but . . .

Resurs Bank

Upgrading OpenShift

- 3.7 → 3.10 was ok, but . . .
- 2018-11-20

```
PV/PVC failed using vSphere-managed storage

E1123 09:44:27.960669       1 vsphere.go:1077]
   Failed to get shared datastore:
   No shared datastores found in the Kubernetes cluster for nodeVmDetails:
         [{NodeName:node1 vm:0xc42501bf20 VMUUID:}
          {NodeName:node2 vm:0xc42501bf60 VMUUID:}
          {NodeName:node3 vm:0xc42501bfa0 VMUUID:}
          {NodeName:nodeX vm:0xc42501bfe0 VMUUID:}]
```

Resurs Bank

vSphere Storage will NOT work

- 2018-12-19

  This configuration will **NOT work**.
  I(eng) also think that, this **never worked** in 3.7 too.

- 2019-01-03

  In Openshift-3.7 we(I am using we but it is vmware that changed this code) **used to allow** provisioning of volumes even if datastores being used was not shared between all VMs in the cluster.

Resurs Bank

Ceph

- Decided to move persistent storage from vSphere cloud provider to Ceph.
- New Ceph cluster installed, spanning both datacenters and all three compute/storage tiers. Block based (RBD) and S3 style access.

- OpenShift uses RBD storage
- Services use S3 storage

Resurs Bank

Continuing the upgrade

- Migrate from vSphere cloud provider to Ceph storage
- Upgrade test clusters to 3.11
- Upgrade production clusters to 3.11

- Install and configure new OpenShift 4.2 clusters

Resurs Bank

Why not 4.1?

- You can disable Telemetry, but …
  - You cannot perform subscription management
  - No disconnected subscription management

- Waiting for 4.2
  - Can we do disconnected subscription management
  - Disable Telemetry

2019 Roadmap

Q3 CY2019
OpenShift 4.2

DEV
- Developer Console GA
- Serverless w/ Knative Tech Preview
- OpenShift Pipelines (Tekton) Tech Preview
- CodeReady Containers GA
- Developer CLI (odo) GA

APP
- GPU metering
- OperatorHub Enhancements
- Operator Deployment Field Forms
- Application Binding with Operators
- Application Migration Console

PLATFORM
- ~~Kubernetes 1.14 w/ CRI-O runtime~~
- Disconnected Install and Update
- ~~Automated Installer for Azure, OSP, GCP~~
- OVN Tech Preview
- FIPS
- Federation Workload API
- Automated App cert rotation
- OpenShift Container Storage 4.2

HOSTED
- UHC Multi-Cluster deployment
- Proactive Support Operator

Resurs Bank

Monitoring

- AppDynamics is used for legacy services
- In OpenShift, just a new base image
  - `FROM base/`**`resurs-minimal-java8-appdyn:`**`latest`
  - And two AppDynamics environment variables
- But . . .

Resurs Bank

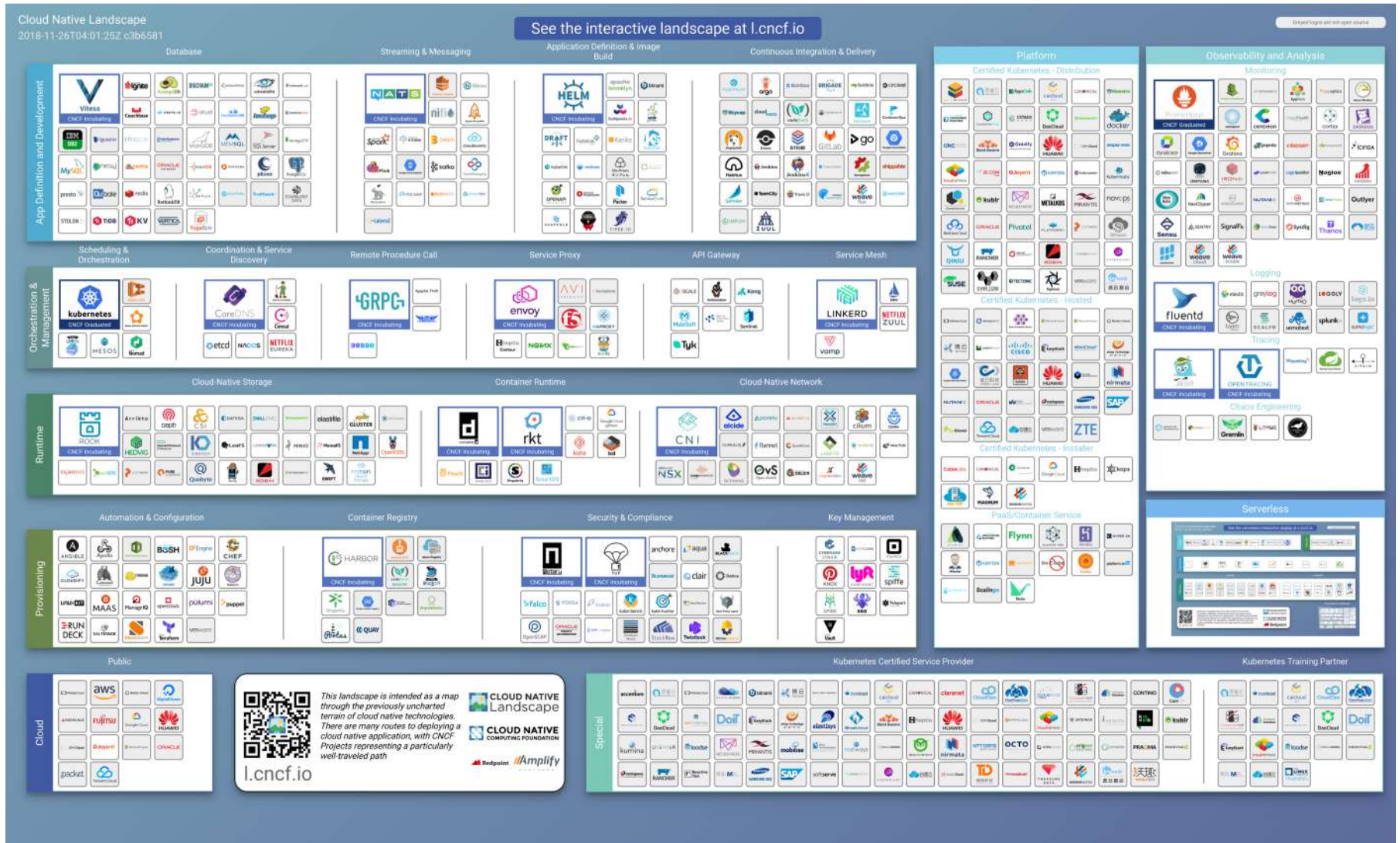
AppDynamics issues in OpenShift

- Needs agent
- License count not managed properly
- Problems with picking up some traffic
- Dependent of specific technology and versions
  - No Java 11 support, yet
  - No project Reactor support, yet

Resurs Bank

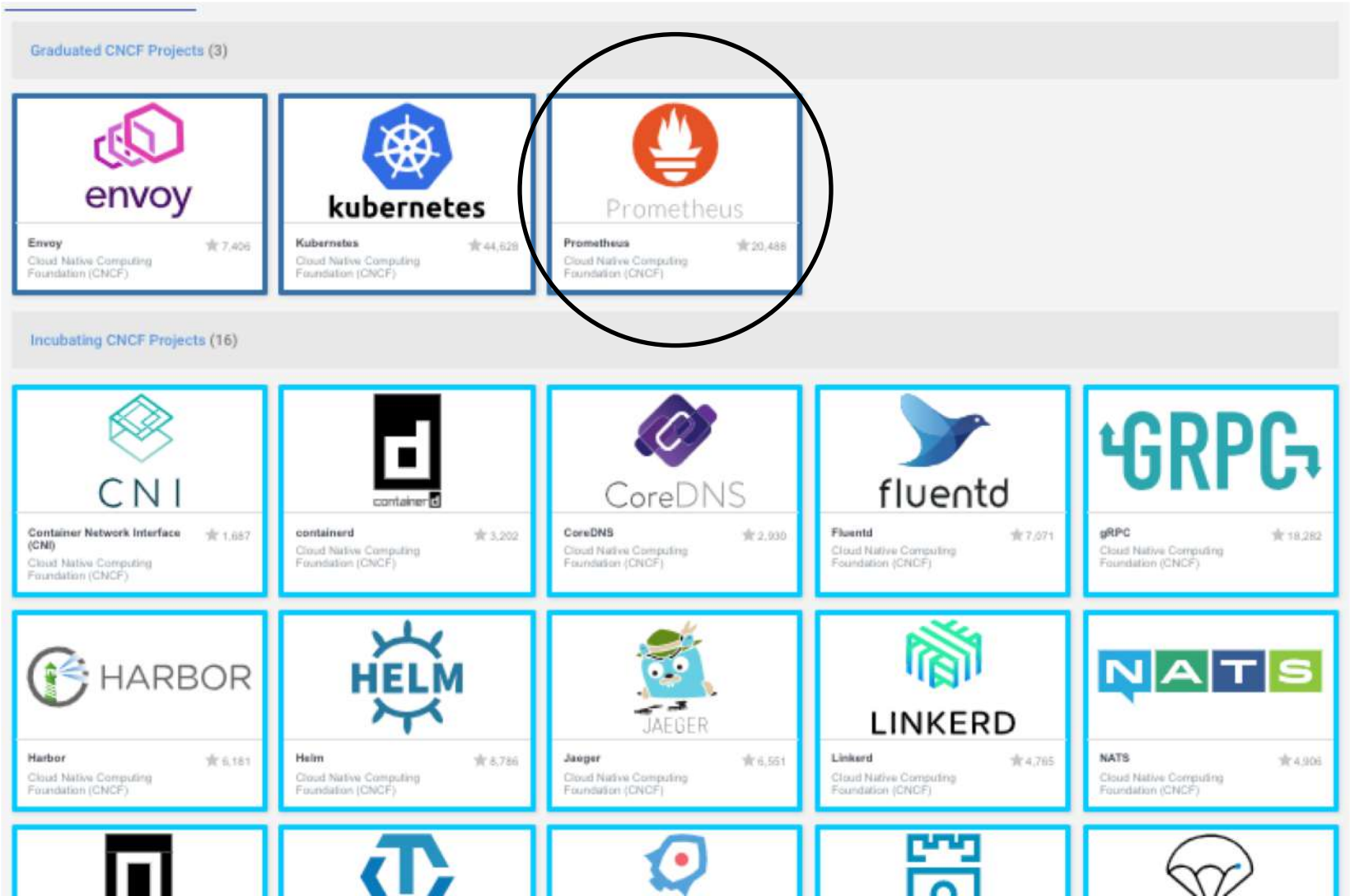
CNCF Landscape

https://landscape.cncf.io



Resurs Bank

Graduated projects



Resurs Bank

Prometheus in OpenShift

- Prometheus is the de-facto standard for Cloud Native applications
- All services are already Prometheus ready

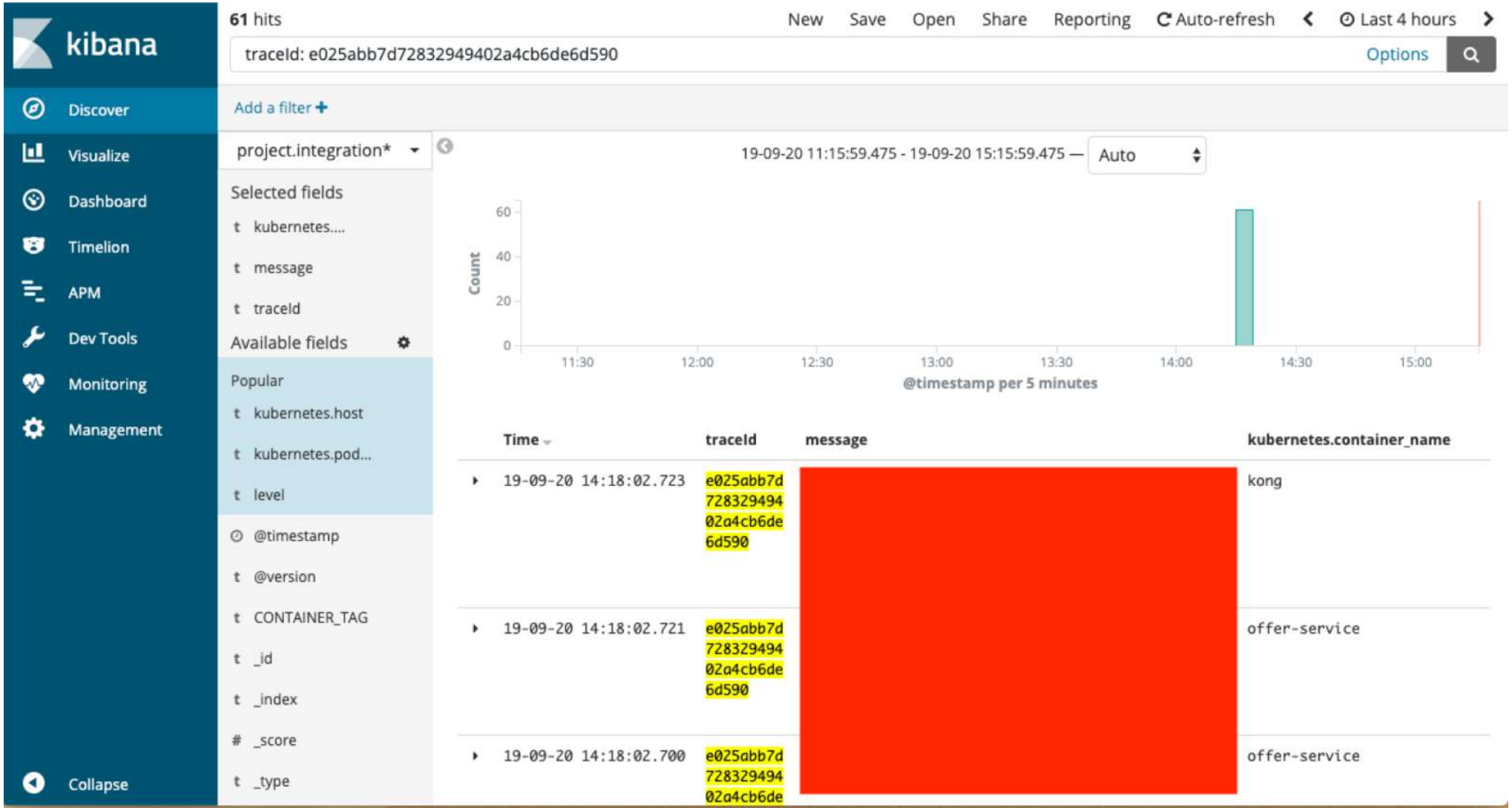
Feature	OCP 3.9	OCP 3.10	OCP 3.11
Prometheus Cluster Monitoring	TP	TP	GA

*Prometheus, a CNCF project that collects time-series data as a source for triggering alerts, has emerged as a **leading standard for cloud-native monitoring** within Kubernetes.**

*https://www.redhat.com/en/blog/generally-available-today-red-hat-openshift-container-platform-311-ready-power-enterprise-kubernetes-deployments

Resurs Bank

Kibana



Resurs Bank

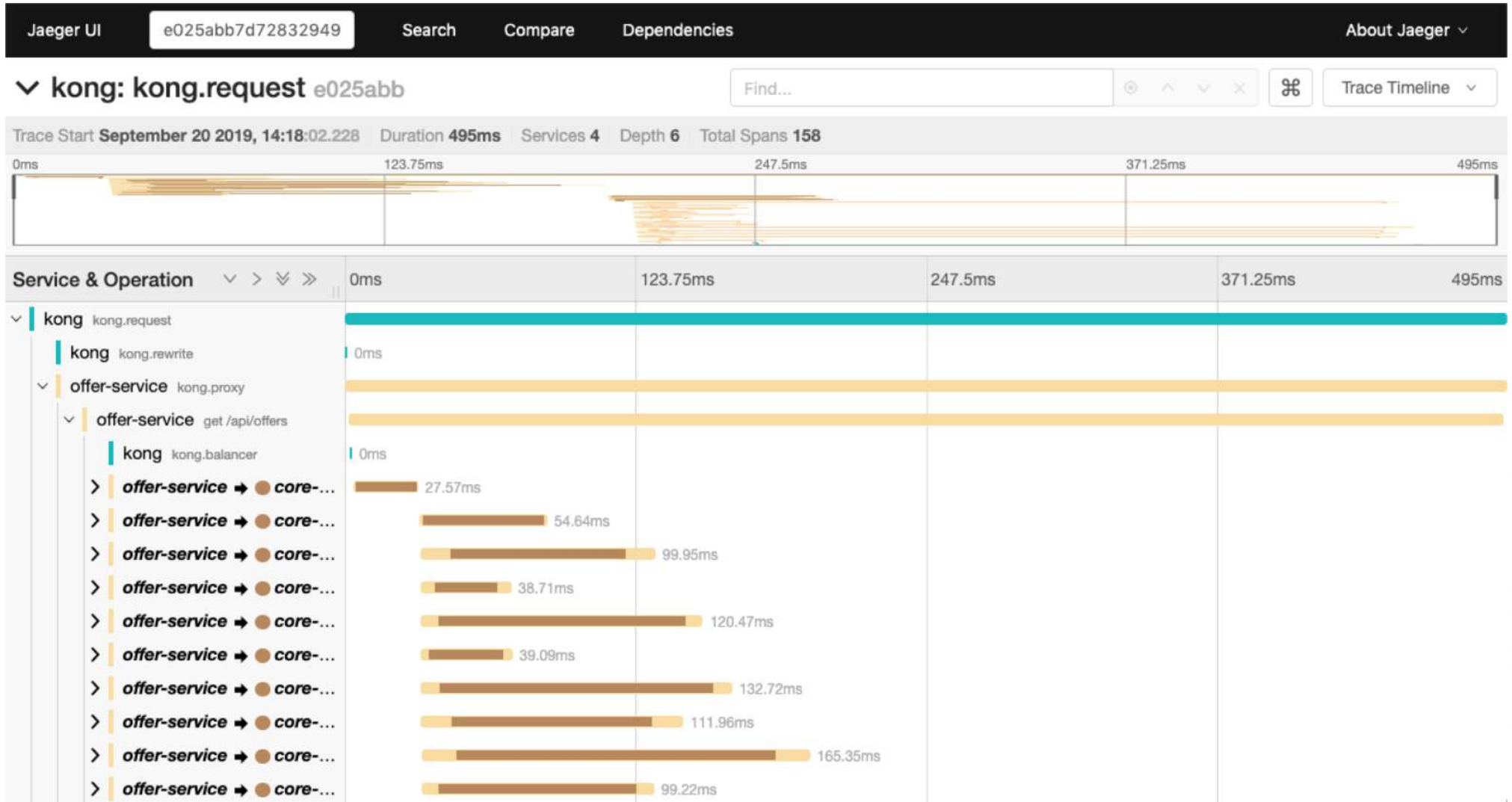
log_format for Kong

```
{
    "traceId":  "$http_x_b3_traceid",
    "spanId":   "$http_x_b3_spanid",
    "consumer": "$http_x_consumer_username",
    "status":   $status,
    "duration": $request_time,
    "message":  <standard access log format>
}
```

- duration > 1.5 AND status: 200
- consumer: THE_CONSUMER AND status: 500 AND message: "/api/demo_service/xyz"

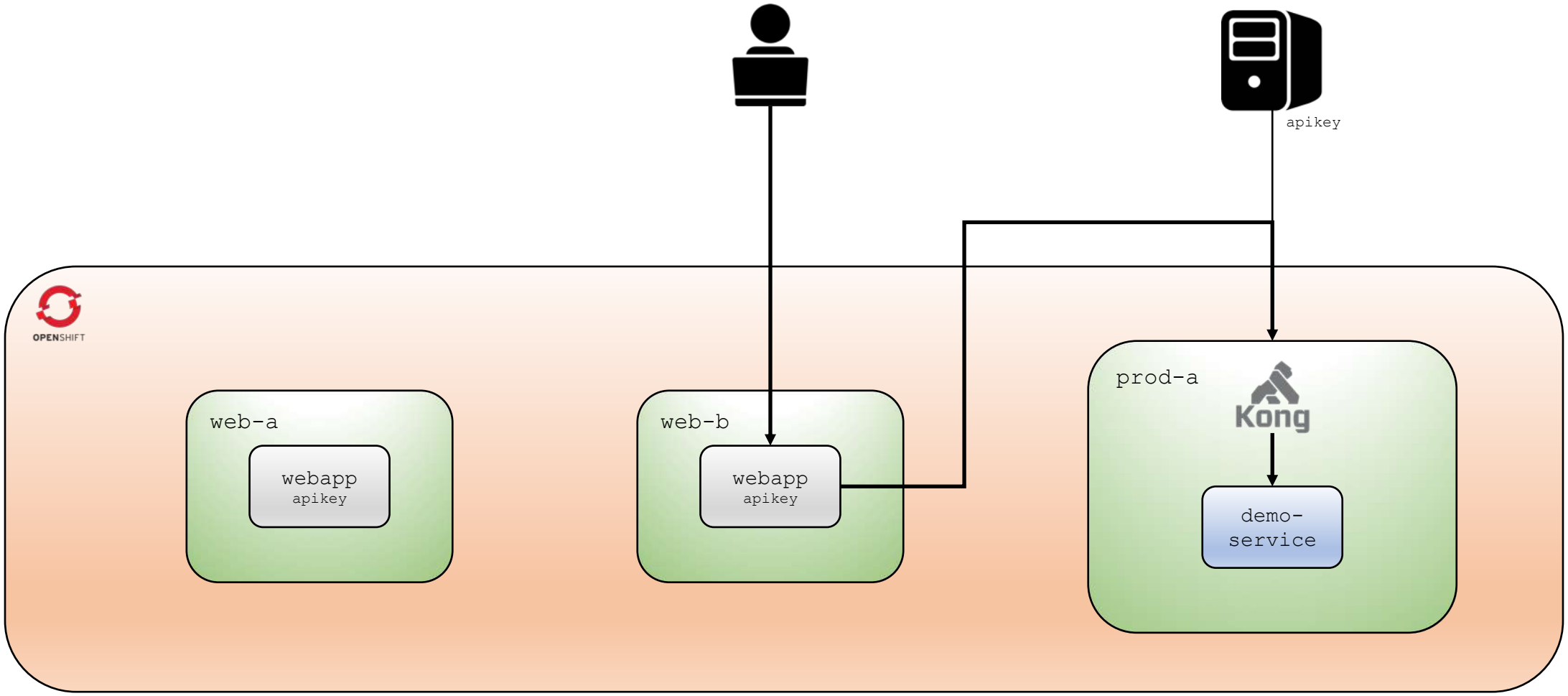
Resurs Bank

Jaeger



Resurs Bank

New namespace for web application



Resurs Bank

Status of today

- 85 services
- 11 web applications
- All teams have something running in OpenShift
  - Even departments not part of IT

Resurs Bank

The future, a never ending story

- Legacy application namespace
- Quarkus
- Operators
- OpenShift Pipelines (Tekton)
- Service Catalog
- Serverless with Knative
- Service Mesh
- NetworkPolicy
- Image signing

Resurs Bank

And finally

- Succeed with a small, focused team
- Make OpenShift the place where the developers want to be
- It's hard to satisfy everyone
- Automate everything