KubeCon | CloudNativeCon

Europe 2022

WELCOME TO VALENCIA

KubeCon CloudNativeCon Europe 2022

Show Me Your Labels I'll Tell You Who You Are

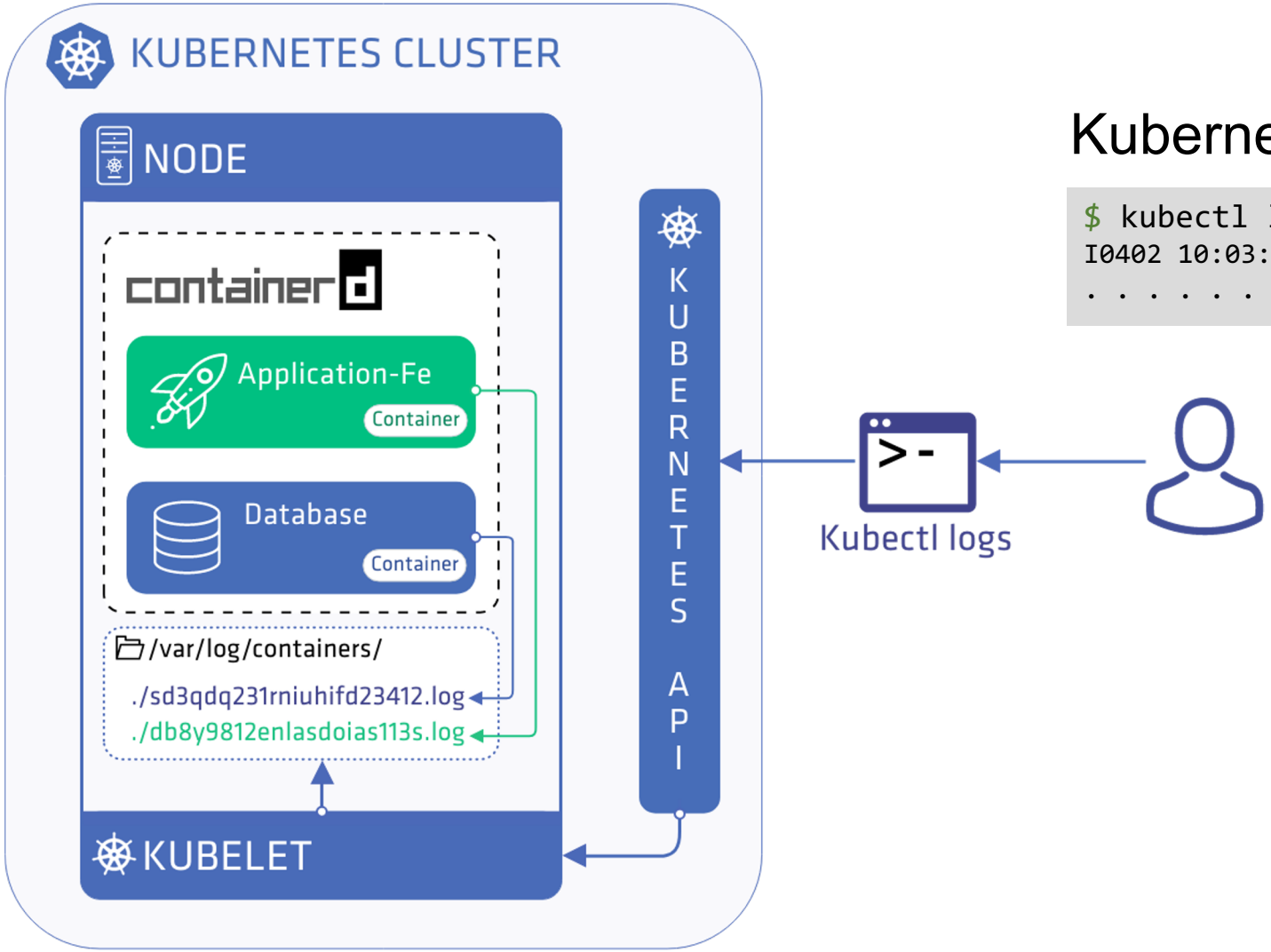
Sandor Guba, Cisco

Show Me Your Labels I'll Tell You Who You Are

- Co-founder & Developer at Banzai Cloud
- 5+ years experience with Kubernetes
- 5+ years experience with fluent ecosystem
- Logging Operator maintainer
- Observability enthusiast
- Prometheus and Thanos fan

Sandor Guba
Engineering Technical Leader, *Cisco*

Logging On Kubernetes



Kubernetes have minimal support for logs

```
$ kubectl logs nginx-5c213699
I0402 10:03:10.301280      6 flags.go:208] "Watching for Ingress”
. . . . . .
```

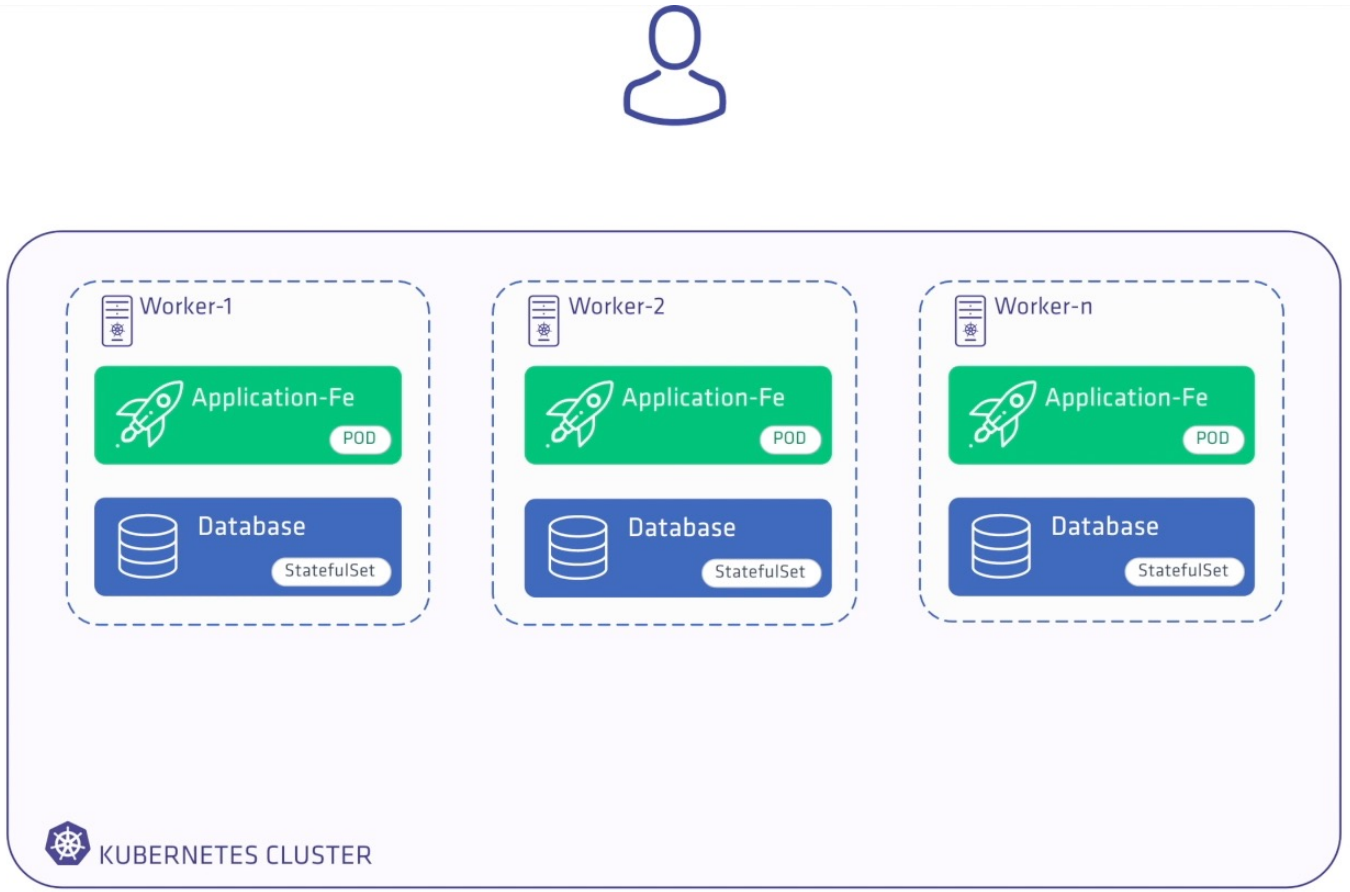
Logging On Kubernetes

- There is pretty much nothing else to configure in Kubernetes!
- What are the problems with this?
- Logs are stored on node's disk
- Eventually, logs will be rotated
- Maybe you want analytics etc…

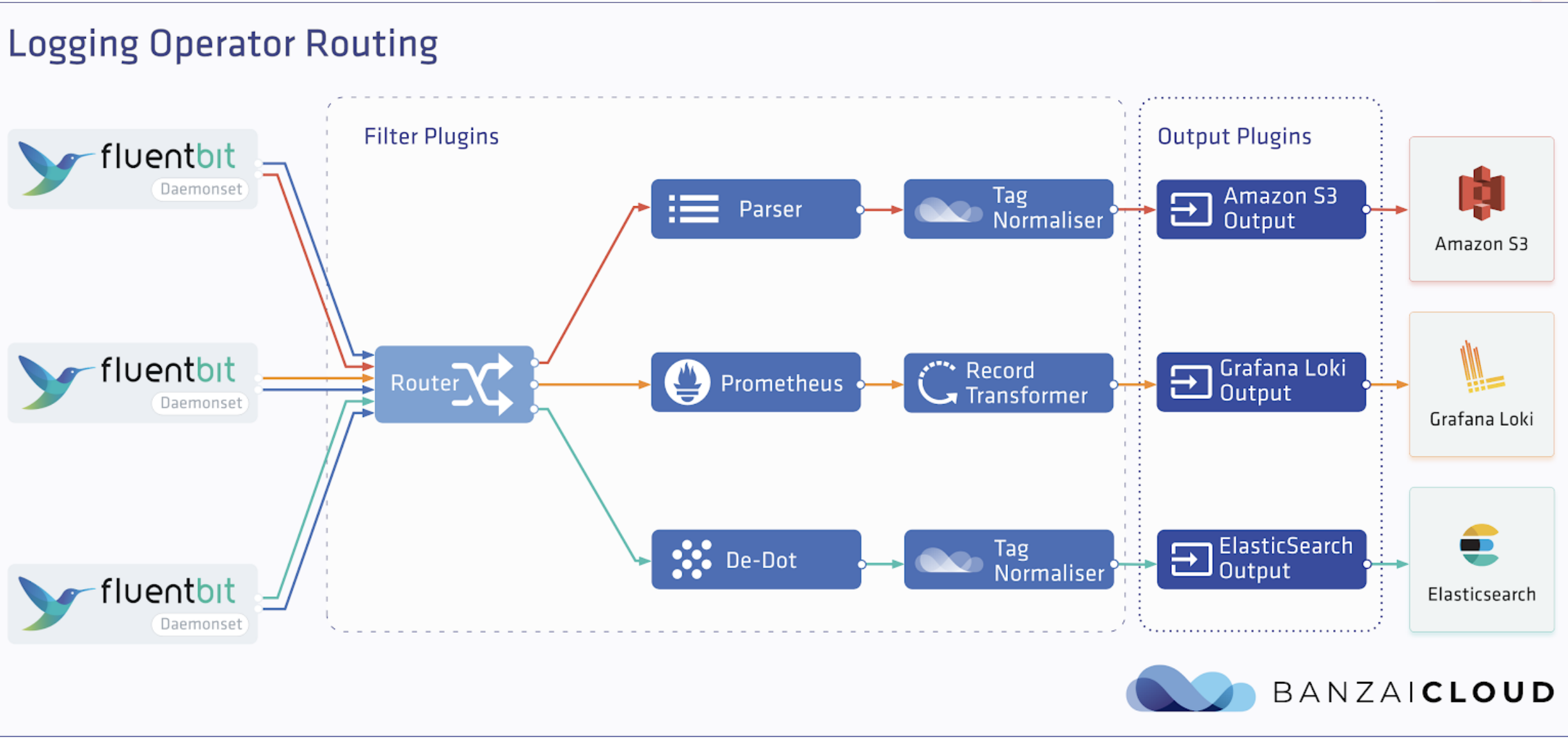
Okay so we ship those logs to elsewhere!

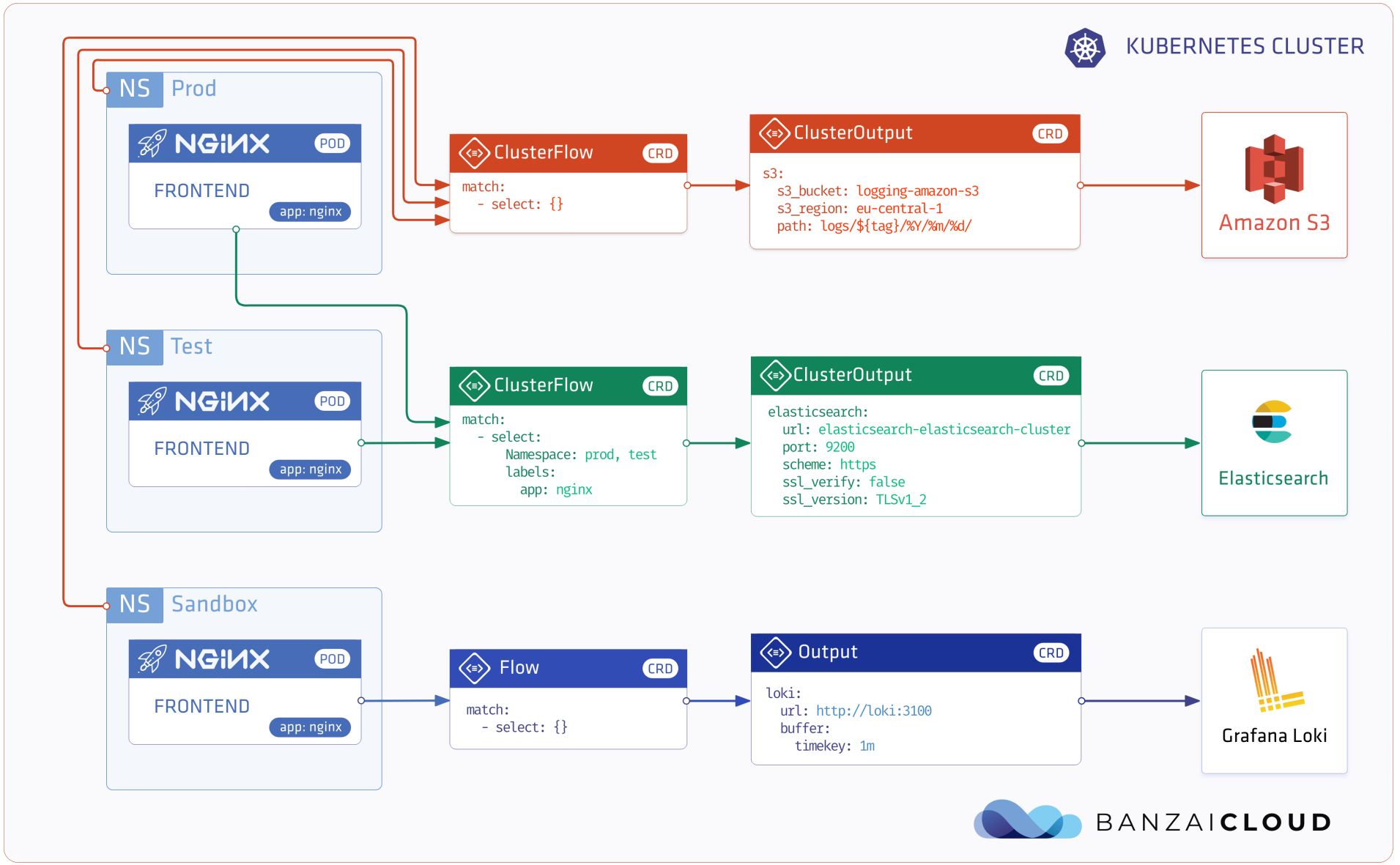
- You need to access the Host filesystem, so you need **privileged** containers.
- No separation on the Host you get all the running Pods' logs.
- They are still just a bunch of files

Logging On Kubernetes



Logging Operator Architecture




KUBERNETES CLUSTER
NS Prod
NGINX POD
FRONTEND
app: nginx
NS Test
NGINX POD
FRONTEND
app: nginx
NS Sandbox
NGINX POD
FRONTEND
app: nginx
ClusterFlow CRD
match:
- select: {}
ClusterOutput CRD
s3:
s3_bucket: logging-amazon-s3
s3_region: eu-central-1
path: logs/${tag}/%Y/%m/%d/
Amazon S3
ClusterFlow CRD
match:
- select:
Namespace: prod, test
labels:
app: nginx
ClusterOutput CRD
elasticsearch:
url: elasticsearch-elasticsearch-cluster
port: 9200
scheme: https
ssl_verify: false
ssl_version: TLSv1_2
Elasticsearch
Flow CRD
match:
- select: {}
Output CRD
loki:
url: http://loki:3100
buffer:
timekey: 1m
Grafana Loki
BANZAICLOUD

Logging Operator - Flow


```yaml
apiVersion: logging.banzaicloud.io/v1beta1
kind: Flow
metadata:
  name: geoip-sample
spec:
  selectors:
    app: nginx
  filters:
    - parser:
        key_name: message
        remove_key_name_field: true
        parsers:
          - type: nginx
  localOutputRefs:
    - mys3-output
```

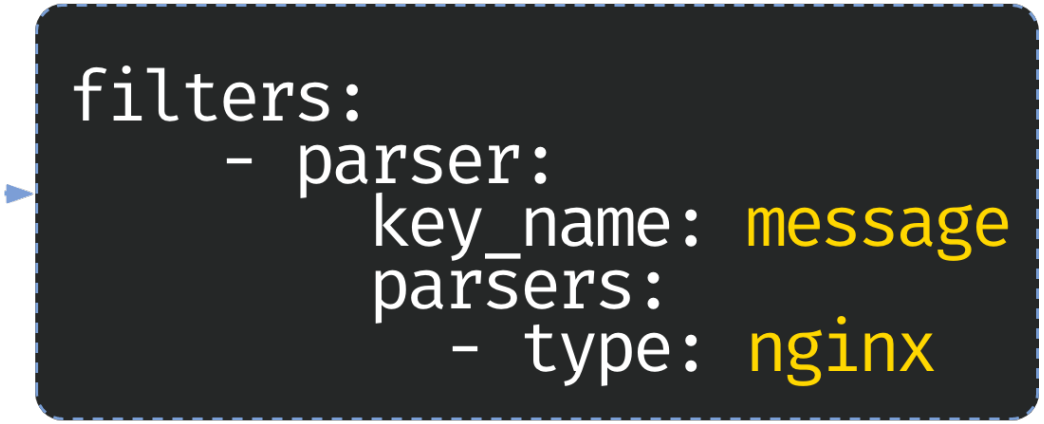
Flow vs ClusterFlow

```yaml
selectors: {}
```

```yaml
selectors:
    app: nginx
```

Logging Operator - Flow


```yaml
apiVersion: logging.banzaicloud.io/v1beta1
kind: Flow
metadata:
  name: geoip-sample
spec:
  selectors:
    app: nginx
  filters:
    - parser:
        key_name: message
        remove_key_name_field: true
        parsers:
          - type: nginx
  localOutputRefs:
    - mys3-output
```



Logging Operator - Flow


```yaml
apiVersion: logging.banzaicloud.io/v1beta1
kind: Flow
metadata:
  name: geoip-sample
spec:
  selectors:
    app: nginx
  filters:
    - parser:
        key_name: message
        remove_key_name_field: true
        parsers:
          - type: nginx
  localOutputRefs:
    - mys3-output
```

```yaml
outputRefs:
    - my-es-output
    - my-s3-output
```

Logging Operator - Output


```yaml
apiVersion: logging.banzaicloud.io/v1beta1
kind: Output
metadata:
  name: s3-output-sample
spec:
  s3:
    aws_sec_key:
      valueFrom:
        secretKeyRef:
          name: s3-secret
          key: awsSecretAccesKey
    s3_bucket: example-logging-bucket
    s3_region: eu-central-1
    path: logs/${tag}/%Y/%m/%d/
    format:
      type: json
```

Output vs ClusterOutput

```yaml
aws_sec_key:
  valueFrom:
    secretKeyRef:
      name: s3-secret
      key: awsSecretKey
```

Logging Operator – First release: August 2018

GitHub

900+ stars

slack

500+ community member

RANCHER®

Logging Operator is the officical logging backend

Notable features

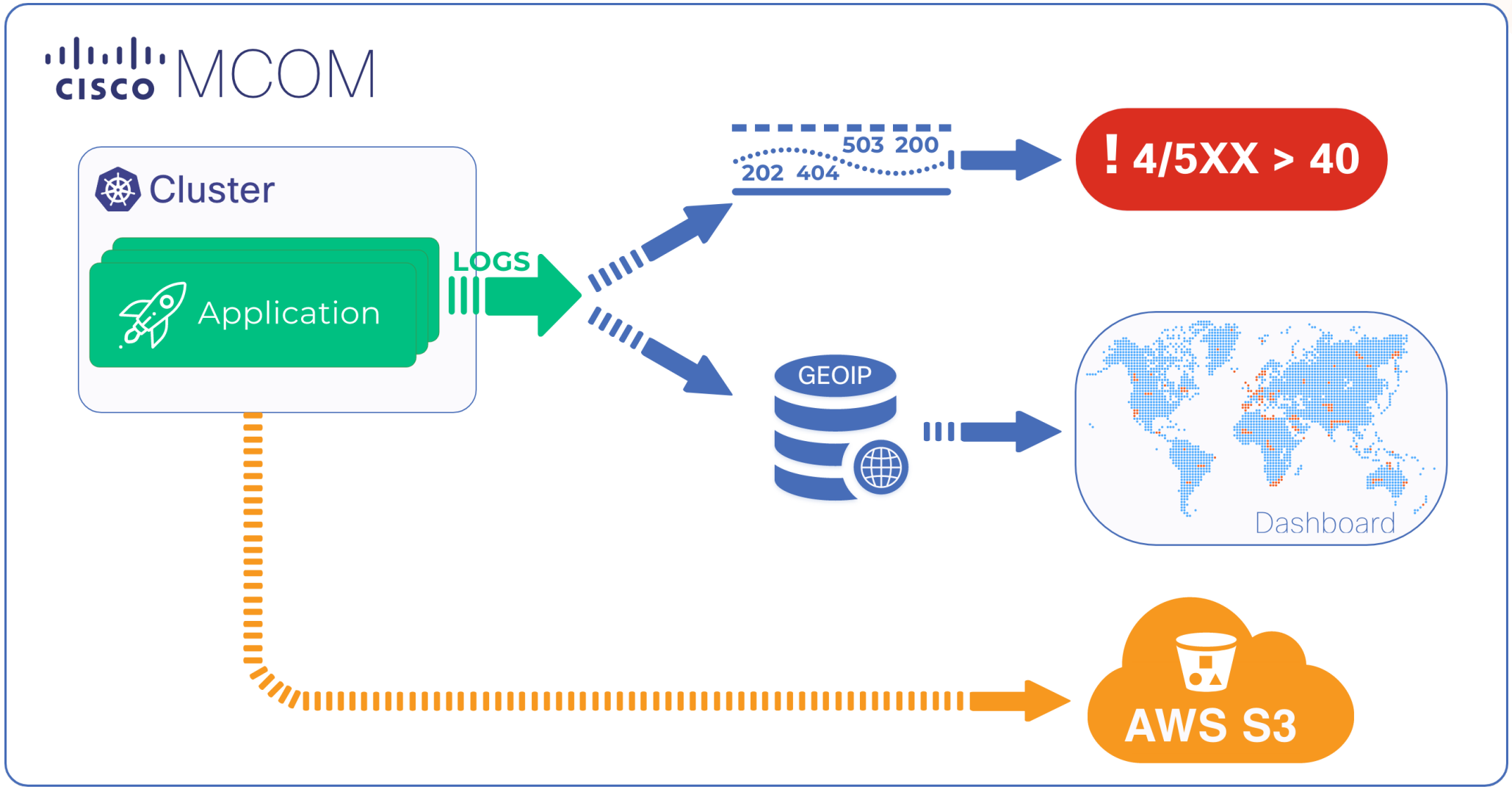
- Configuration check
- Scaling & Draining fluentd pods
- Scrape buffer metrics
- Secret & Cert management
- Webhook for sidecar tailing
- Tailing host logs

I can do this!
Hold my beer…

Let’s Help Bob!

- Bob has to run a legacy application on Kubernetes that's producing Nginx access logs.
- The **support team** needs an alert to trigger when the 5XX/4XX HTTP response codes reach more than 40% of the total requests.
- The **customer success** team requires a World Map that visualizes the origins of the clients.
- The **security** team wants all logs stored for at least 1 year without modification.

Let's Help Bob!



Let’s Help Bob!

- With this solution we satisfied all three groups
- We have response codes from access logs as metrics
- We visualized the source addresses on a World map
- We have an archive of all logs on an S3 bucket
- More over with the kubernetes labels it is almost trivial to correlate logs to metrics or traces

Are we happy?

There is still one thing that bothers Bob

Are we happy?

There is still one thing that bothers Bob

What if want to tail our logs?

Should we use **kubectl**?

Are we happy?

There is still one thing that bothers Bob

What if want to tail our logs?

Should we use **kubectl**?

- In a flow you see a normalized log with all the enrichment and filtering applied!
- With a flow you are able to tail different deployments or individual pods!

Just tail it...

So let's see how to tackle this task!

- We need a CLI where we can define what we want to tail…
- We need a service that can work as a fluentd output…
- That service needs to communicate with the CLI

Just tail it...

So let's see how to tackle this task!

- We need a CLI where we can define what we want to tail…
- We need a service that can work as a fluentd output…
- That service needs to communicate with the CLI

Do we want more?

Just tail it...

So let's see how to tackle this task!

- We need a CLI where we can define what we want to tail…
- We need a service that can work as a fluentd output…
- That service needs to communicate with the CLI

Do we want more?

Let's make it more interesting and add access control to those logs!

Just tail it... with RBAC

- Imagine a situation where you have many identical services but for different teams that manage them
- You want to specify who can access which pod
- There are several possible solutions and one of them is to use **labels**!
- Working example using RBAC based on labels is **Prometheus Label Proxy**

  https://github.com/prometheus-community/prom-label-proxy

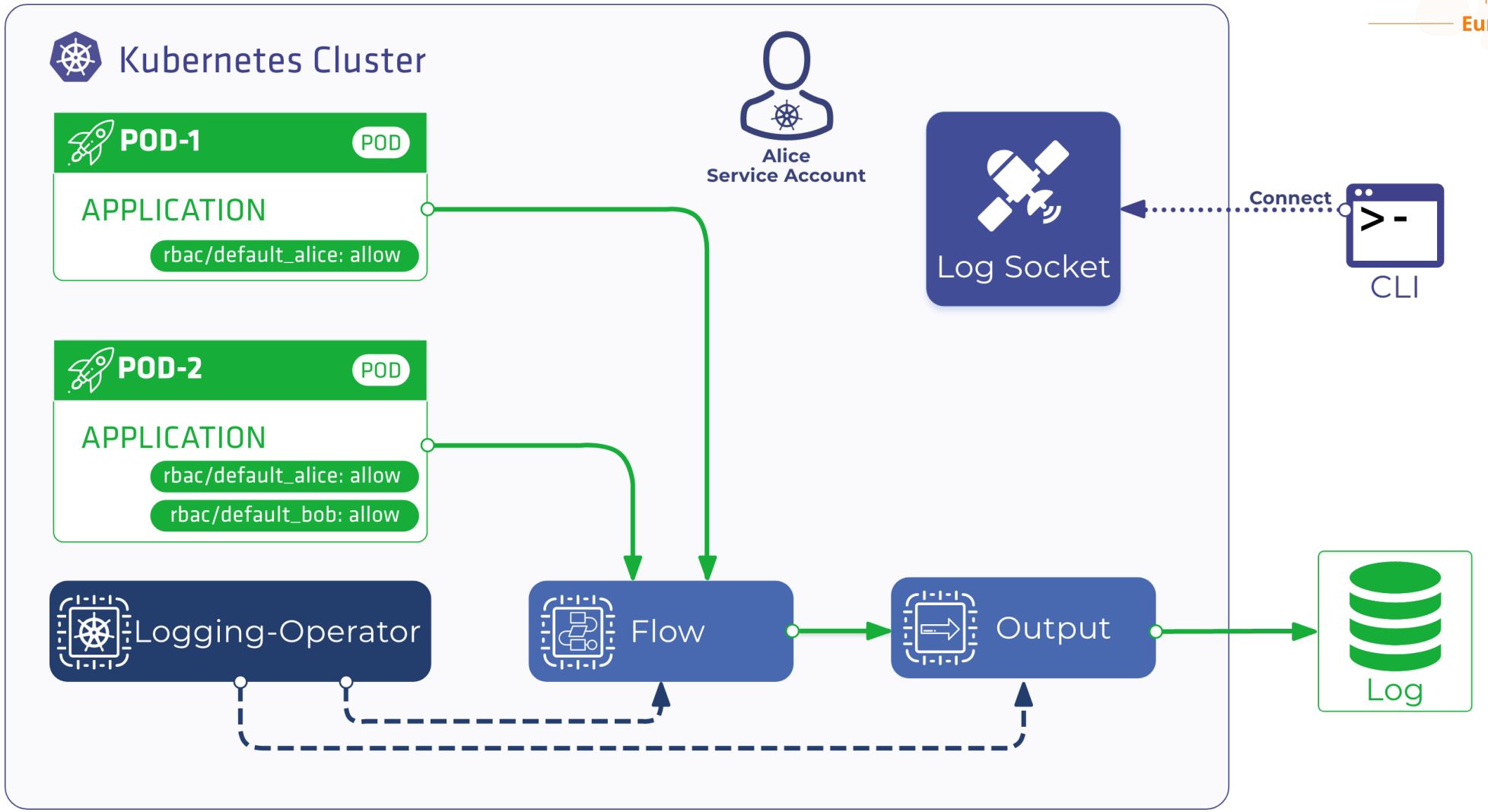
Just tail it... with RBAC

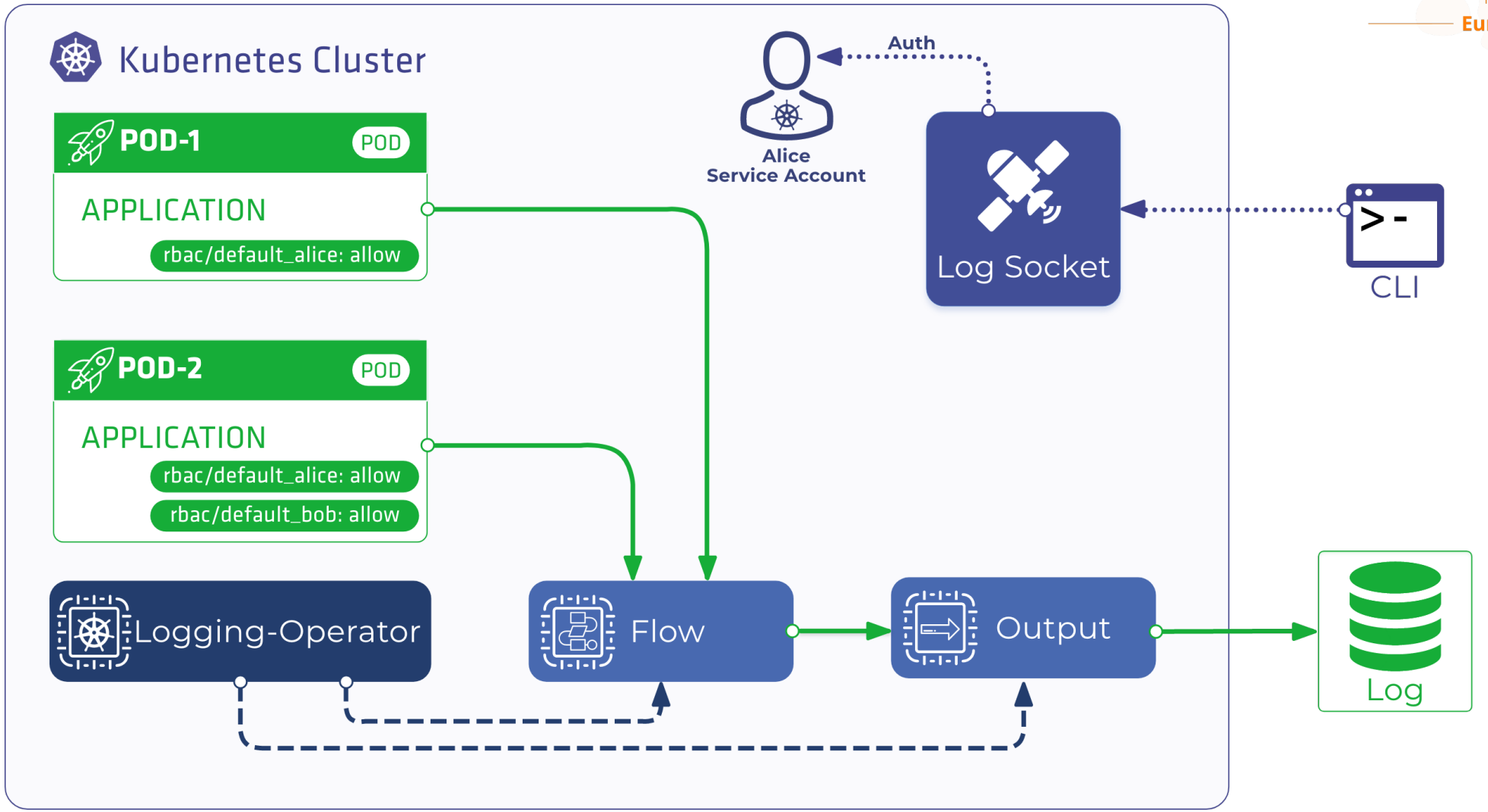
Let’s define the label structure is rbac/$namespace_$serviceaccount: $policy

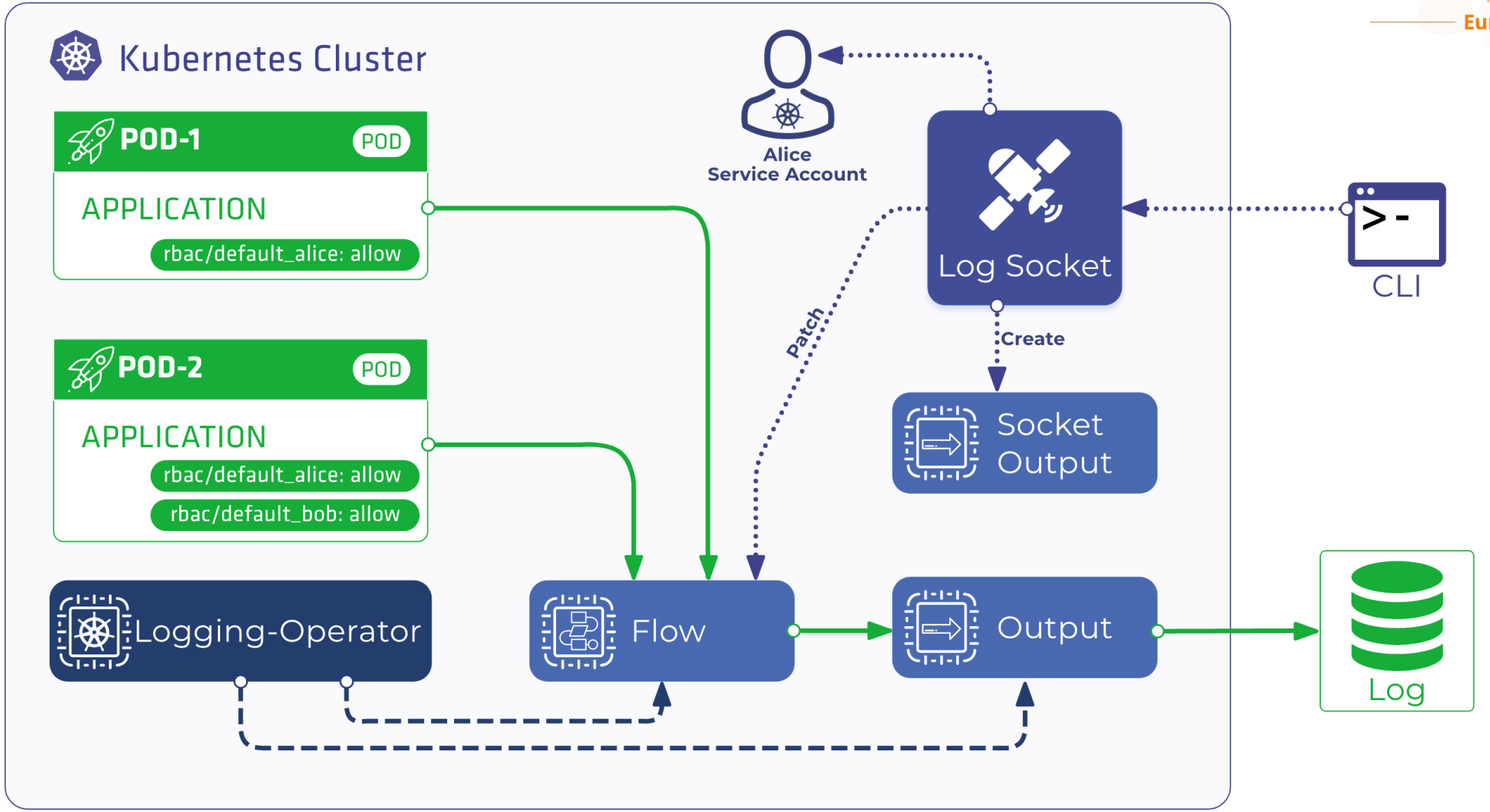
Examples:

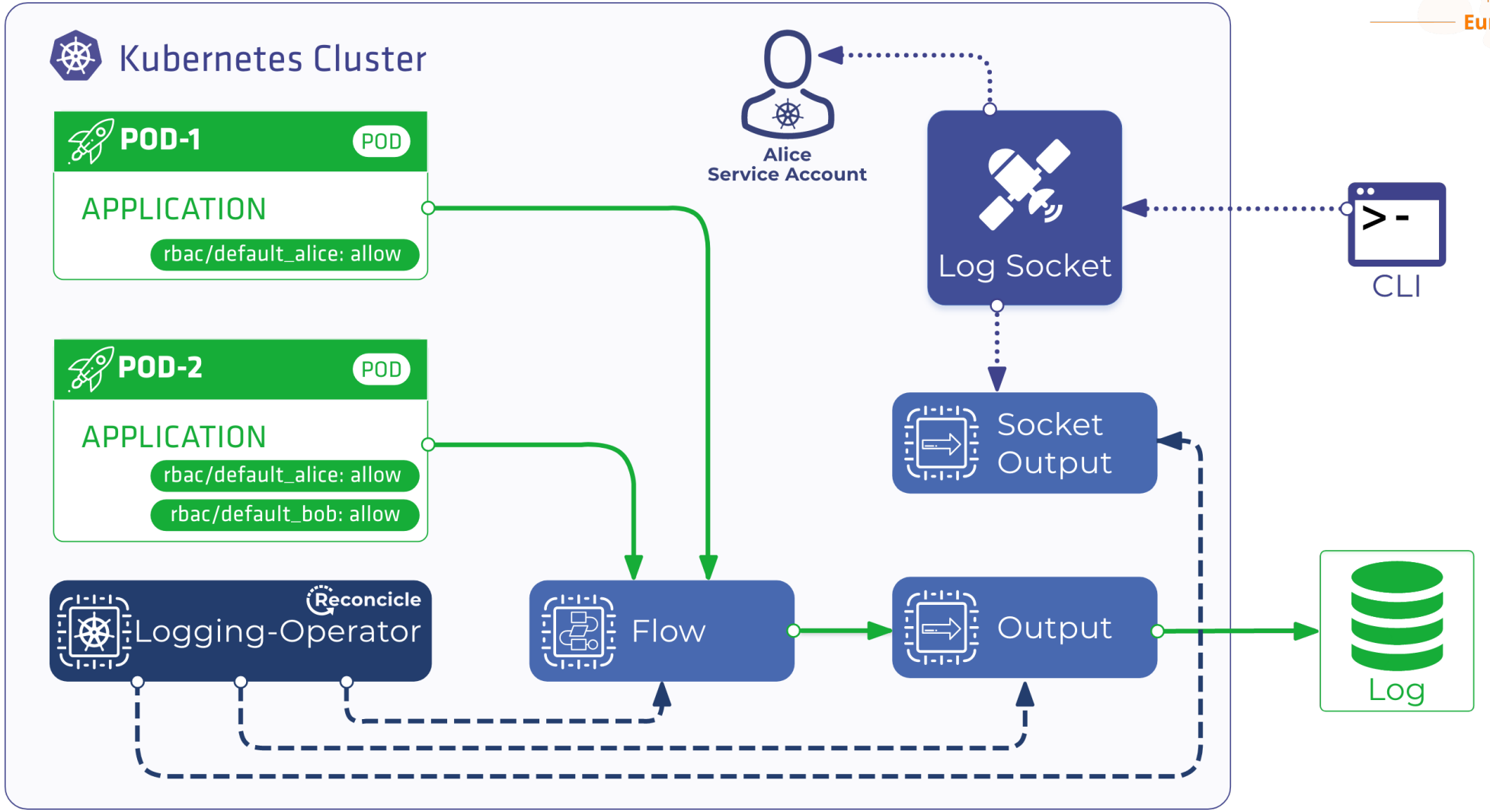
- **rbac/default_alice: allow**
- **rbac/default_bob: deny**
- **rbac/policy: allow (for default policy)**

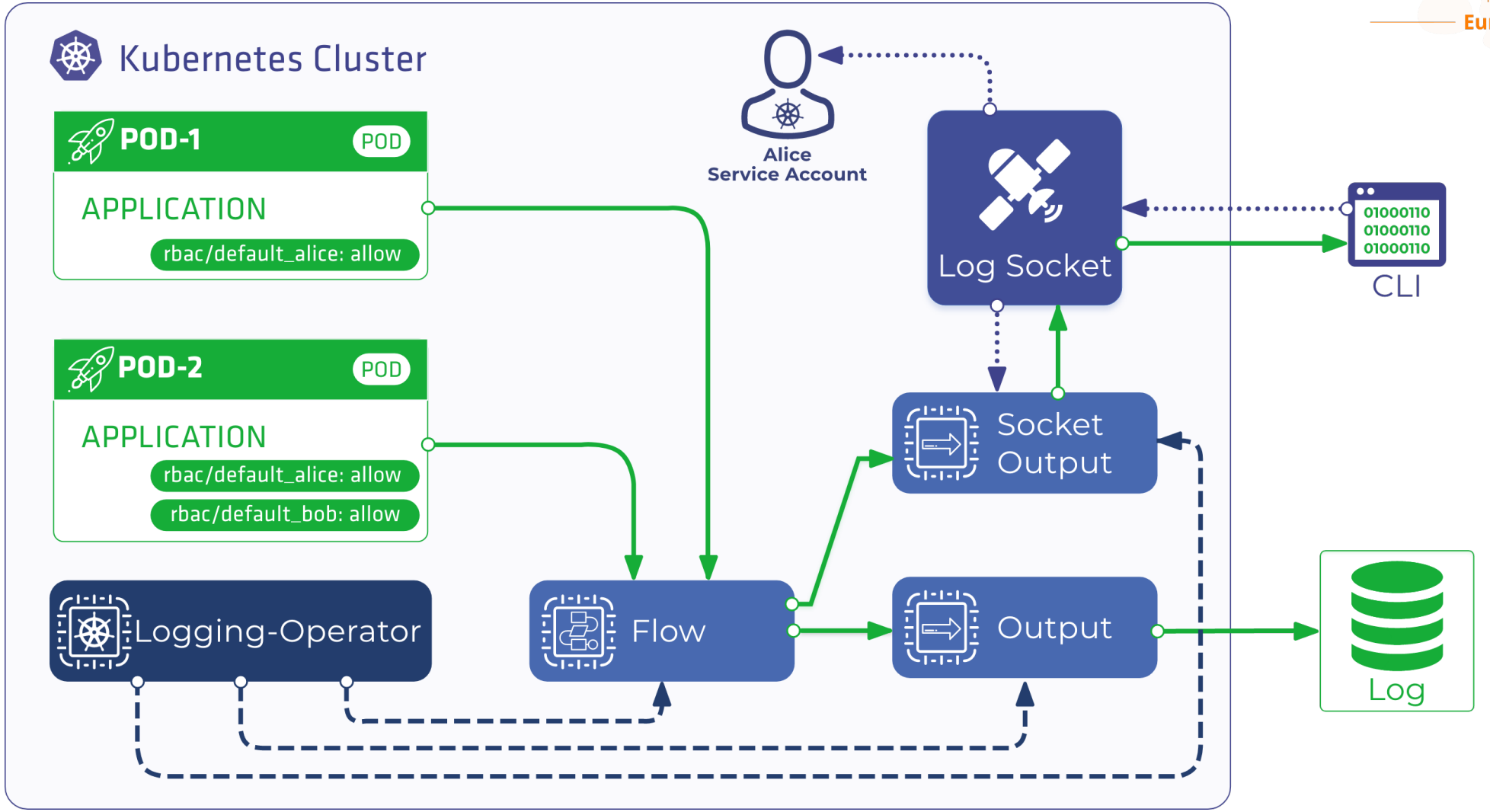
For simplicity we can use Kubernetes serviceaccount tokens to do the auth!


Kubernetes Cluster
POD-1
POD
APPLICATION
rbac/default_alice: allow
POD-2
POD
APPLICATION
rbac/default_alice: allow
rbac/default_bob: allow
Alice
Service Account
Log Socket
Connect
CLI
Logging-Operator
Flow
Output
Log


Kubernetes Cluster
POD-1
POD
APPLICATION
rbac/default_alice: allow
POD-2
POD
APPLICATION
rbac/default_alice: allow
rbac/default_bob: allow
Logging-Operator
Auth
Alice
Service Account
Log Socket
CLI
Flow
Output
Log


Kubernetes Cluster
POD-1
POD
APPLICATION
rbac/default_alice: allow
POD-2
POD
APPLICATION
rbac/default_alice: allow
rbac/default_bob: allow
Logging-Operator
Alice
Service Account
Log Socket
CLI
Patch
Create
Socket
Output
Flow
Output
Log


Kubernetes Cluster
POD-1
POD
APPLICATION
rbac/default_alice: allow
POD-2
POD
APPLICATION
rbac/default_alice: allow
rbac/default_bob: allow
Reconcicle
Logging-Operator
Alice
Service Account
Log Socket
CLI
Socket
Output
Flow
Output
Log


Kubernetes Cluster
POD-1
POD
APPLICATION
rbac/default_alice: allow
POD-2
POD
APPLICATION
rbac/default_alice: allow
rbac/default_bob: allow
Logging-Operator
Alice
Service Account
Log Socket
01000110
01000110
01000110
CLI
Socket
Output
Flow
Output
Log

Recap what we achieved!

- Complex logging flows that you can manage dynamically
- Tail flows on demand
- Apply RBAC rules on a Pod level

Future of the Logging Operator

Advanced routing and metadata

- Support special metadata like node labels, namespace labels
- Route based on log content

Apply RBAC rules to different log stream

Logging API

- Unify the experience of the Logging Operator
- Provides Authn and Authz
- Easier to use as component in a bigger system

There are several interesting projects emerge in the Observability space

- Open Telemetry Collector
- Tremor

Useful links and stuff

- Logging Operator - https://github.com/banzaicloud/logging-operator
- Kurun - https://github.com/banzaicloud/kurun
- Log-socket - https://github.com/banzaicloud/log-socket
- Cisco Tech Blog - https://techblog.cisco.com/tags/logging-operator
- https://banzaicloud.com/invite-slack/
- Demo - https://github.com/tarokkk/kubecon-valencia

Questions?