



Andrii Salnikov
on behalf of IT Infra

Kubernetes @ MAX IV

as an integral part of IT Infrastructure





SYNCHROTRON
LIGHT SOURCE

“Kubernetes, also known as K8s, is an open-source system for automating deployment, scaling, and management of containerized applications”

[©kubernetes.io](https://kubernetes.io)

“Automating deployment”

- It's a **Kubernetes Cluster**
 - multiple nodes (resilience)
 - storage provisioning
 - backups
 - access control
 - metrics and logs collection
- **GitOps**
 - CI/CD pipelines
 - Impersonation (Tokenizer by *Dmitrii Ermakov*)
- **Staged rollouts**
 - prod/dev, blue/green, etc



Generated with AI · Image Creator in Bing

“Kubernetes strictly ensures that all the containers are always in the desired state using set of Controllers”

“Automating management”

- Proper **multi-tenancy**
 - by design without root
- Web-UI **Dashboards** and **CLI**:
 - monitor, restart, read logs, debug shell, etc
- **Resource usage** monitoring
- **High availability** by-design
 - health-checks
 - automatic failover
 - topology constraints
- **Abstraction layers** and **self-service**
 - no address/ports pre-allocation
 - no specific pre-provisioning
- **API** to manage containers



Generated with AI Image Creator in Bing

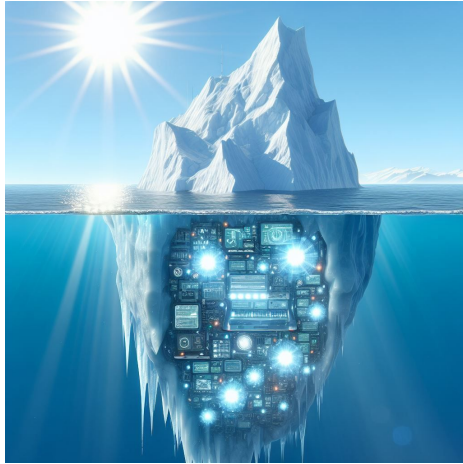
“Automating scaling”

- **Hardware scalability**
 - easy to expand/replace nodes
- **Environment homogeneity**
 - no need to manually sync software layer, permissions, etc
- **Workloads scalability**
 - automatic scaling based on resource utilization
- **Human efforts scalability**
 - more efforts to do initial deployment, *but minimal efforts to reproduce*
 - create another workload instance in almost zero time
 - simplified/unified Day2 Ops



Generated with AI - Image Creator in Bing

At what cost?



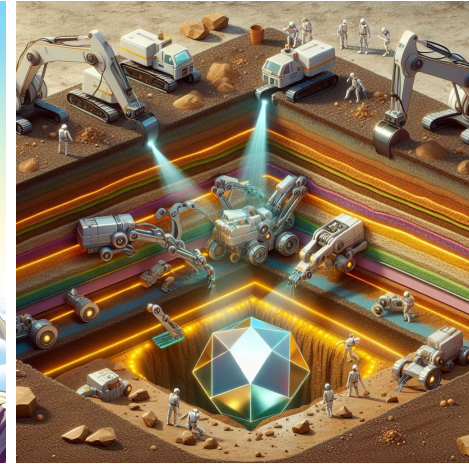
Generated with AI · Image Creator in Bing



Generated with AI · Image Creator in Bing



Generated with AI · Image Creator in Bing



Generated with AI · Image Creator in Bing

Level of complexity

requires high level of expertise at IT Infrastructure

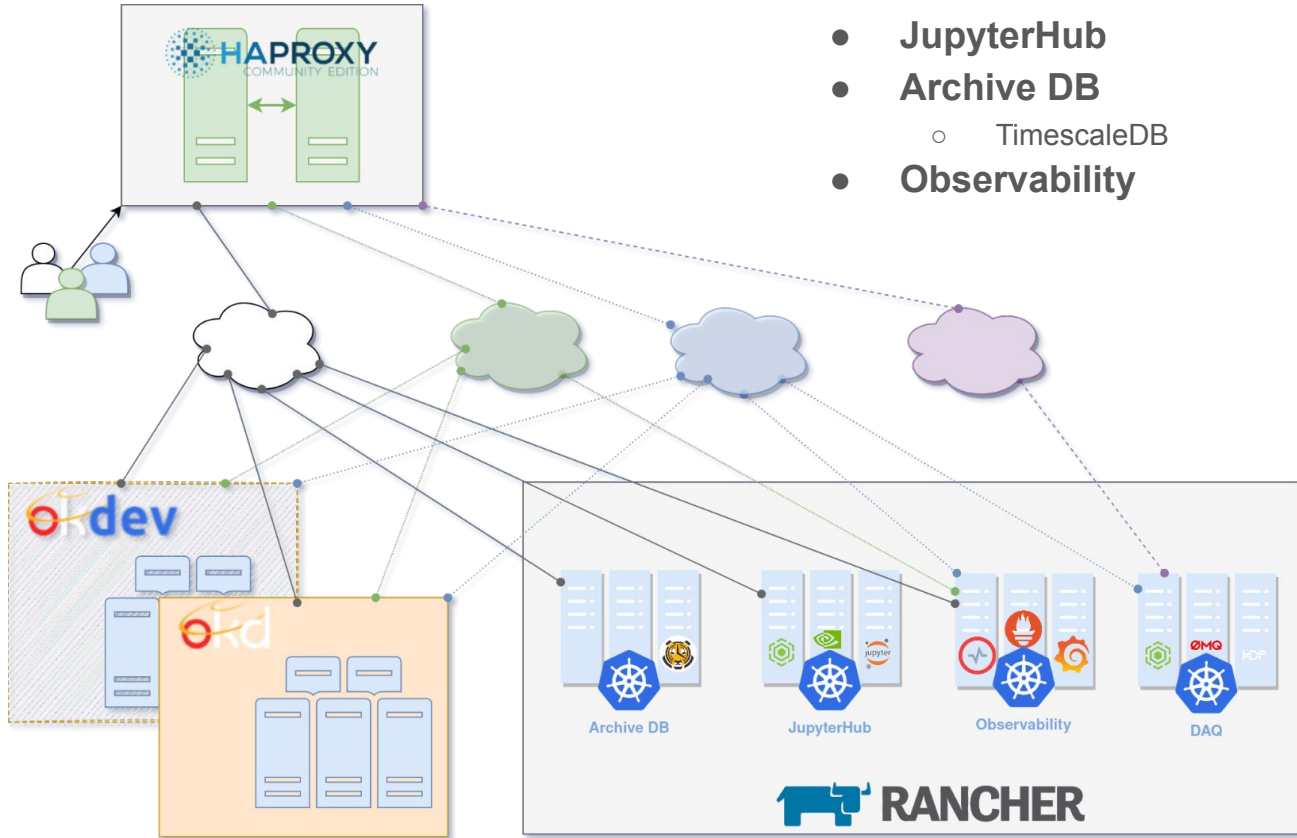
Breaking **bad habits** efforts

Steep Fear of **learning curve**

“Workload is somewhere there”

... *but with minimal efforts to reproduce* :)

Which Kubernetes?



- DAQ
- JupyterHub
- Archive DB
 - TimescaleDB
- Observability

- OKD + OKDev
 - Web-applications
 - webpage
 - wiki
 - taranta
 - taiga
 - metabase
 - cabledb
 - archviewer
 - archwizard
 - hdbppviewer
 -
 - AWX
 - Artifactory
 - Harbor
 - Kafka
 - Keycloak SSO
 - Gitlab Runners

What is OKD?



kubernetes

- **container orchestration platform**
- community supported
- core framework
- user is responsible for integrations beyond core



OPENSIFT

- **container orchestration ecosystem**
- RedHat supported
- extends core functionality with more abstractions, **security**, dashboard, etc
- integration of common CI/CD features

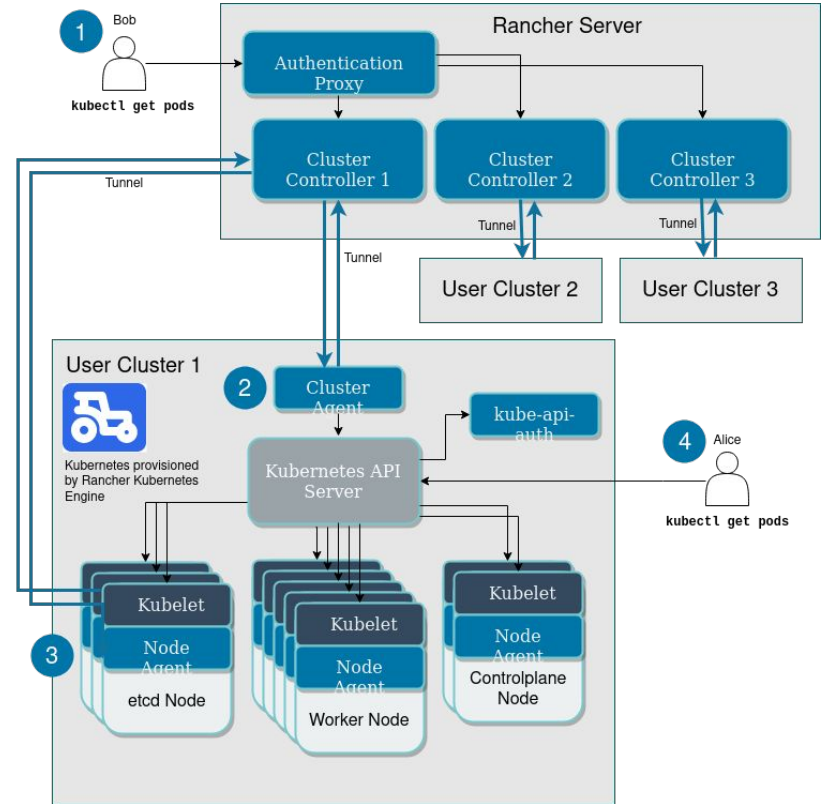
okd

- **community upstream** for OpenShift
- community supported
- formerly “*OpenShift Origin*”
- OKD is not an acronym, just OKD

What is Rancher?



- “Rancher lets you deliver **Kubernetes-as-a-Service**”
- Dedicated service *in front of* **Kubernetes** to manage:
 - cluster provisioning, authN/authZ, security provisioning, dashboard
 - “standard” applications automated provisioning (monitoring/logging)
 - kubernetes itself is just **pure core K8s** without extra fancy controllers (contrary to OKD)
- Rancher “**management plane**” is kind of proxy to K8s cluster

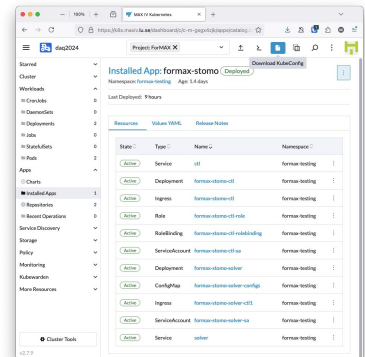
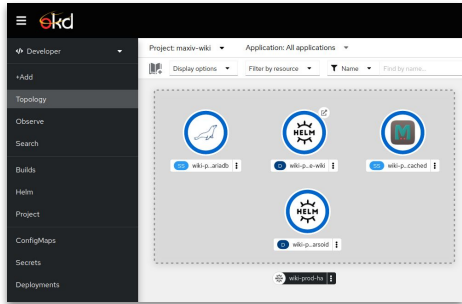


Why we use both?

Full-blown ecosystem with **security hardening** for lifecycle management of *web-apps* and *infrastructure workloads*



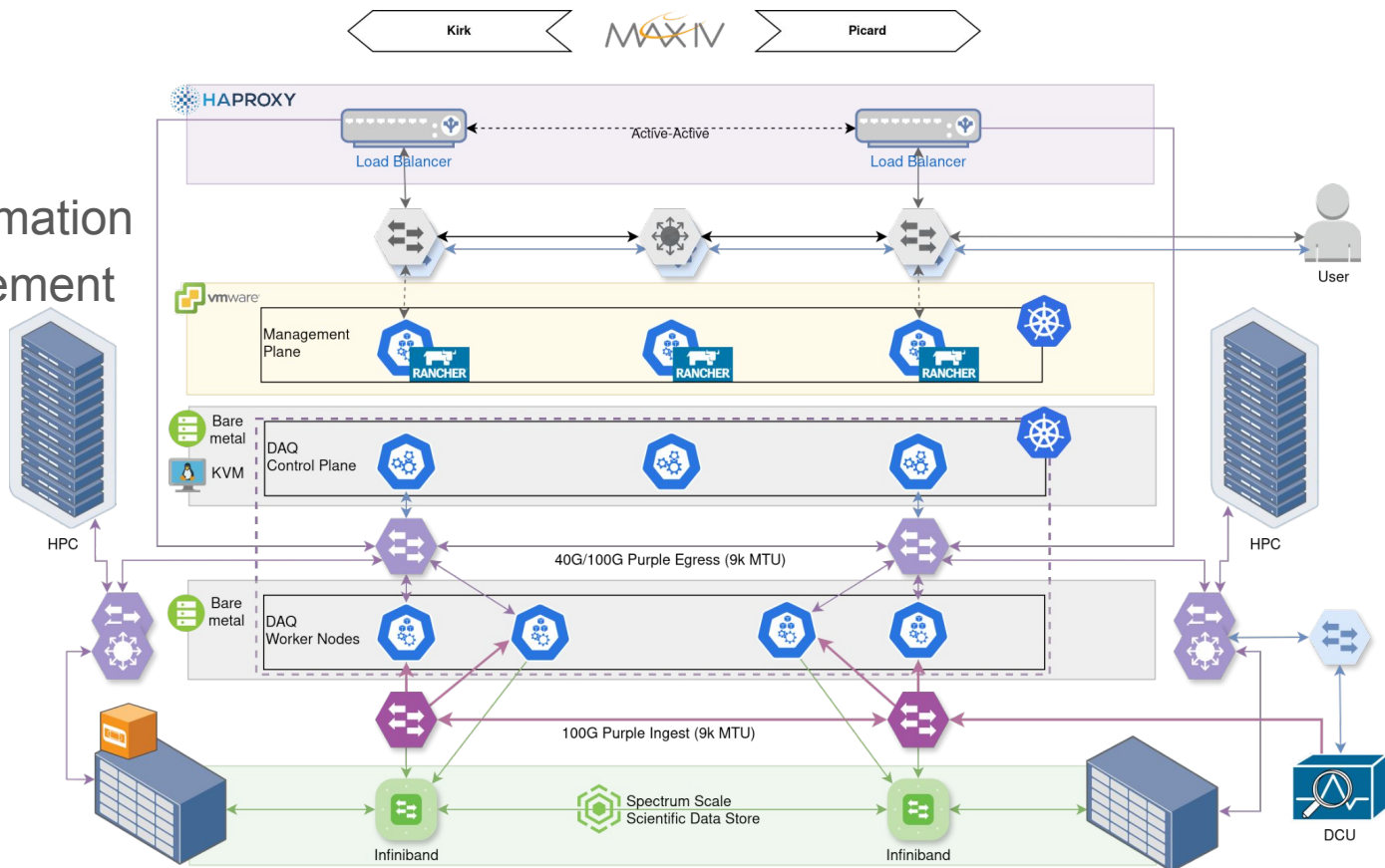
Lightweight, use-case specific setup, when we just need to “*automate deployment, scaling, and management*”





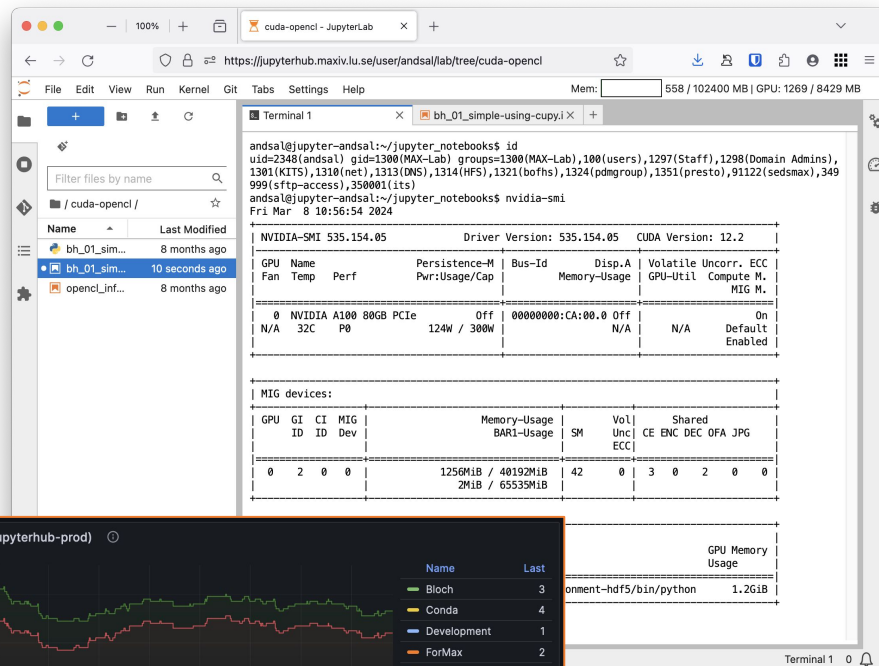
- Resource pool
- Deployment automation
- Pipelines management
- Scientific Data
- Security
- 100G networks

Artifact Hub kubepie



JupyterHub

- Resource pool
- Deployment automation
- GPU sharing
- Features build upon K8s



The screenshot shows a JupyterLab interface with a terminal window open. The terminal displays the output of the `id` and `nvidia-smi` commands. The `nvidia-smi` output shows the NVIDIA-SMI version, driver version, and CUDA version. Below this, a table lists GPU details including name, fan speed, temperature, performance, persistence mode, bus ID, display mode, volatile GPU memory, and error correction. A second table lists MIG devices with columns for GPU ID, GI ID, CI, MIG Dev, memory usage, and shared resources. The terminal also shows the output of the `comment-hdfs/bin/python` command, indicating a GPU memory usage of 1.2GiB.

GPU Name	Fan	Temp	Perf	Persistence-M Pwr:Usage/Cap	Bus-Id	Disp.A Memory-Usage	Volatile GPU-Util	Uncorr. Compute M. MIG N.	ECC Default
0 NVIDIA A100 80GB PCIe	N/A	32C	P0	Off 124W / 300W	00000000:CA:00.0	Off N/A	N/A	On	Default Enabled

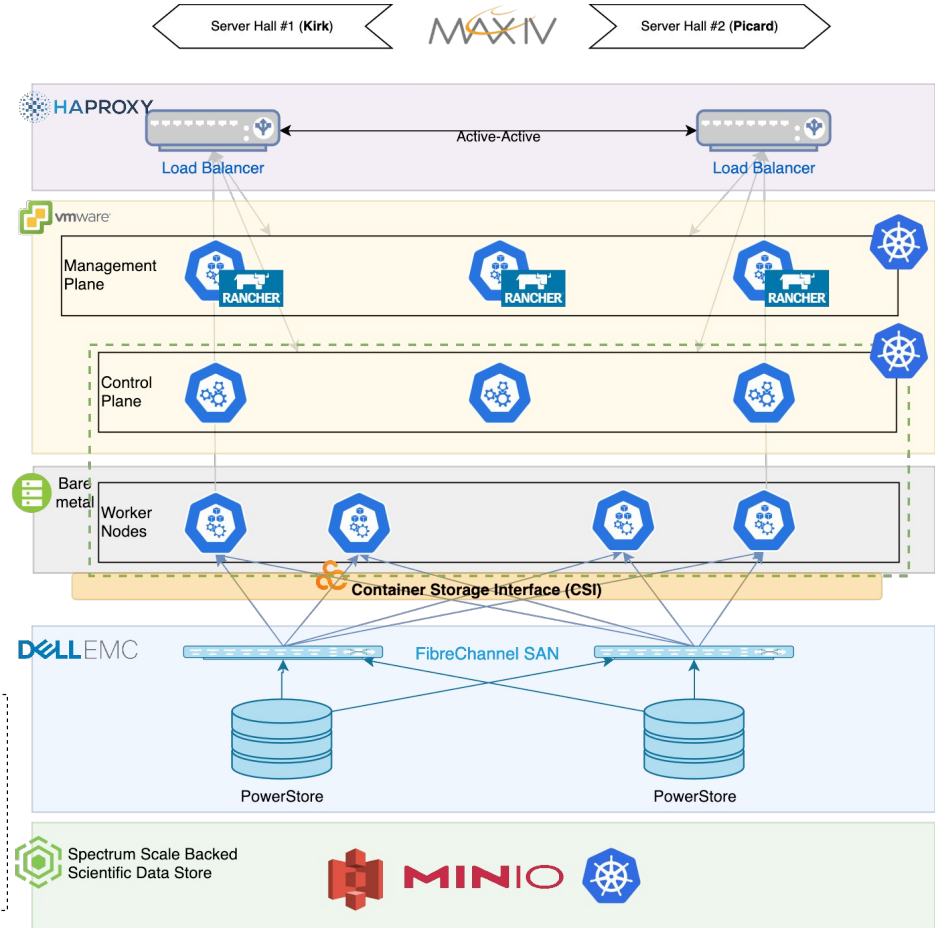
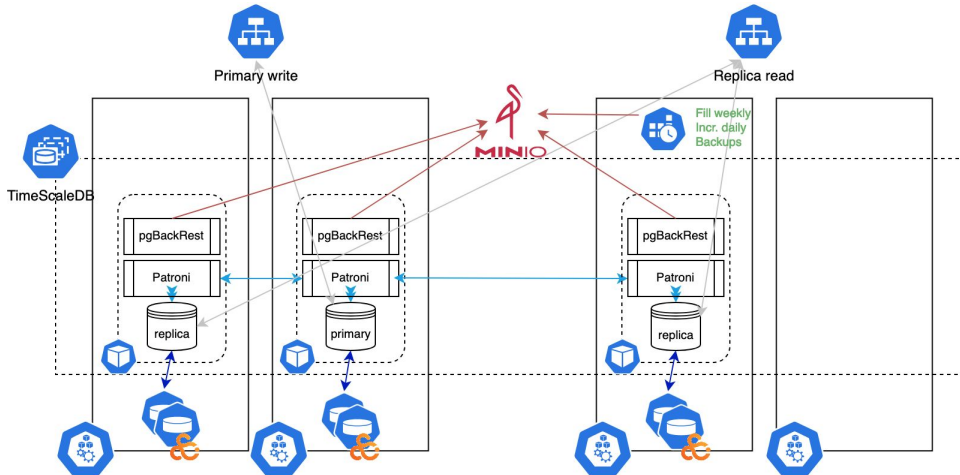
GPU ID	GI ID	CI	MIG Dev	Memory-Usage BAR1-Usage	Vol Unc ECC	Shared CE	BNC	DEC	OFA	JPG
0	2	0	0	1256MiB / 40192MiB 2MiB / 65535MiB	42	0	3	0	2	0

Name	Last
Bloch	3
Conda	4
Development	1
ForMax	2
HDFS	64
Matlab	1
MAXPEEM	1
Tomography	1
Total	73

Archiving



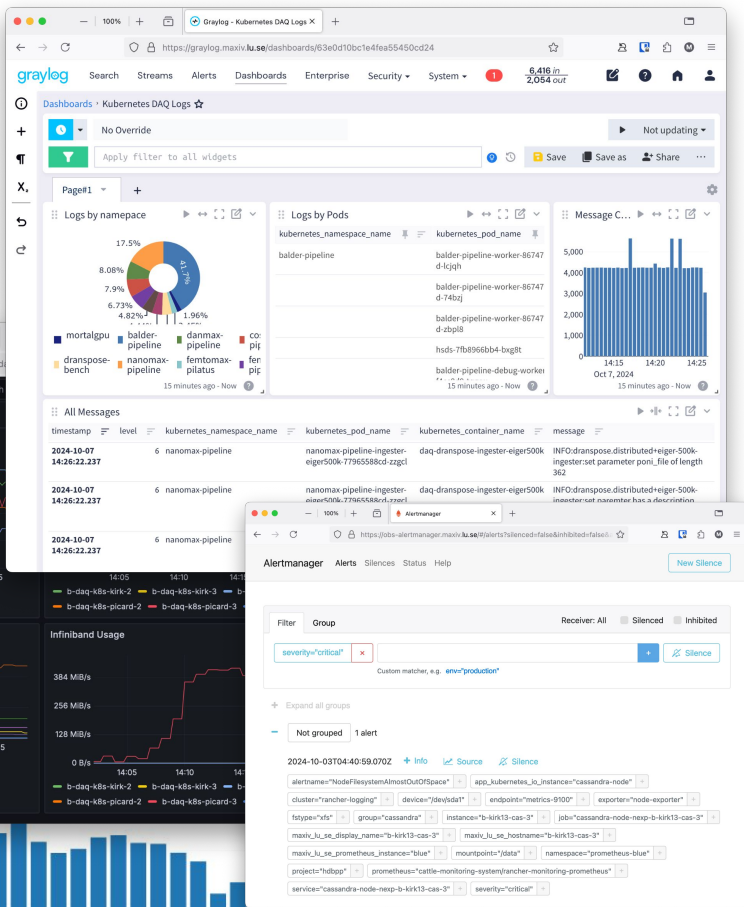
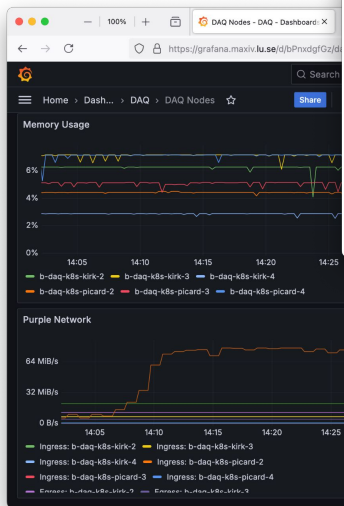
- **Maintenance** automation
 - auto failover and Day2 ops
- Infrastructure for **vendor supported Helm Chart**
 - minimize software expertise needed
- **Prod/dev** deployment

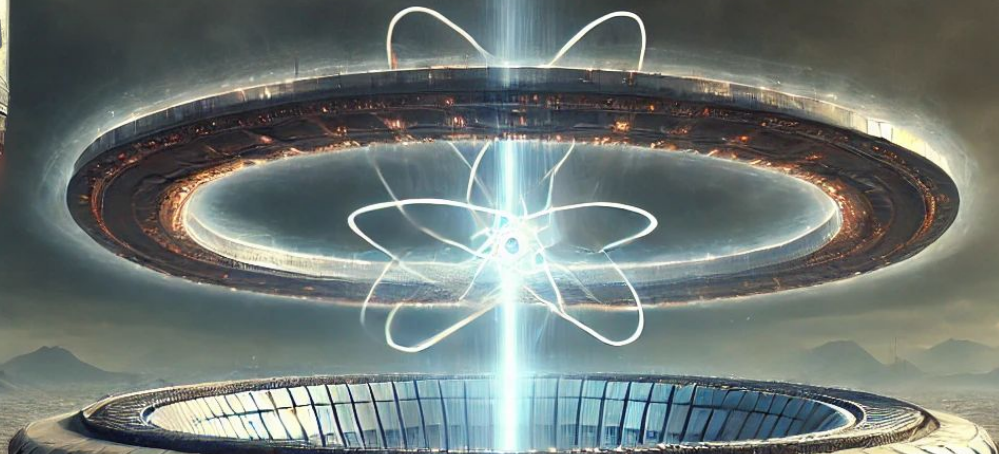


Observability

- Self contained cluster
- Deployment automation
 - Prometheus
 - Grafana
 - Alertmanager
 - Opensearch
 - Graylog
 - data sources specific workloads
- High volumes of logs and metrics (no license costs)

Outgoing traffic Last 90 days: 28.4TiB





Thank you for attention!

