MY OWN CUSTOM KAFKA CLUSTER

IN 60 SECONDS

Innovation Process Technology AG

Daniel Strebel & Jonathan Gan

Welcome

whois ipt.ch

```
This information is subject to an Acceptable Use Policy.
See https://www.nic.ch/terms/aup/


Domain name:
ipt.ch

Holder of domain name:
Innovation Process Technology Inc.
Thomas Schaller
Poststrasse 14
CH-6300 Zug
Switzerland

Technical contact:
Comfox AG
Martin Fuchs
```

ipt AG @ ERFA

whoami #Daniel

🇨🇭 Born in Zurich

🤓 Computer Science Masters UZH

💼 Analytics, Corporate R&D, Innovation Labs

🚀 Cloud Architecture, Data, ML

💙 Running, Hiking, Travel

whoami #Jonathan

🇸🇬 Born in Singapore

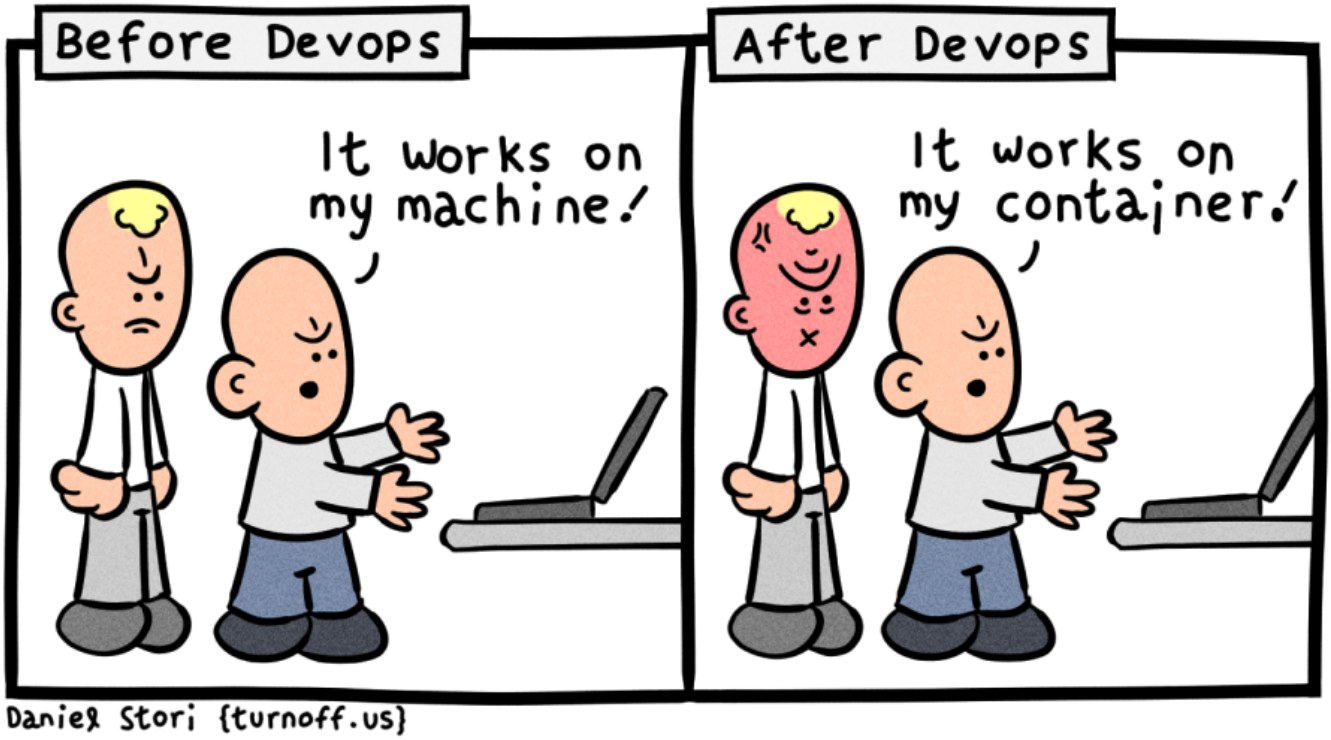
🤓 Robotics Masters ETH

💼 {Disney, IBM} Research

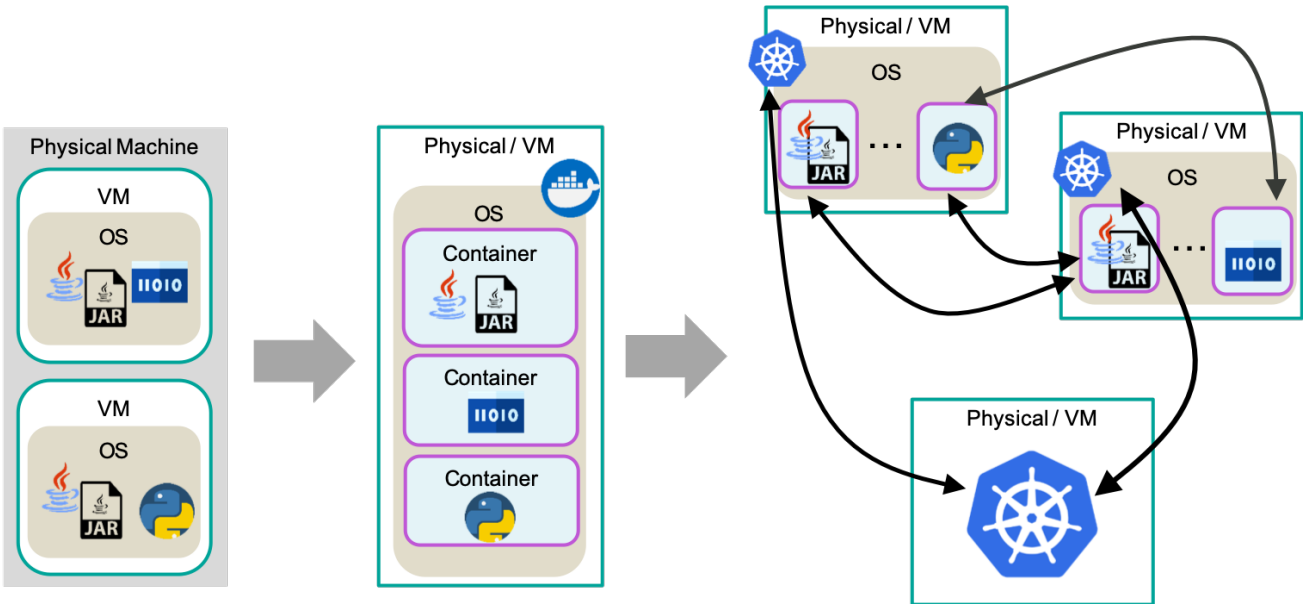
🚀 Computer Vision, ML, DevOps

💙 Music, Cooking, Gym

Kubernetes in a Nutshell



From Bare Metal to Container Orchestration



k8s Lingo

Pod

Container

Deployment

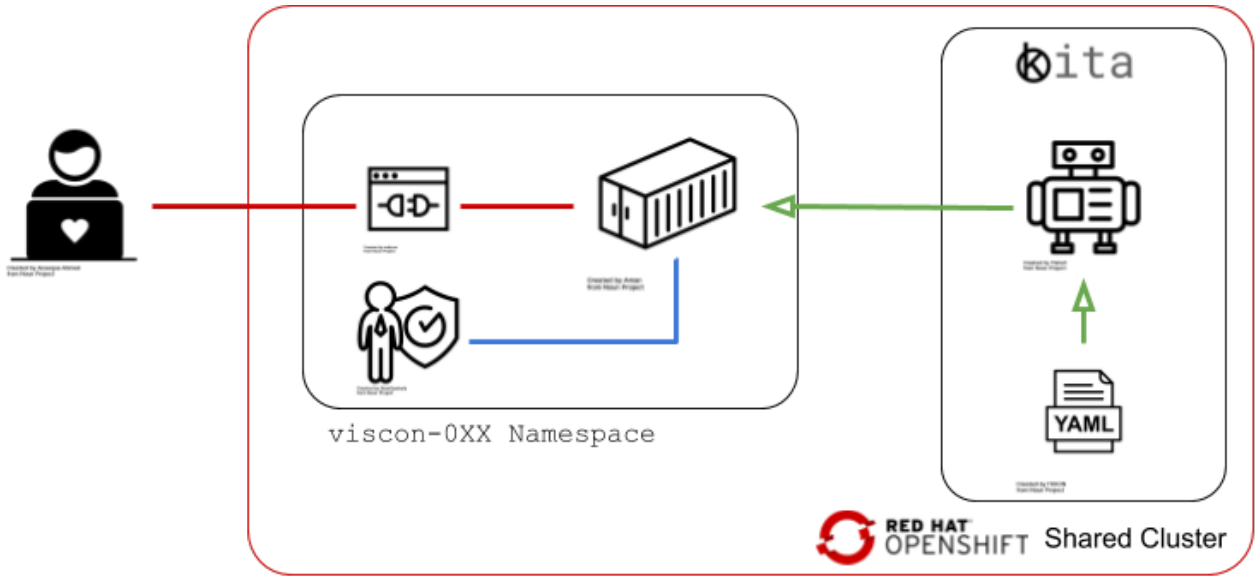
Service

Route

Serviceaccount

Intro to the Lab Environment

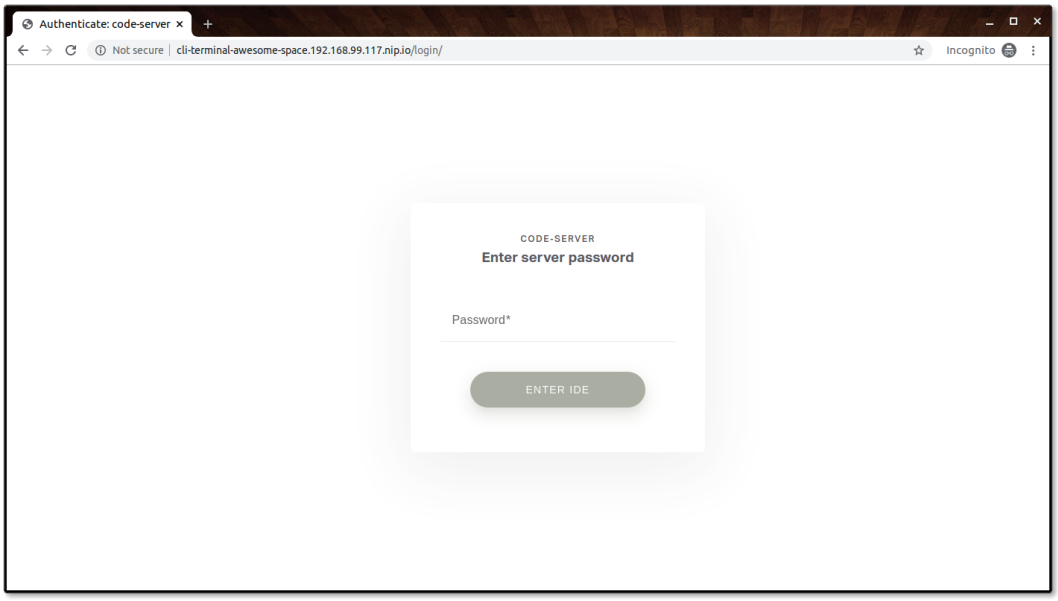
The high level picture



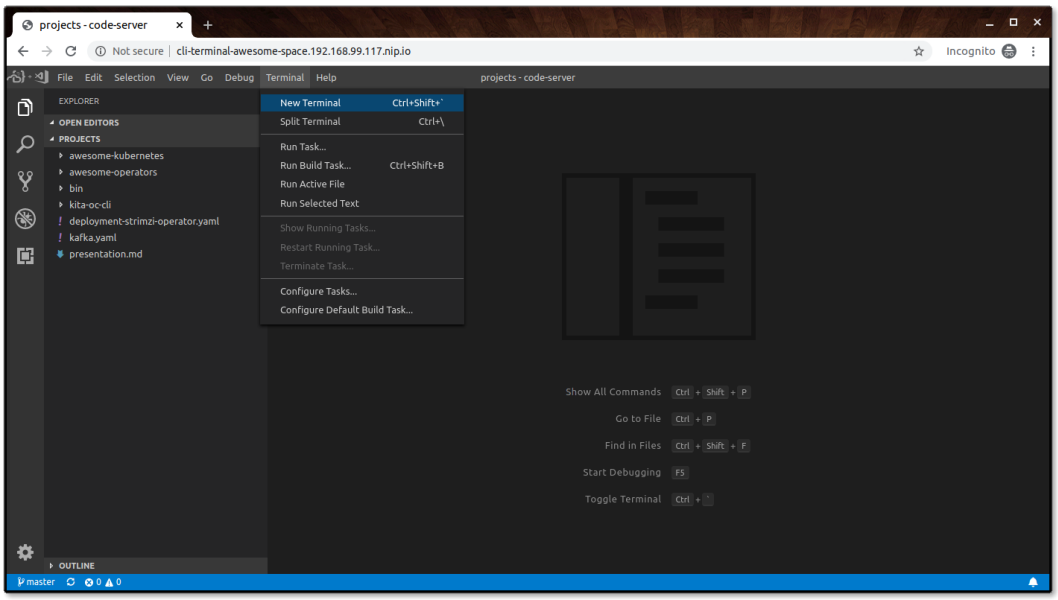
Rule 1/1: Stick to the Instructions

🙌 Play with your Workspace

Visit the URL on your snippet and enter the password to access the workspace



🙌 Prepare your Shell



🙌 Openshift CLI

Download and install the OC CLI

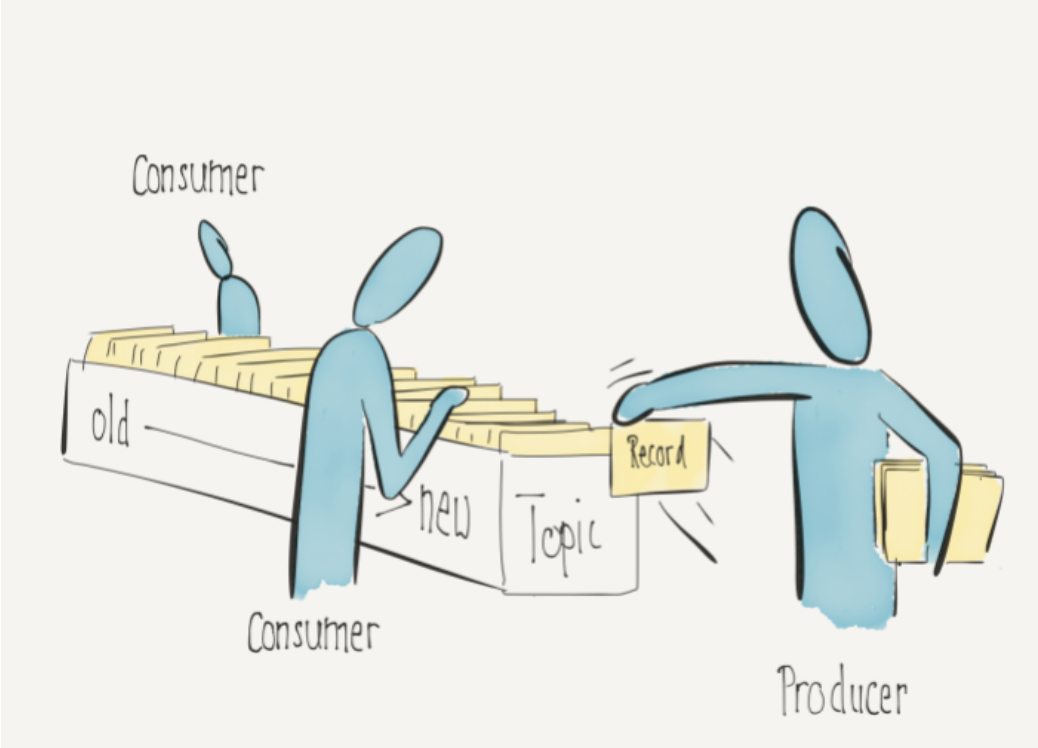
```
source ./kita-oc-cli/install.sh
```

Try a few things

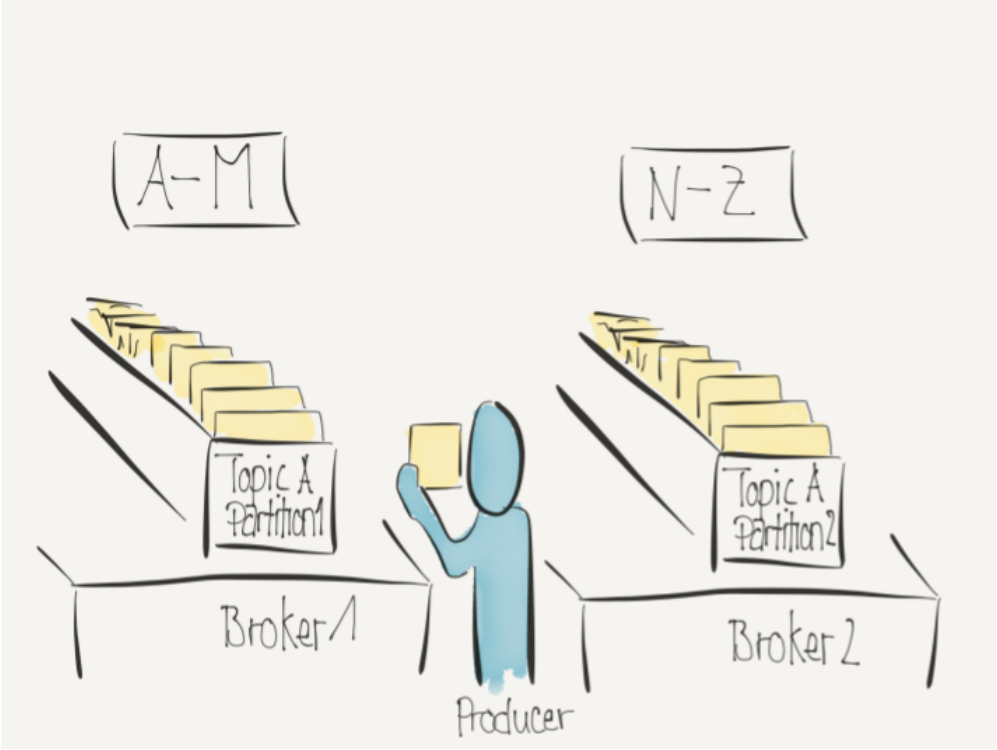
```
oc whoami

oc get all
```

Apache Kafka

Kafka High-Level



Kafka Secret Sauce



What is Strimzi

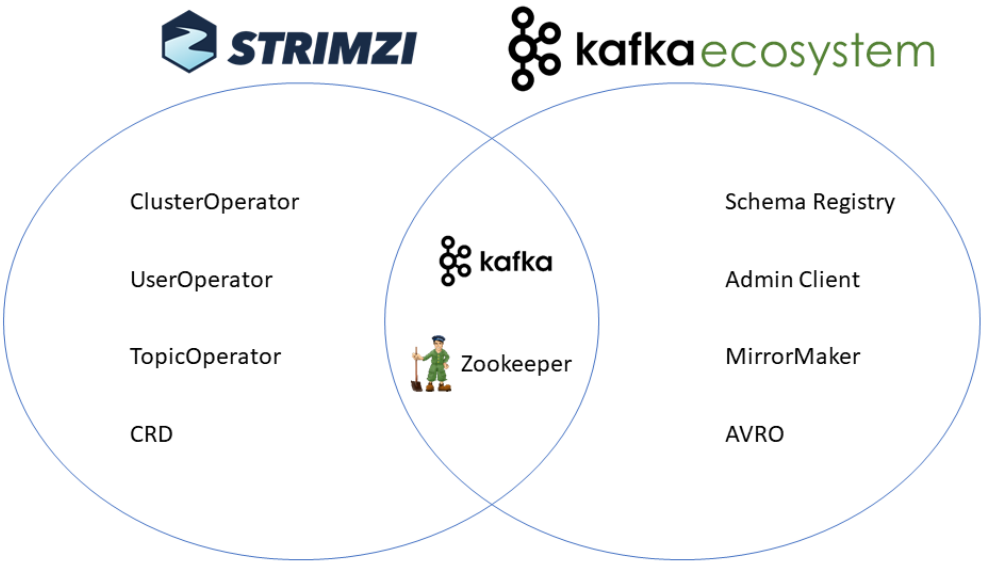
Open Source Project (started by Red Hat)

Recencly adopted as a CNCF Project

Using Operators to deploy and manage Apache Kafka on Kubernetes / Openshift

Used by ipt in production since GA in 2018

Strimzi Components



Strimzi Installation

What has been done for you
(because it requires cluster admin rights)

Custom Resource Definitions

```
oc get CustomResourceDefinitions | grep kafka


kafkas.kafka.strimzi.io
kafkatopics.kafka.strimzi.io
kafkausers.kafka.strimzi.io
```

Strimzi Installation Cont.

What has been done for you
(because it requires cluster admin rights)

Roles and ClusterRoles

```
oc get ClusterRoles | grep strimzi

NAME
strimzi-cluster-operator-global
strimzi-cluster-operator-namespaced
strimzi-entity-operator
strimzi-kafka-broker
strimzi-topic-operator
```

And assigned them to the `strimzi-cluster-operator` service account as well as your "user"

🙌 Setting up the cluster

Change into the tutorial project

```
cd /home/coder/projects/viscon-strimzi
```

🙌 Strimzi Custom Resource

```yaml
apiVersion: kafka.strimzi.io/v1beta1
kind: Kafka
metadata:
  name: viscon-cluster
spec:
  kafka:
    version: 2.3.0
    replicas: 3
    listeners:
      plain: {}
      tls: {}
    config:
      #...
    storage:
      type: ephemeral
  zookeeper:
    replicas: 1
```

see `./strimzi/kafka/kafka-plain.yaml`

🙌 Strimzi Operator Deployment

First, we deploy the strimzi operator

```
oc apply -f strimzi/deployment-strimzi-operator.yaml
```

and wait until it is ready

```
oc get pod -l name=strimzi-cluster-operator
NAME                                        READY     ST
strimzi-cluster-operator-6bfb66c5f7-n7qr2   1/1       Ru
```

we can also inspect its logs by running

```
oc logs -f $(oc get pod -l name=strimzi-cluster-operator
```

🙌 Kafka Cluster Deployment

Create a custom Kafka resource

```
oc apply -f strimzi/kafka/kafka-plain.yaml
```

Watch the operator create a full Kafka cluster
(in about 40s 🎉)

```
oc get pods

NAME                                          READY
strimzi-cluster-operator-6bfb66c5f7-n7qr2     1/1
viscon-cluster-entity-operator-6cbbd888bc-hn9fm   3/3
viscon-cluster-kafka-0                        2/2
viscon-cluster-kafka-1                        2/2
viscon-cluster-zookeeper-0                    2/2
```

First End To End Example

Start simple producers and consumers to write to and read from a Kafka topic.

🙌 First Kafka Producer

Start a console producer pod

```
oc run kafka-producer -ti --image=strimzi/kafka:0.14.0-k
 --rm=true --restart=Never -- bin/kafka-console-producer
 --broker-list viscon-cluster-kafka-bootstrap:9092 \
 --topic viscon-console-demo
```

and produce a few messages

```
If you don`t see a command prompt, try pressing enter.
>Hello Viscon
>2nd hello
>3rd hello
```

Kill the session with CTRL + C when done

🙌 First Kafka Consumer

Start a console consumer pod

```
oc run kafka-consumer -ti --image=strimzi/kafka:0.14.0-k
 --rm=true --restart=Never -- bin/kafka-console-consumer
 --bootstrap-server viscon-cluster-kafka-bootstrap:9092
 --topic viscon-console-demo --from-beginning
```

and receive the previously produced messages

```
If you don`t see a command prompt, try pressing enter.
Hello Viscon
2nd hello
3rd hello
```

Kill the session with CTRL + C when done

🙌 Explore Kafka Topics

Explore the Topics within Kafka

```
oc run kafka-topic-list -ti --image=strimzi/kafka:0.14.0
--rm=true --restart=Never -- bin/kafka-topics.sh --list
--bootstrap-server viscon-cluster-kafka-bootstrap:9092

__consumer_offsets
viscon-console-demo
pod "kafka-topic-list" deleted
```

And the generated KafkaTopic Resources

```
oc get KafkaTopics

NAME
consumer-offsets---84e7a678d08f4bd226872e5cdd4eb527fadc1
viscon-console-demo
```

🙌 Kafka Topic - Spec First

Take a look at the KafkaTopic in

`strimzi/console-demo/topic.yaml`

```yaml
apiVersion: kafka.strimzi.io/v1beta1
kind: KafkaTopic
metadata:
  name: viscon-topic
  labels:
    strimzi.io/cluster: viscon-cluster
spec:
  partitions: 6
  replicas: 2
  config:
    retention.ms: 86400000 #24h
```

Now, create the topic by applying the CR

```
oc apply -f strimzi/console-demo/topic.yaml
```

🙌 Kafka Topic - Operator Magic

Let's look at the Entity Operator Logs

```
oc logs -f $(oc get pods -l strimzi.io/name=viscon-clust
```

And the Kafka internal Topic List

```
oc run kafka-topic-list -ti --image=strimzi/kafka:0.14.0
 --rm=true --restart=Never -- bin/kafka-topics.sh --list
 --bootstrap-server viscon-cluster-kafka-bootstrap:9092

__consumer_offsets
viscon-topic
viscon-console-demo
pod "kafka-topic-list" deleted
```

That was cute?!

So what is missing?

Authentication

Processing

Metrics

🙌 Kafka Authentication

We no longer want our topics to be open to the world.

We can enforce mutual tls
by adapting our Kafka resource:

```
diff strimzi/kafka/kafka-plain.yaml strimzi/kafka/kafka-
10,11c10,12
<       plain: {}
<       tls: {}
---
>       tls:
>         authentication:
>           type: tls
```

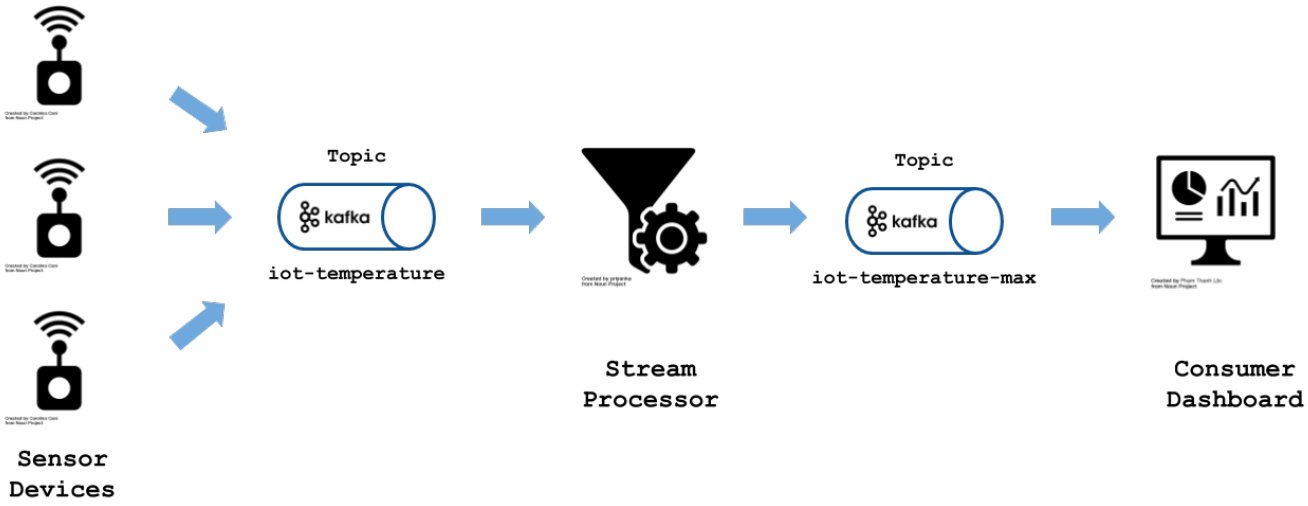
🙌 Kafka Authentication Cont.

To change the broker listener config we apply the `strimzi/kafka/kafka-mtls.yaml` config.

```
oc apply -f strimzi/kafka/kafka-mtls.yaml
```

and wait for the Kafka brokers to update

More Sophisticated Scenario



🙌 Create Demo Producer User

`strimzi/iot-demo/iot-device/users.yaml`

```yaml
apiVersion: kafka.strimzi.io/v1beta1
kind: KafkaUser
metadata:
  name: iot-device
  labels:
    app: iot-demo
    strimzi.io/cluster: viscon-cluster
spec:
  authentication:
    type: tls
  authorization:
    type: simple
    acls:
      - resource:
          type: topic
          name: iot-temperature
          patternType: literal
```

🙌 Create Demo Stream User

`strimzi/iot-demo/iot-stream/users.yaml`

```yaml
apiVersion: kafka.strimzi.io/v1beta1
kind: KafkaUser
metadata:
  name: iot-stream
  labels:
    app: iot-demo
    strimzi.io/cluster: viscon-cluster
spec:
  authentication:
    type: tls
  authorization:
    type: simple
    acls:
      - resource:
          type: topic
          name: iot-temperature
          patternType: literal
```

🙌 Create Demo Consumer User

`strimzi/iot-demo/iot-consumer/users.yaml`

```yaml
apiVersion: kafka.strimzi.io/v1beta1
kind: KafkaUser
metadata:
  name: iot-consumer
  labels:
    app: iot-demo
    strimzi.io/cluster: viscon-cluster
spec:
  authentication:
    type: tls
  authorization:
    type: simple
    acls:
      - resource:
          type: topic
          name: iot-temperature-max
          patternType: literal
```

🙌 Create all Users

```
oc apply -f strimzi/iot-demo/iot-device/users.yaml
oc apply -f strimzi/iot-demo/iot-stream/users.yaml
oc apply -f strimzi/iot-demo/iot-consumer/users.yaml
```

Verify the ACL for `iot-temperature`

```
oc exec viscon-cluster-zookeeper-0 -c zookeeper -- /opt/
--authorizer-properties zookeeper.connect=localhost:2181
--list --topic iot-temperature
```

and for `iot-temperature-max`

```
oc exec viscon-cluster-zookeeper-0 -c zookeeper -- /opt/
--authorizer-properties zookeeper.connect=localhost:2181
--list --topic iot-temperature-max
```

🙌 Kafka User Secrets

The entity operator created user certificates for all users which include a CA certificate, a public certificate and a private key.

```
oc get secrets -l strimzi.io/kind=KafkaUser -o yaml
```

🙌 Mounting User Secrets

E.g. `strimzi/iot-demo/iot-device/deployment.yaml`

```
containers:
  - name: device-app
    image: danistrebel/strimzi-device-app:latest
    env:
      - name: USER_CRT
        valueFrom:
          secretKeyRef:
            name: iot-device
            key: user.crt
      - name: USER_KEY
        valueFrom:
          secretKeyRef:
            name: iot-device
            key: user.key
```

🙌 Deploy the IoT Device

```
oc apply -f strimzi/iot-demo/iot-device/deployment.yaml
```

and check the logs

```
oc logs -f $(oc get pod -l name=device-app -o name | hea
```

Full Code

https://github.com/danistrebel/strimzi-lab

🙌 Deploy the IoT Stream Processor

```
oc apply -f strimzi/iot-demo/iot-stream/deployment.yaml
```

and check the logs

```
oc logs -f $(oc get pod -l name=stream-app -o name | hea
```

Full Code

https://github.com/danistrebel/strimzi-lab

🙌 Deploy the Consumer

```
oc apply -f strimzi/iot-demo/iot-consumer/deployment.yam
```

and check the logs

```
oc logs -f $(oc get pod -l name=consumer-app -o name | h
```

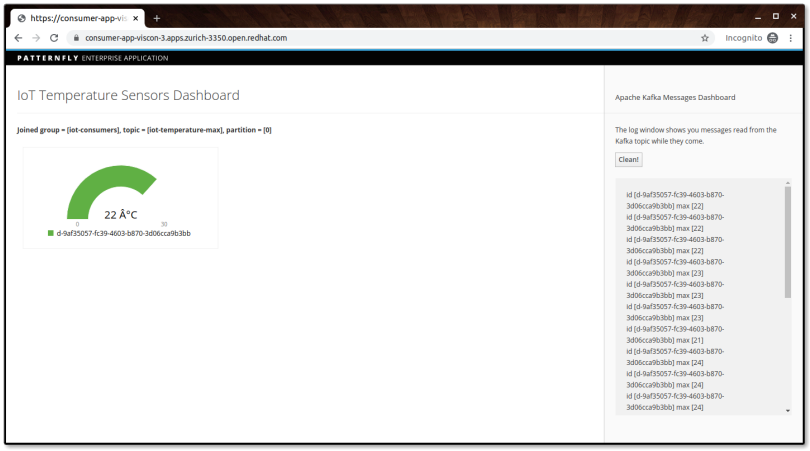
Full Code

https://github.com/danistrebel/strimzi-lab

🙌 Check the Dashboard

```
oc get route consumer-app -o jsonpath='{.spec.host}'
```

and open it in a new Tab

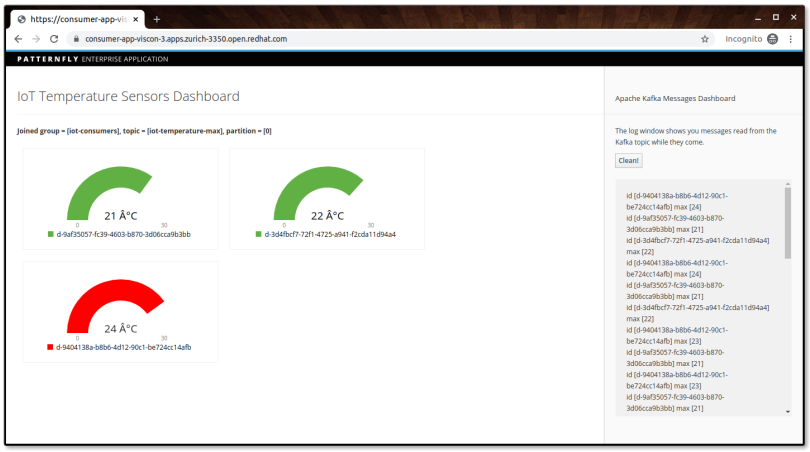


🙌 Scale the Deployment

Increase the number of sensor devices to 3

```
oc scale deployment device-app --replicas 3
```

and wait for the data to show up in the dashboard



🙌 Cluster Monitoring

Create a scrape config

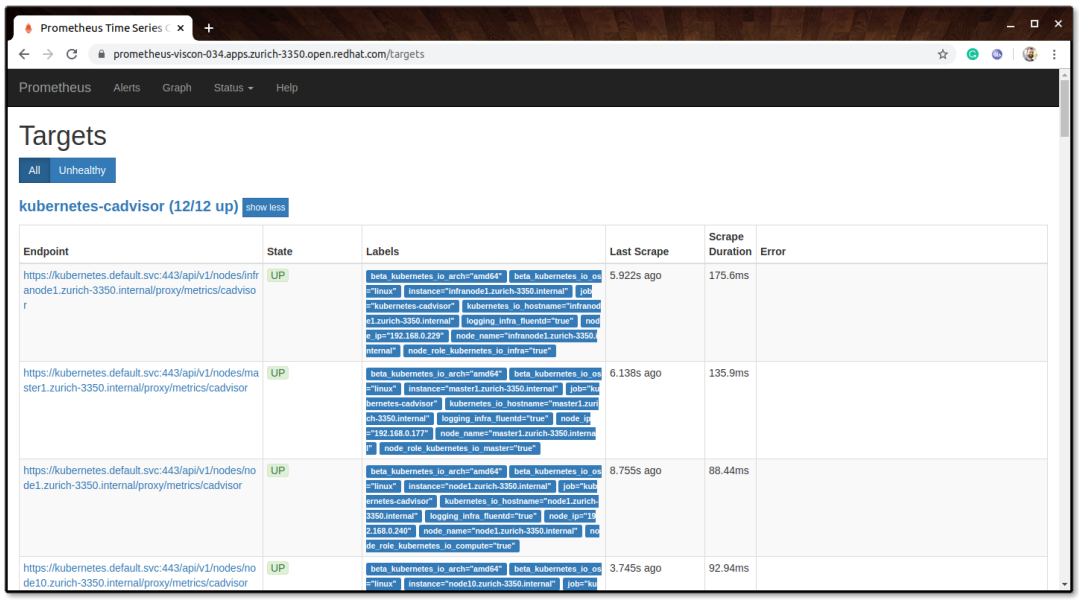
```
oc create secret generic additional-scrape-configs --frc
```

and the Prometheus deployment

```
oc apply -f strimzi/metrics-demo/prometheus.yaml

oc apply -f strimzi/metrics-demo/prometheus-service-moni
```

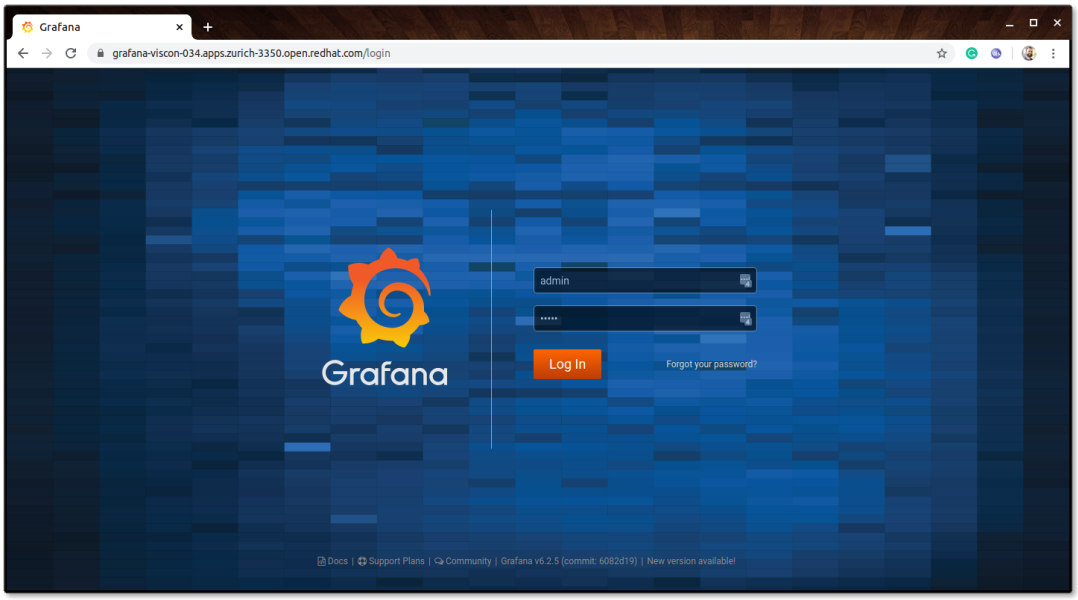
🙌 Check the Prometheus Targets

```
oc get route prometheus
```



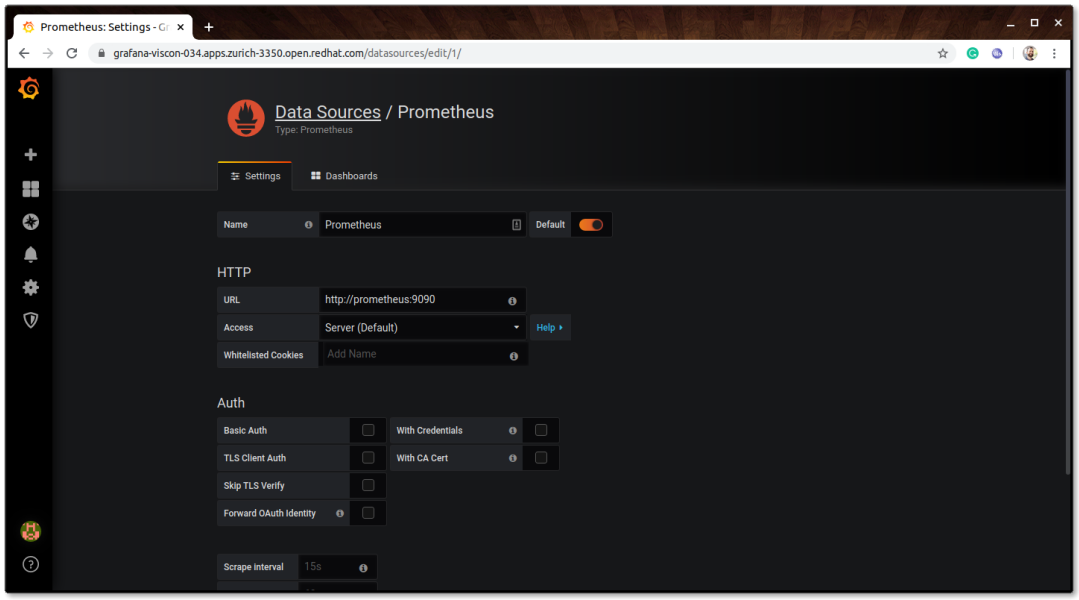
🙌 Deploy Grafana

```
oc apply -f strimzi/metrics-demo/grafana.yaml
oc get routes grafana
```



user: admin, password: admin

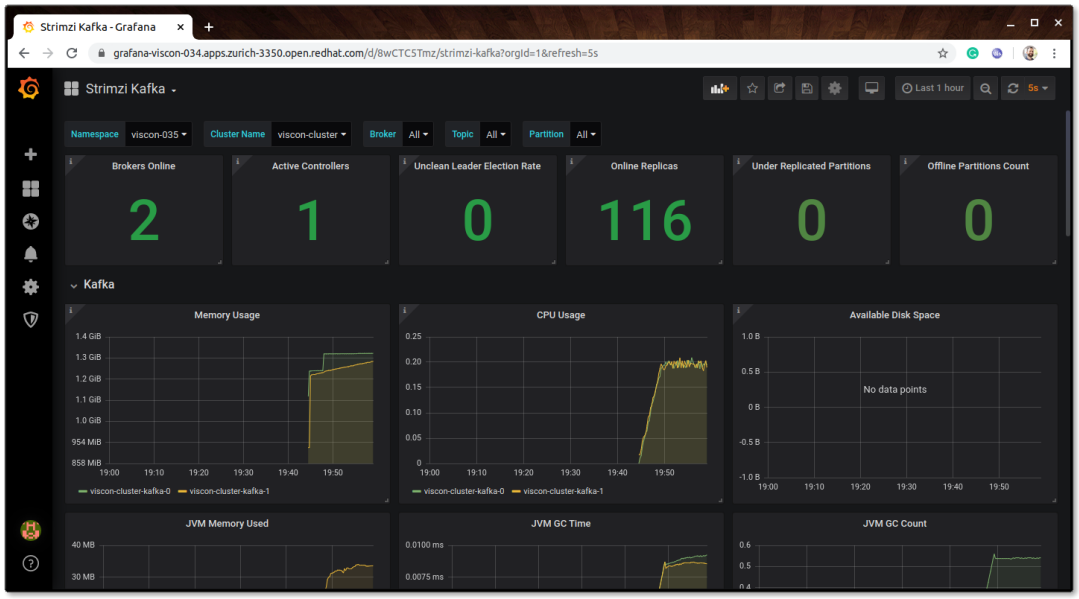
🙌 Configure the Prometheus Datasource



Name: Prometheus, Url: http://prometheus:9090

🙌 Import the Strimzi Dashboard

Copy `strimzi/metrics-demo/kafka-dashboard.json`



Kafka Eco System

Kafka Connect

Mirror Maker

HTTP Bridge

Managed Kafka

- AWS MSK
- Azure Eventhubs
- Confluent on GCP...

Fireside Chat / Q&A

Contacts and Info

Daniel Strebel
Email: daniel.strebel@ipt.ch
Linkedin: linkedin.com/in/danistrebel

Jonathan Gan
Email: jonathan.gan@ipt.ch
Linkedin: linkedin.com/in/jonganej

IPT Careers Portal
ipt.ch/karriere