Building on-Premises MLOps for ISS Columbus Ground Operations

Samo Turk & Christian Geier
JUST ADD AI

Who Are We?

Samo Turk

- Data (Scientist|Engineer|Architect)
- Life science PhD, Pharma, Consulting
- Open source and Linux enthusiast
- Worked with K8s from convenience of cloud

Christian Geier

- Data (Scientist|Engineer|Architect)
- Ex-Physicist (PhD) and Data-Science Consultant
- Long time (Unix) nerd
- No previous exposure to K8s

We will be talking about

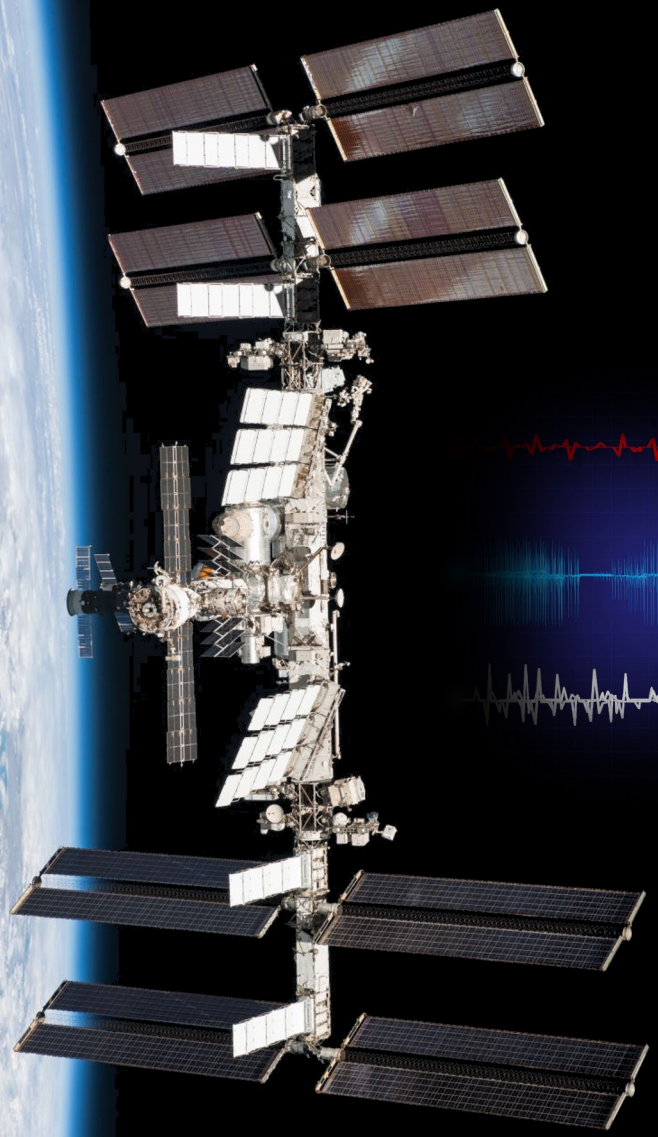
- The high-level overview of our use case
- Our approach designing and building an MLOps platform without the cloud
- The major components we use and why we use those
- Automated deployment and Infrastructure-as-Code
- Persistent storage
- Networking, TLS, and IAM & RBAC
- Logging and monitoring
- Tips and tricks, useful tools

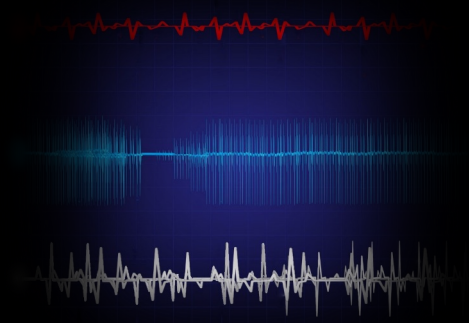
We won‘t be talking about

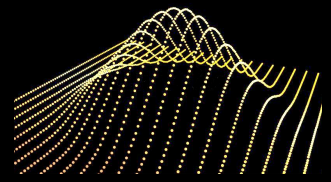
- Kubeflow details, running pipelines, or ML Jobs
- Anomaly detection, root cause analysis and reconfiguration suggestion algorithms
- Details of the ISS

(K)ISS

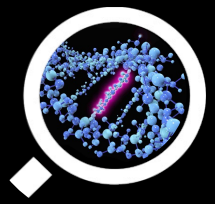
AI for the International Space Station



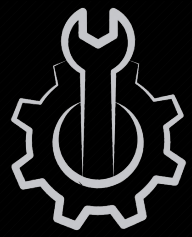




Anomaly Detection



Diagnostics



Reconfiguration

dtec.bw
Zentrum für Digitalisierungs- und Technologieforschung der Bundeswehr

Helmut Schmidt Universität
Universität der Bundeswehr Hamburg

Universität der Bundeswehr München

AIRBUS

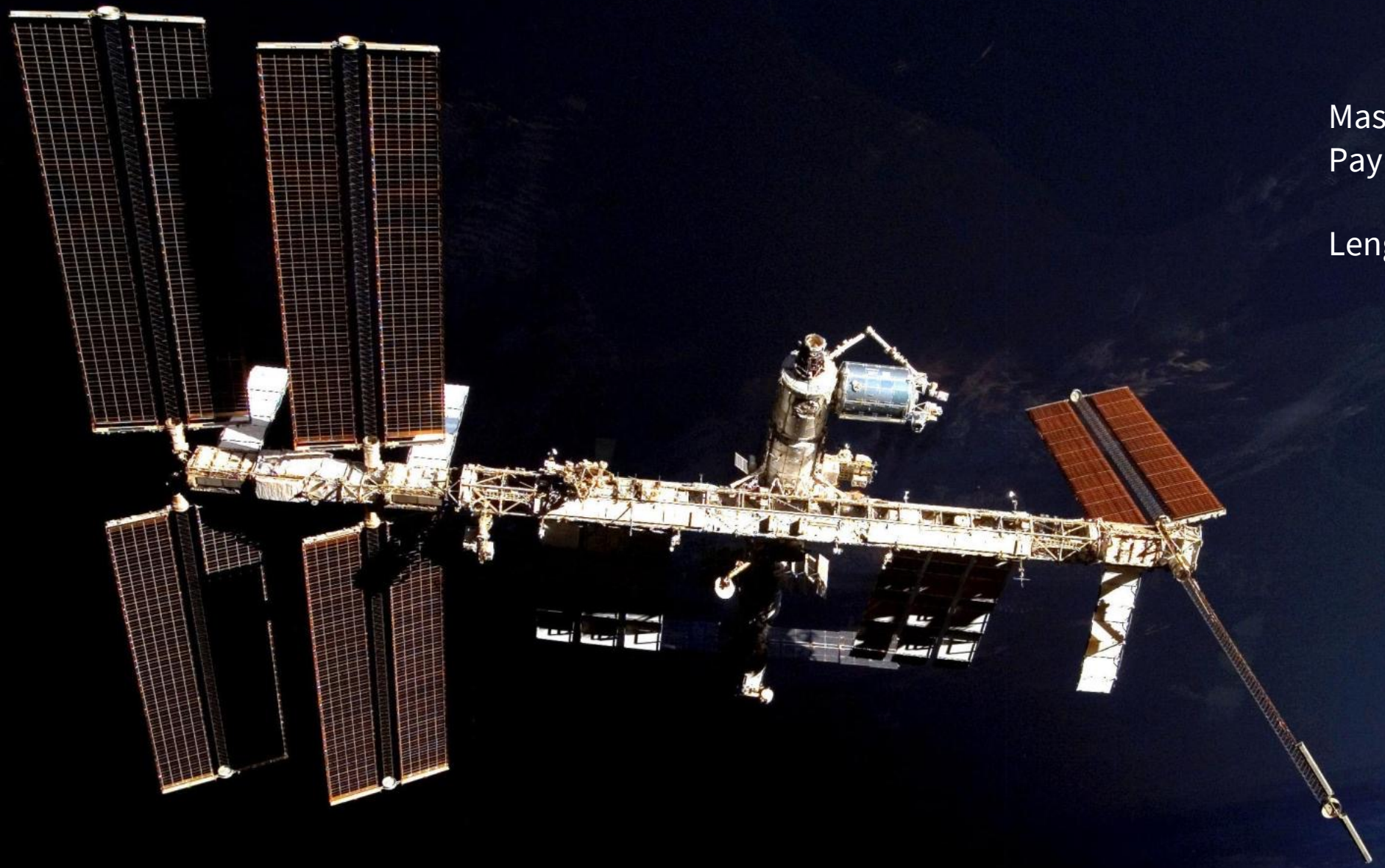
JAAI JUST ADD AI

ISS Columbus Module

Europe's first permanent outpost in orbit

View of the Columbus laboratory from the Space Shuttle Atlantis

Image Credit: NASA



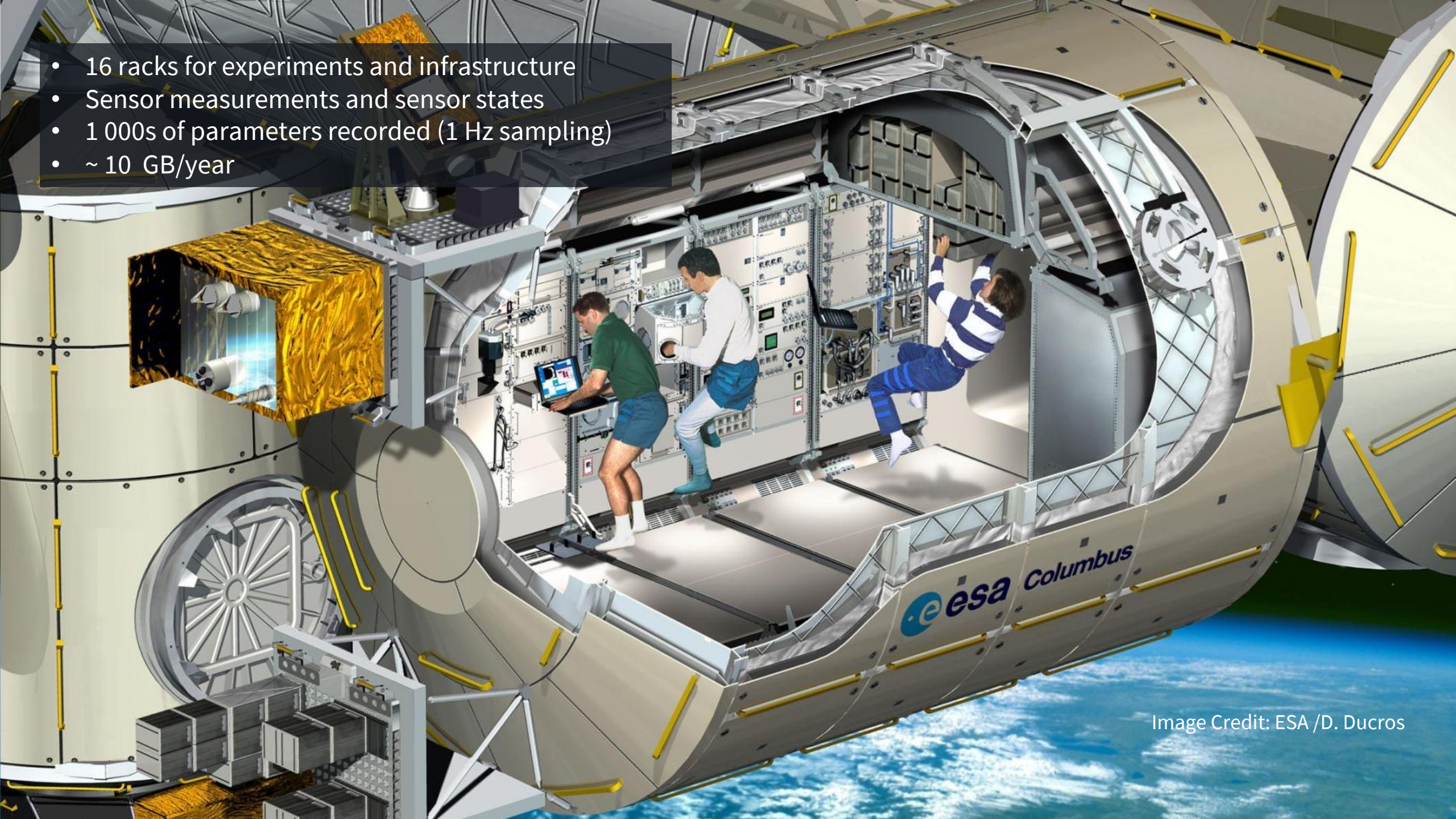
Mass: 10 000 kg
Payload: 9 000 kg

Length: 6.9 m

Image Credit: ESA/NASA

Built by Airbus
Launched 2008 on board of space shuttle Atlantis

Columbus on its way to Kennedy Space Center
Image Credit: NASA



- 16 racks for experiments and infrastructure
- Sensor measurements and sensor states
- 1 000s of parameters recorded (1 Hz sampling)
- ~ 10 GB/year

Image Credit: ESA /D. Ducros

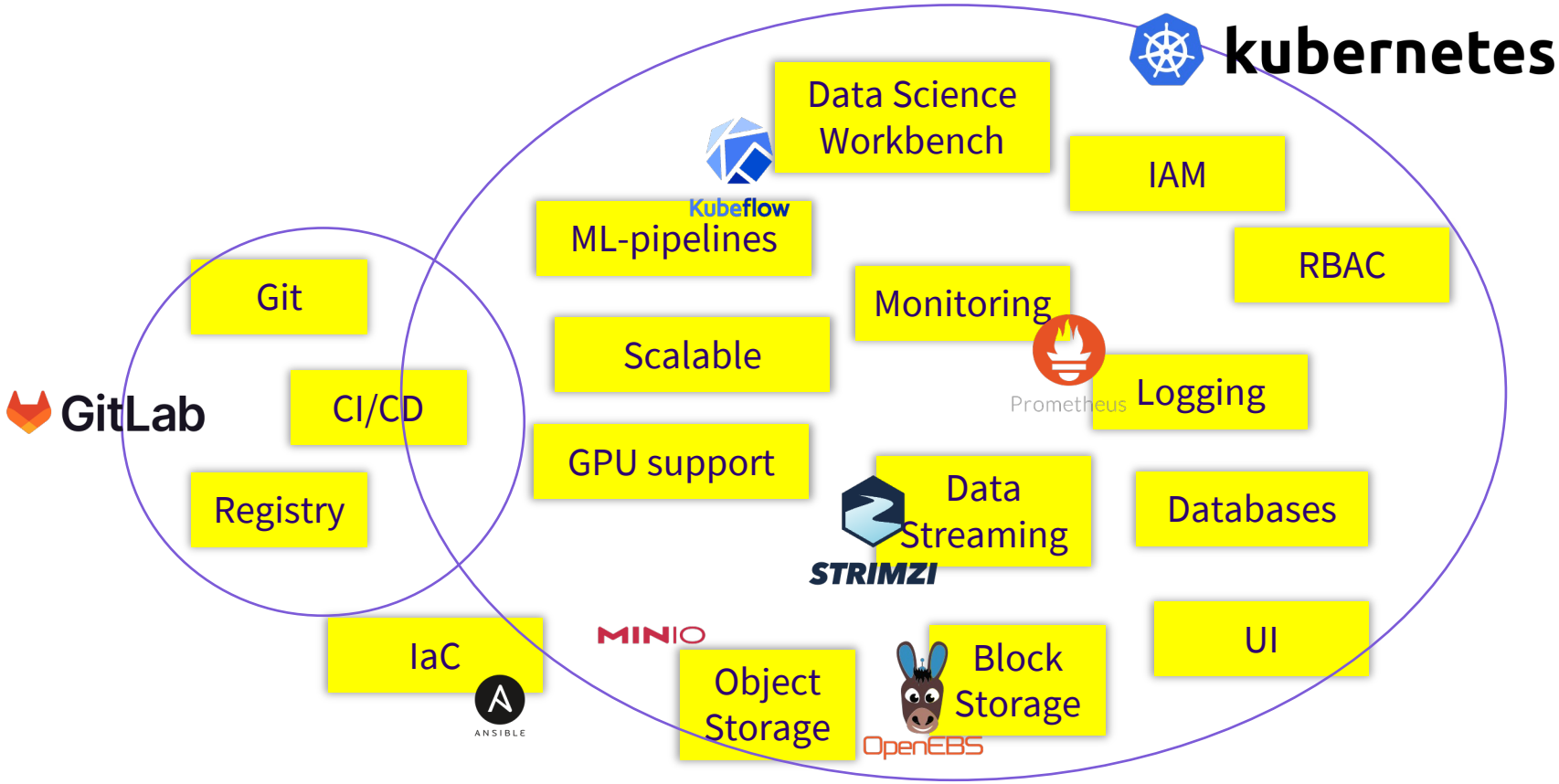
Data not allowed on public clouds

MLOps platform must run at all project partners' sites, meaning:

- Bare metal clusters
- Corporate data center
- Cloud (development)
- Different OS support needed

Only open source components

End-users are Data Scientist and Aerospace Engineers, not Software Engineers


kubernetes
Data Science Workbench
IAM
Kubeflow
ML-pipelines
RBAC
Git
Monitoring
Scalable
Prometheus
Logging
GitLab
CI/CD
GPU support
Data Streaming
Databases
Registry
STRIMZI
MINIO
IaC
UI
Object Storage
Block Storage
ANSIBLE
OpenEBS

Suddenly you need to take care of:

- Installing, configuring, and running K8s
- Storage
- IAM
- Loadbalancers
- TLS
- Monitoring
- …

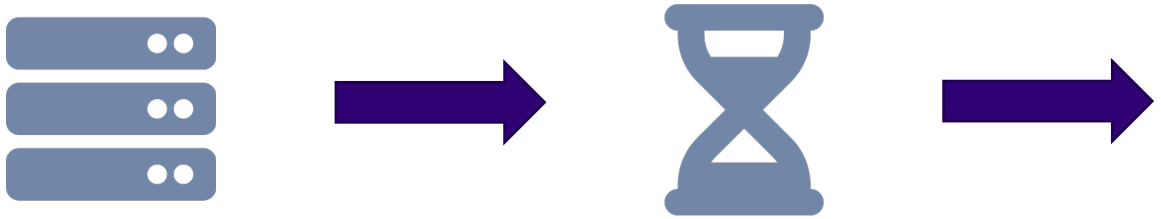
- GitLabCE is open source
- VCS i.e. git
- CI/CD
- Container registry

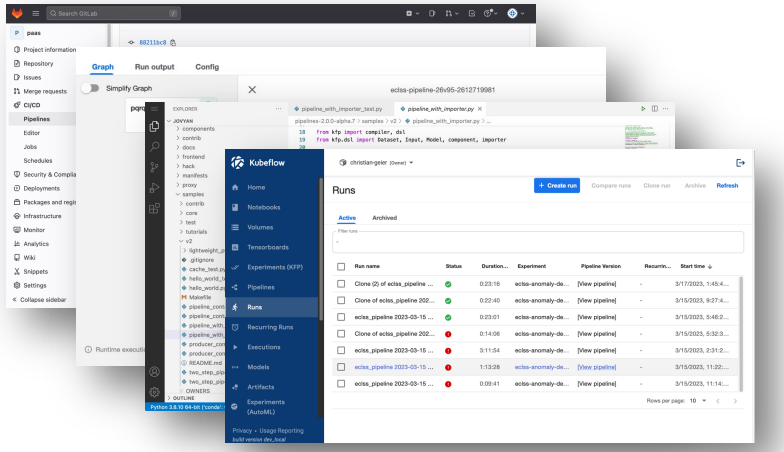
- Easy to deploy
- Works on most Linux distributions
- Supports multi-node
- CNCF certified
- Add-ons get you started very fast

- Complete data science workbench
- K8s native
- Mature and stable
- Actively developed

“heroism is not sustainable”*

* Kubernetes values: https://github.com/kubernetes/community/blob/master/values.md





Completely automated deployment

- Bare metal
- Corporate data center (proxy joys)
- Cloud

Why automation?

- Quick iterations (<1h for complete deployment)
- Reproducible
- Scalable

Components

- Ansible (IaC)
- GitLab CI/CD
- Kustomize

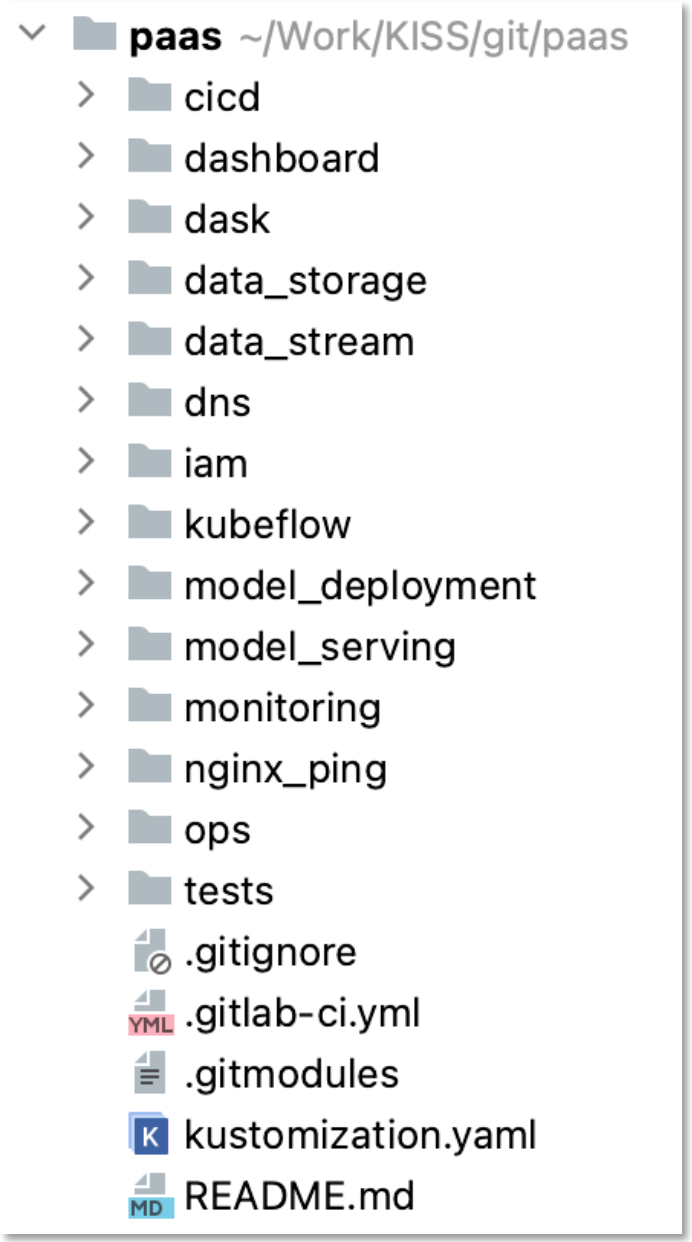
VS.

Imperative
Templating
Packages
Added abstraction
Not natively supported

Declarative
Built into kubectl
Works on plain yamls
Minimal
Overlays/patches

```
#!/bin/bash
for _ in {1..3}
do
  kubectl apply -k . || true
  echo "Retrying to apply resources"
  sleep 10
done
```

- Separate repository for all manifests (versions pinned)
  - official manifests or
  - rendered helm chart (`helm template`)
- Custom overlays
- GitLab's CI/CD to deploy (`kubectl apply -k .`)
  - Evaluating ArgoCD
- A lot of software comes with operators
- Custom operator to handle dynamic changes (e.g. new users)
  - Tip: shell-operator



PVC
Persistent Volume Claim

- is mounted to a pod
- behaves like any Unix volume
- namespaced

PV
Persistent Volume

- cluster resource
- created manually or dynamically

SC
Storage Class

- depends on storage engine/provisioner
- replication
- access mode

K8s was initially not developed for stateful workloads

OpenEBS hostpath

Mayastor for high availability deployments

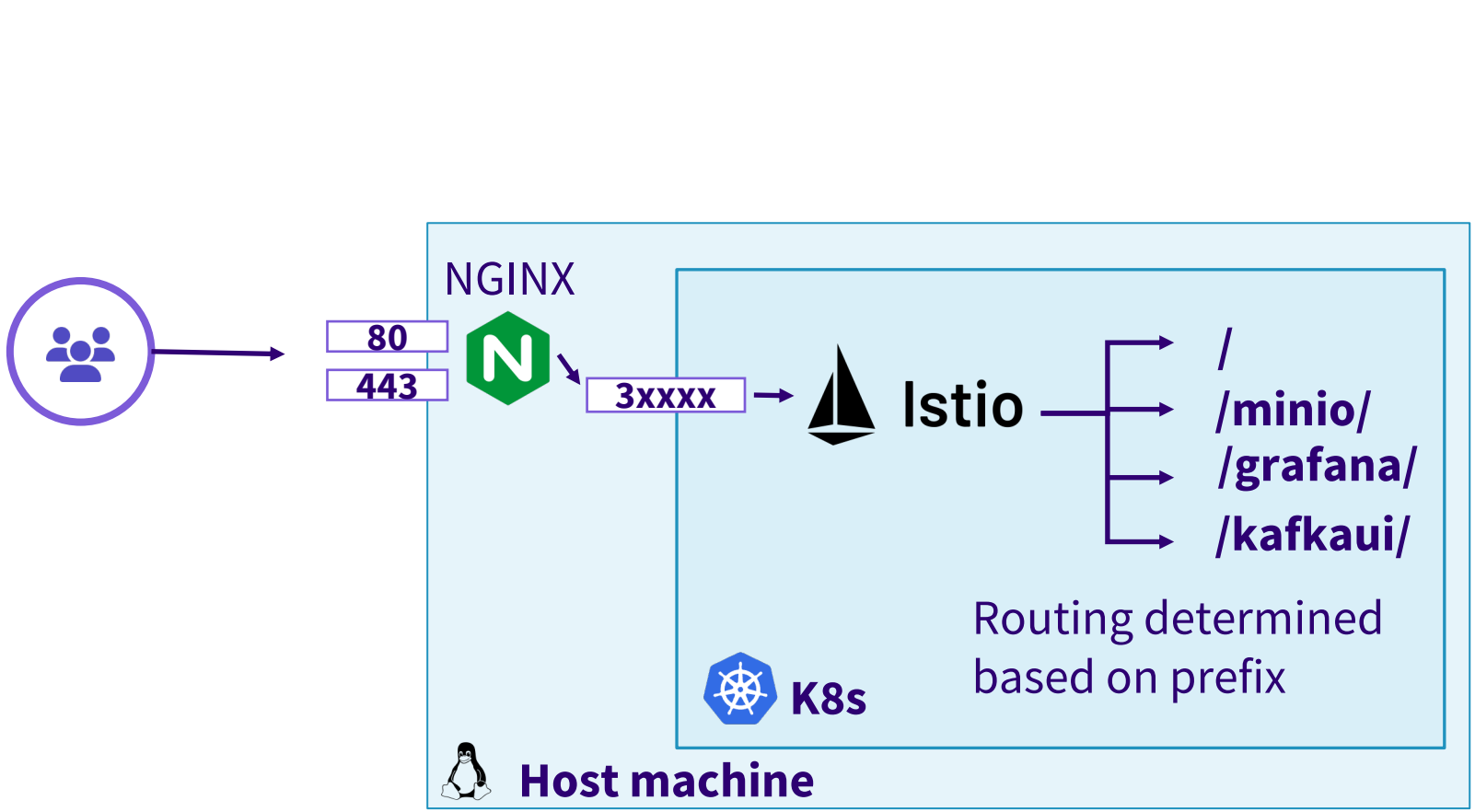
- Replication
- Needs minimum 3 nodes
- Hardware/performance requirements
- Dedicated block device or sparse file

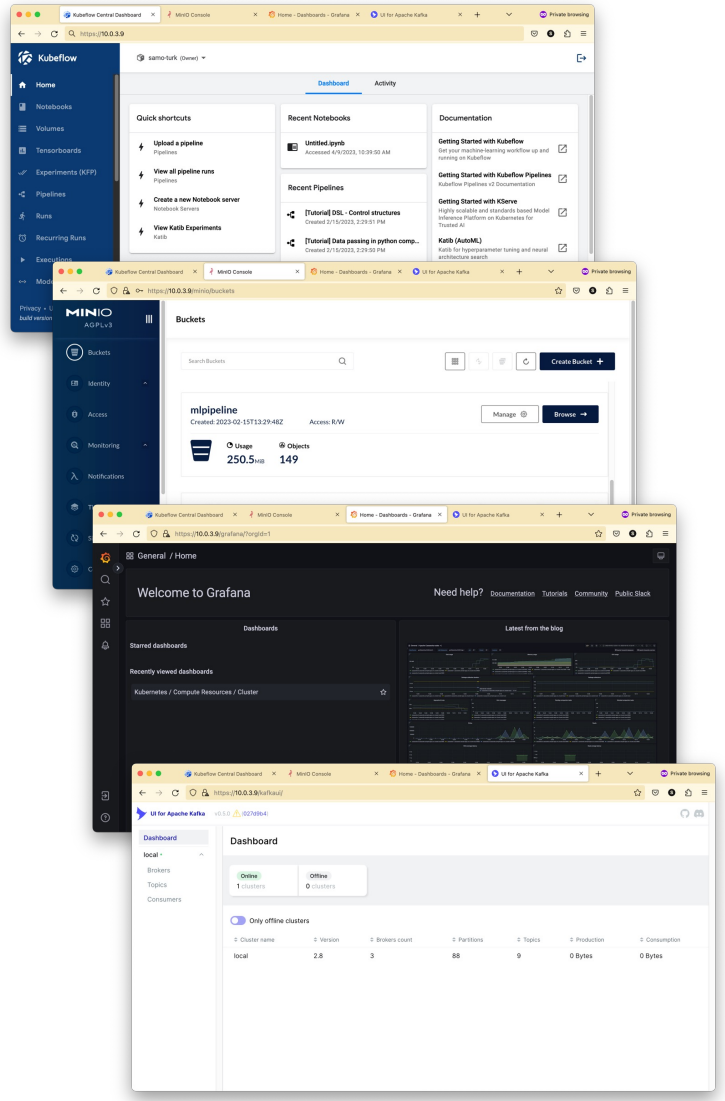
```
…/Work/KISS ❯ kubectl get sc
NAME                          PROVISIONER                    RECLAIMPOLICY   V
mayastor-two-replicas         io.openebs.csi-mayastor        Delete          W
openebs-hostpath              openebs.io/local               Delete          W
mayastor-one-replica (default)   io.openebs.csi-mayastor     Delete          W
```

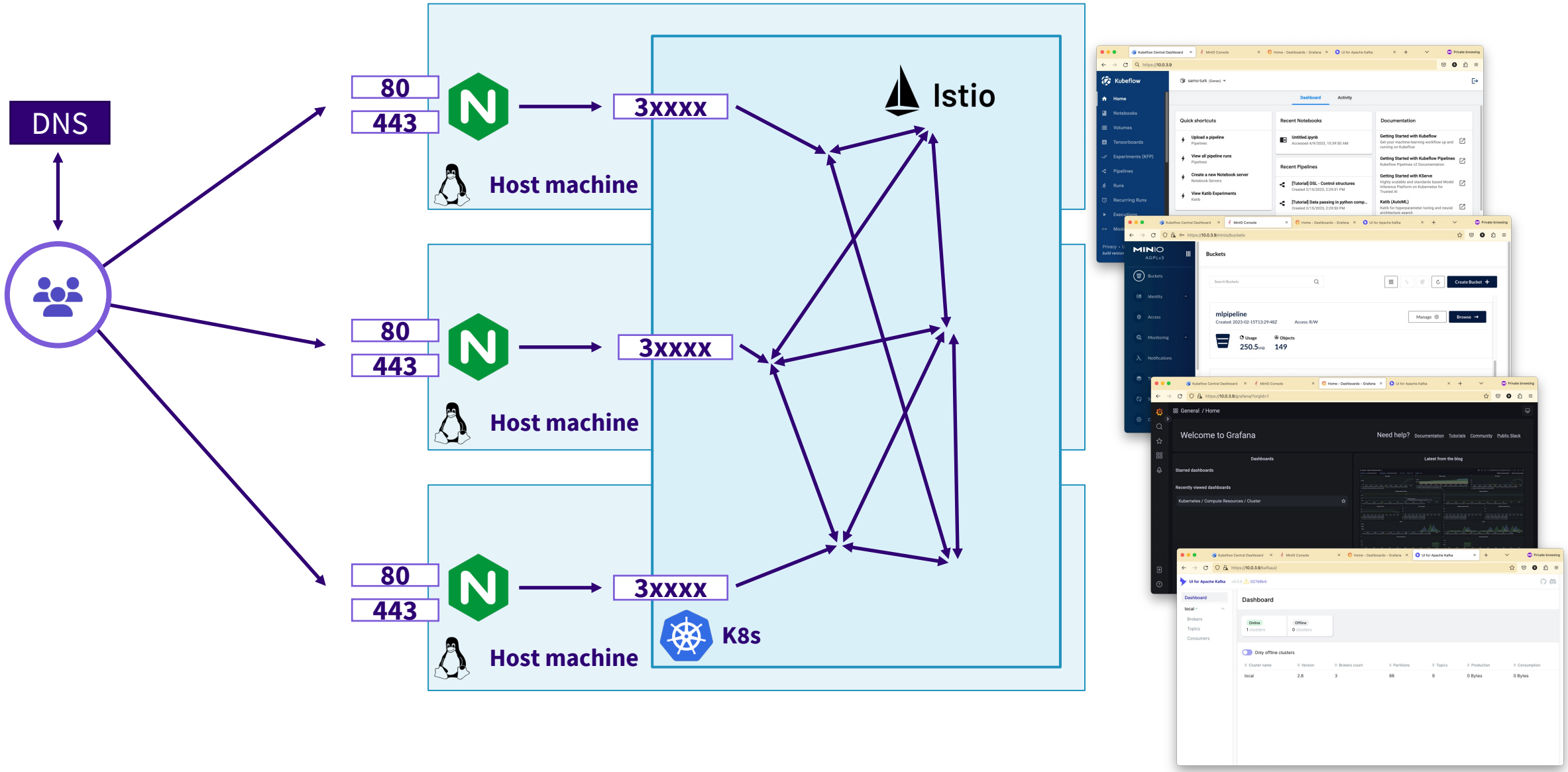
MINIO

Object storage

- S3 – compatible storage
- MinIO
- OpenEBS hostpath SC
- Does replication itself
- Older version bundled with Kubeflow
- Newer versions are under AGPL


NGINX
80
443
3xxxx
Istio
/
/minio/
/grafana/
/kafkaui/
Routing determined based on prefix
K8s
Host machine


Kubeflow
Buckets
mlpipeline
Welcome to Grafana
Dashboard


DNS
80
443
3xxxx
Host machine
80
443
3xxxx
Host machine
80
443
3xxxx
Host machine
Istio
K8s

TLS is important and important to get right.

TLS is hard and you might be tempted to disable it.

TLS is hard and you might be tempted to disable it. Don’t.

Don’t disable TLS*

- Security
- Some software refuses to run without TLS

Take the time to familiarize yourself with TLS!

- How certificate signing works?
- How your distribution of choice handles certificates?

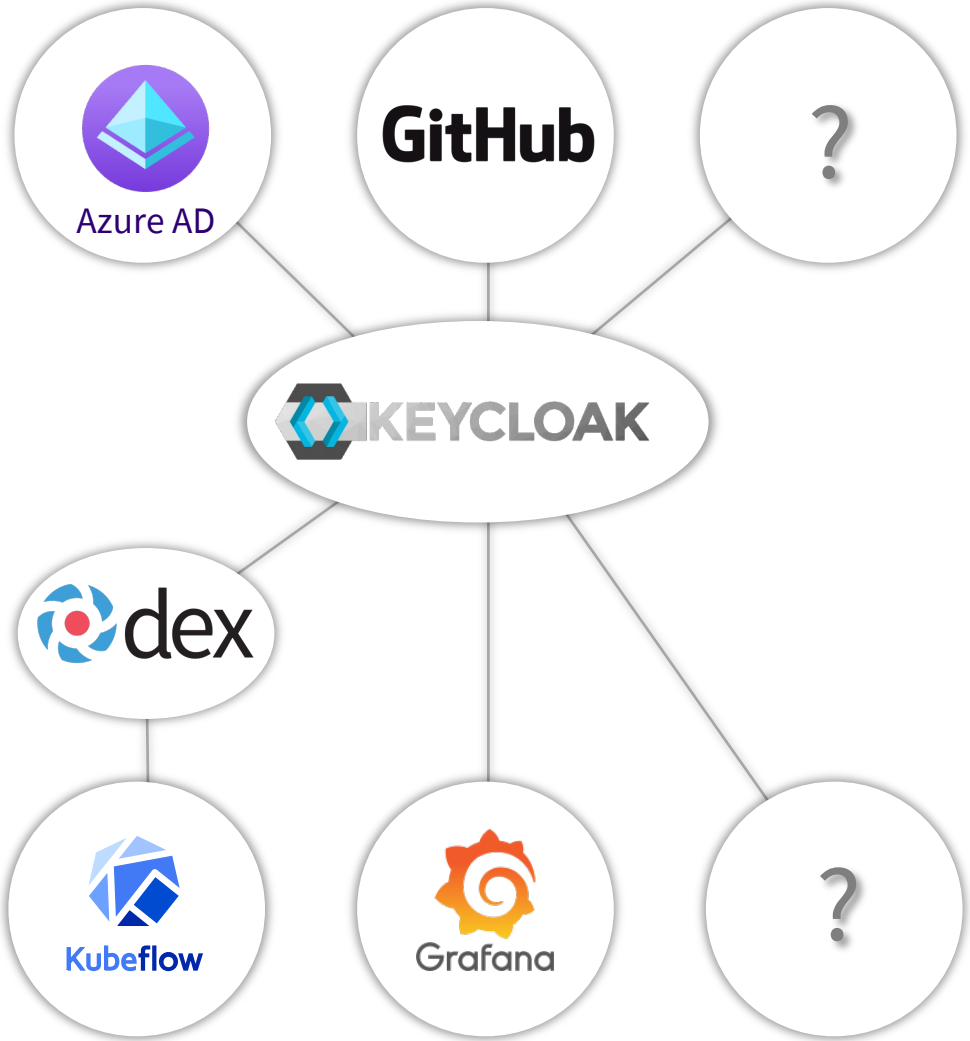
Certificate signing

- Use public certificate authorities (if possible)
- If self-signing, make sure to propagate your root certificate

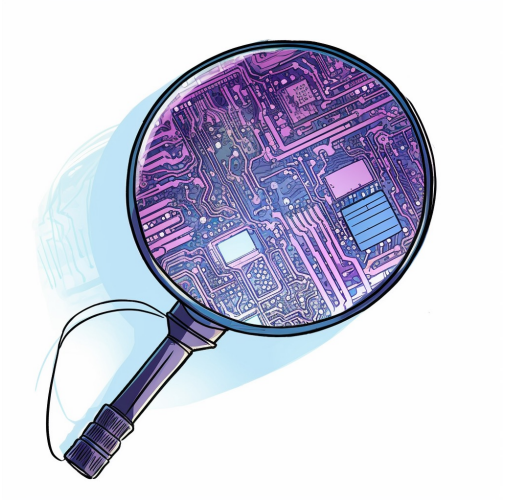
*It is potentially ok for services exposed through istio/intra cluster communication

Identity and Access Management (IAM)
≠
Role-Based Access Control (RBAC)

- Kubeflow comes with dex for IAM
- Keycloak is more fully featured and better documented
- You can use dex & Keycloak



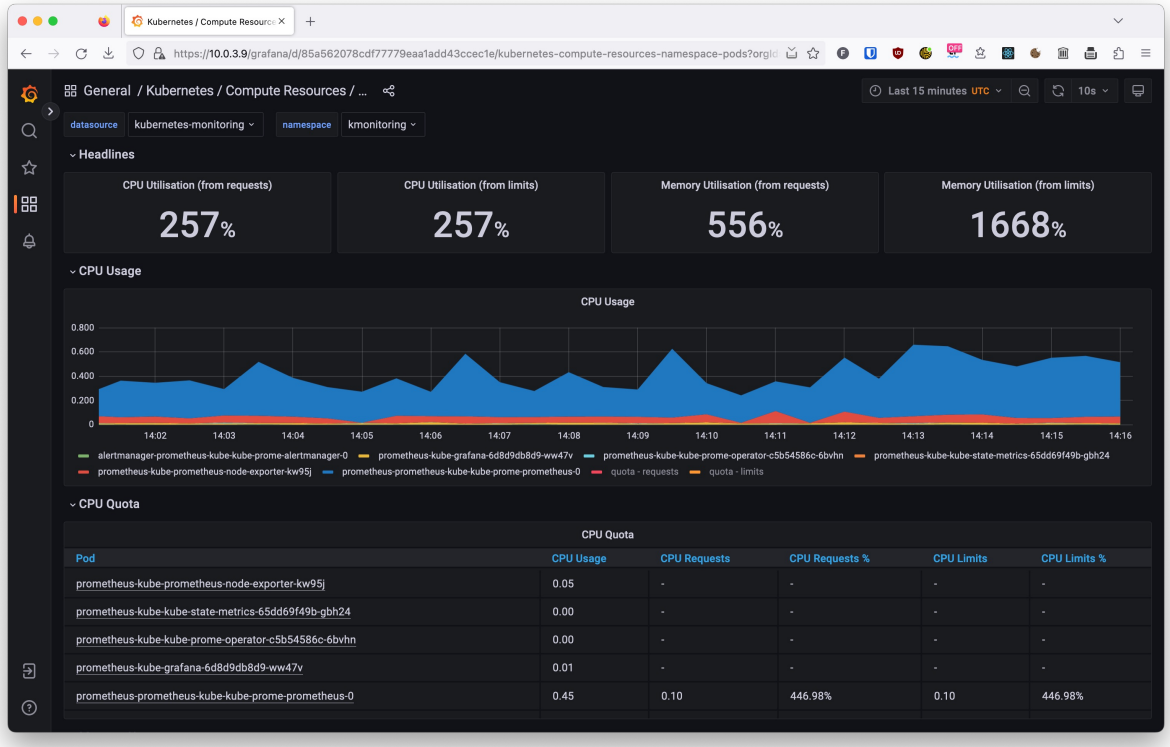
What‘s going on in your cluster?

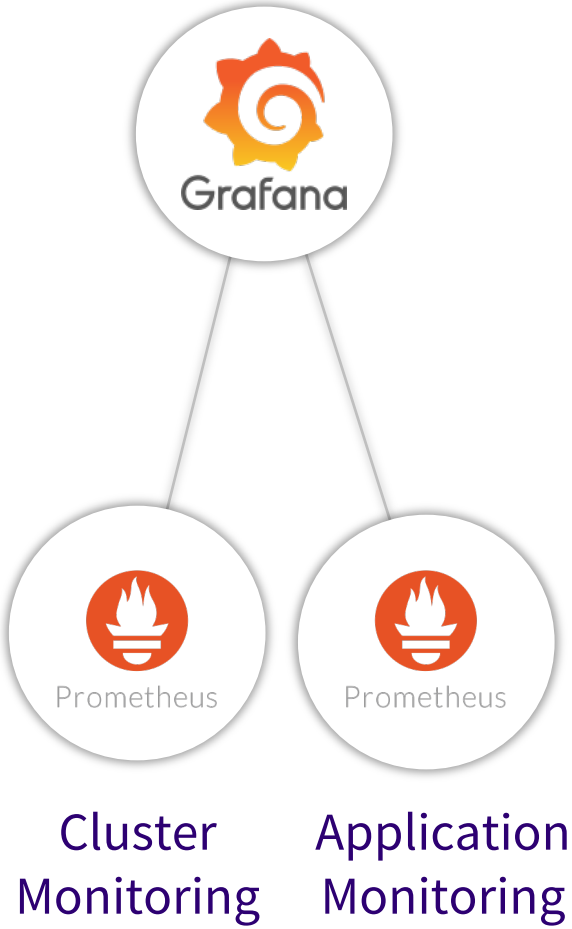


- Prometheus for monitoring
- Grafana as frontend

- Kube-prometheus for monitoring K8s
  - Comes with everything, including dashboards
  - Metrics API Support

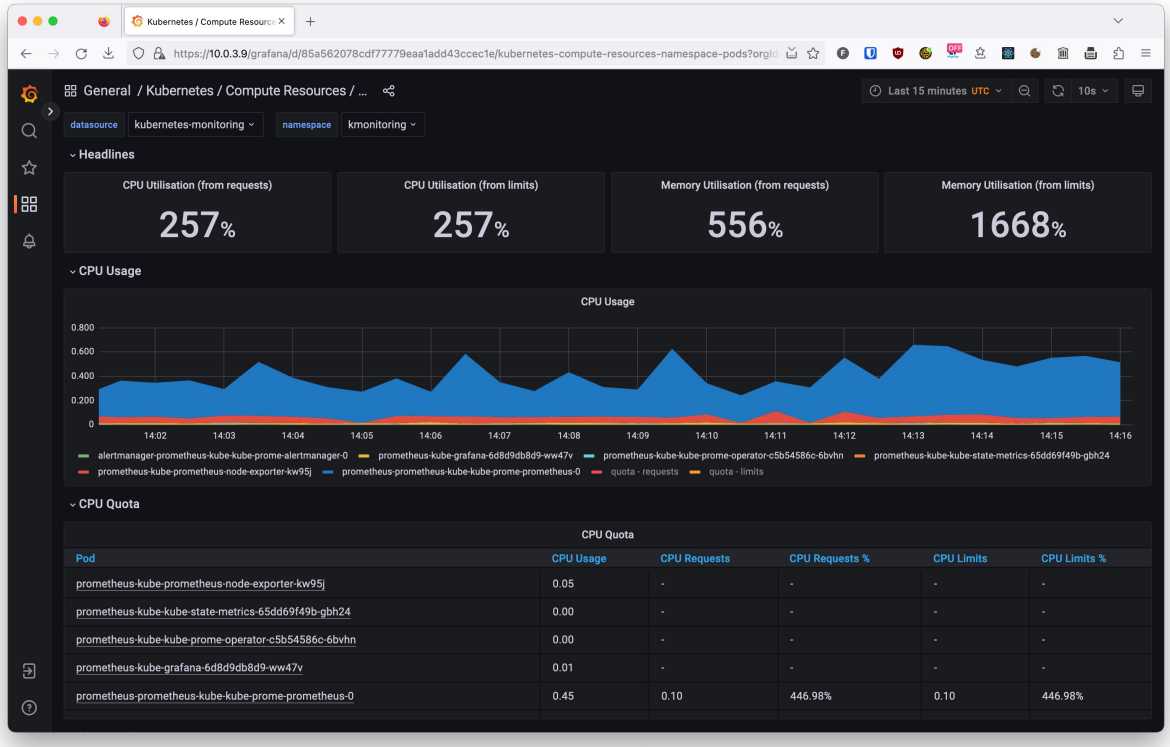
- Second Prometheus for application monitoring
- Separately installed Grafana (`helm template`)

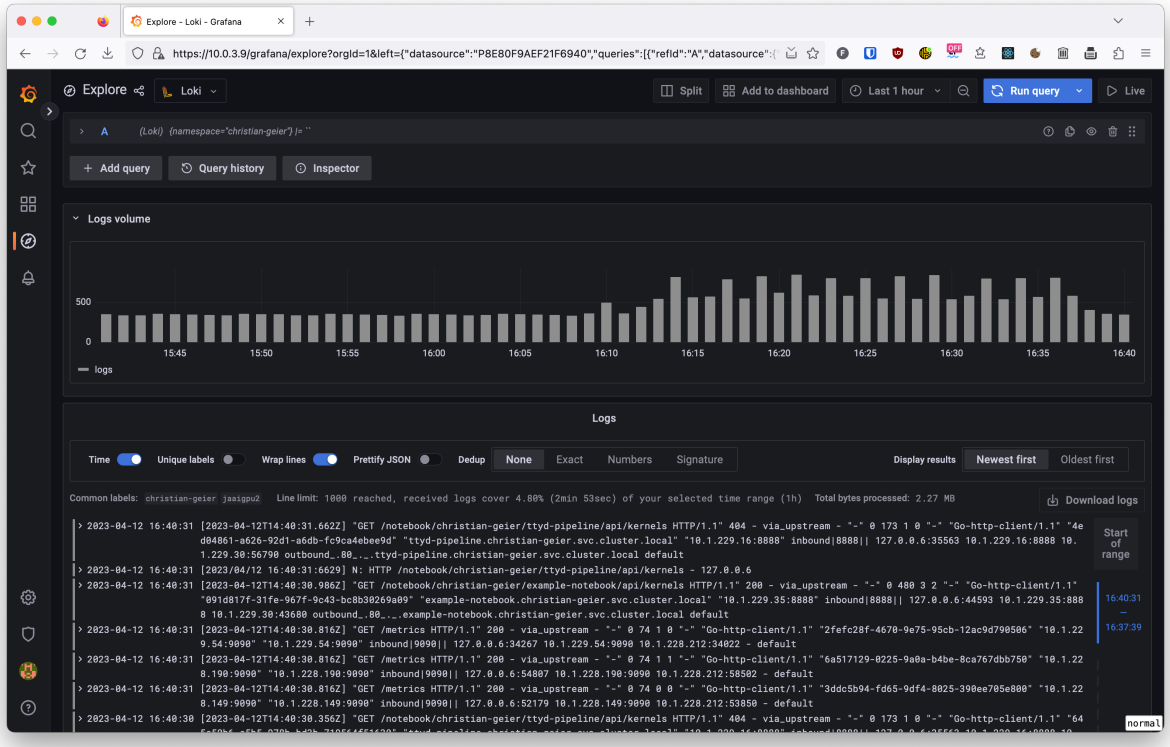




Cluster Monitoring

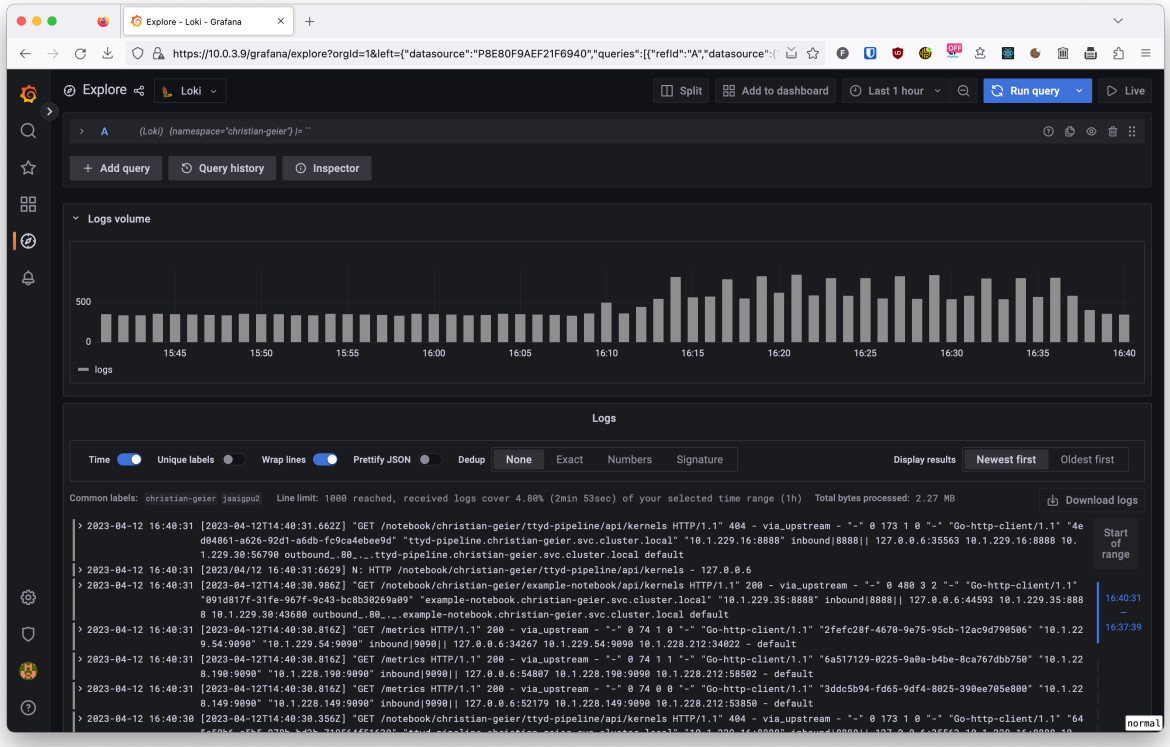
Application Monitoring





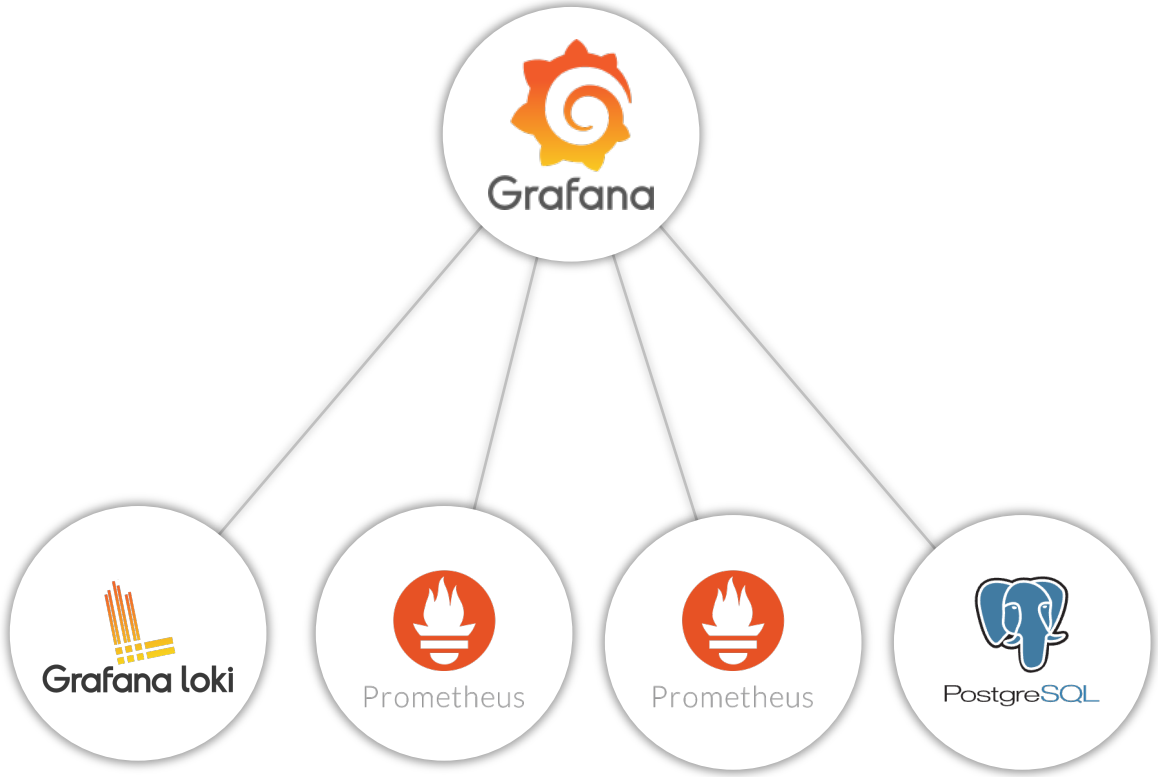
Logging is as important as monitoring

(if not more so)



- Used by cluster-admins and end-users
- Derived from PLG helm-chart (no Grafana)

- Logging and monitoring tied together by Grafana
- Loki, Prometheus, Grafana installed separately



Kubeflow is great!

Kubeflow is flexible!

Kubeflow is *very* flexible!

Debugging

- `kubectl logs`
- `kubectl exec`
- `k9s`
- (Open)Lens

K8s yaml validator for your IDE

`jq` and `yq`, json and yaml processors are very useful

Check hardware requirements

Check version requirements

Tests with bests Bash Automated Testing System

Shell-Operator

Set limits (can’t spawn 10k jobs)

PostgreSQL needs hugepages

```yaml
resources:
  limits:
    hugepages-2Mi: 500Mi
    memory: 500Mi
```

- Automate as much as possible, including testing
- Don’t be afraid of asking for help
- Find the right mix between experimentation and learning the basics

It is hard but doable even with a small team

Acknowledgements

Questions?

Don’t forget to rate this talk!