Kubernetes meets Data Scientists

Prof. Dr. Peter Tröger

Beuth University of Applied Sciences, Berlin

GridKa Summer School

August 27th 2019

Beuth

- ~ 13.000 students
- One of the largest universities for applied science („Fachhochschule“) in Germany
- Mechanical engineering, electrical engineering, architecture, media computer science, data science, biotechnology, biophysics, event management …

Datexis

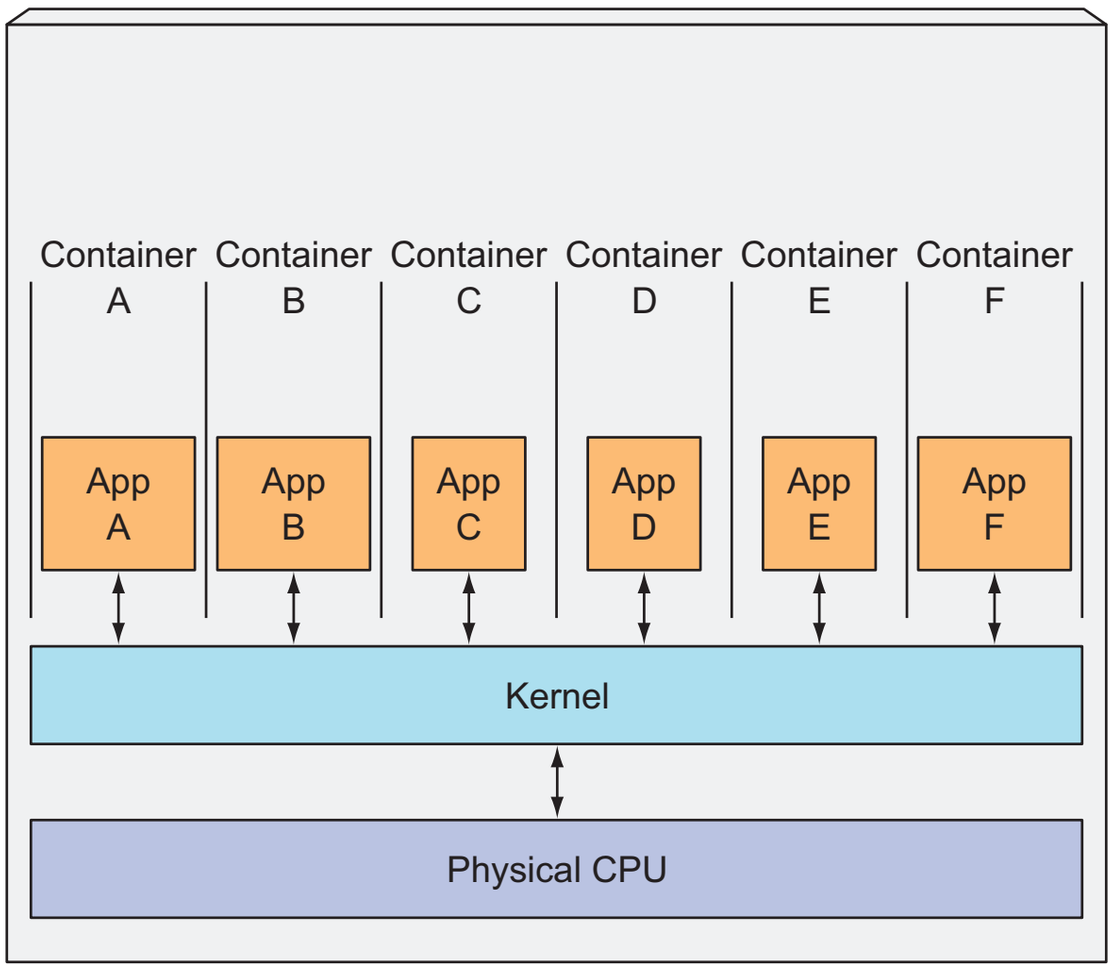
- Professors and PhDs in data science, databases, distributed systems, and mathematics
- Research on natural language processing and deep learning
  - Analysis of transformer representations
  - Topic segmentation and classification
  - Machine learning with medical text documents
- Fast-paced research field

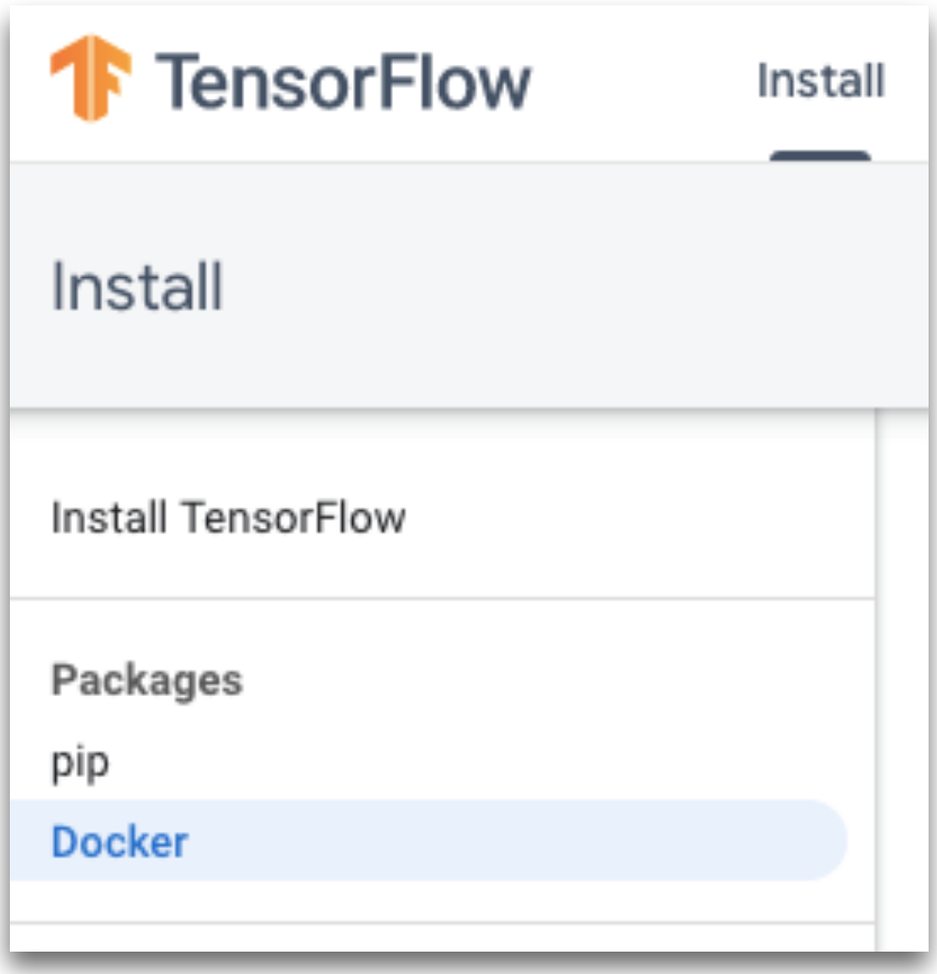
Datexis Infrastructure

- 30 machines, wild collection
  - Hadoop servers (24 disks, 64 cores, 512 MB RAM)
  - In-memory database hardware
  - Multi-GPU machines (P100, V100) with NVLink
- Workload
  - Hundreds of TB of text data to be preprocessed
  - Complex machine learning pipelines

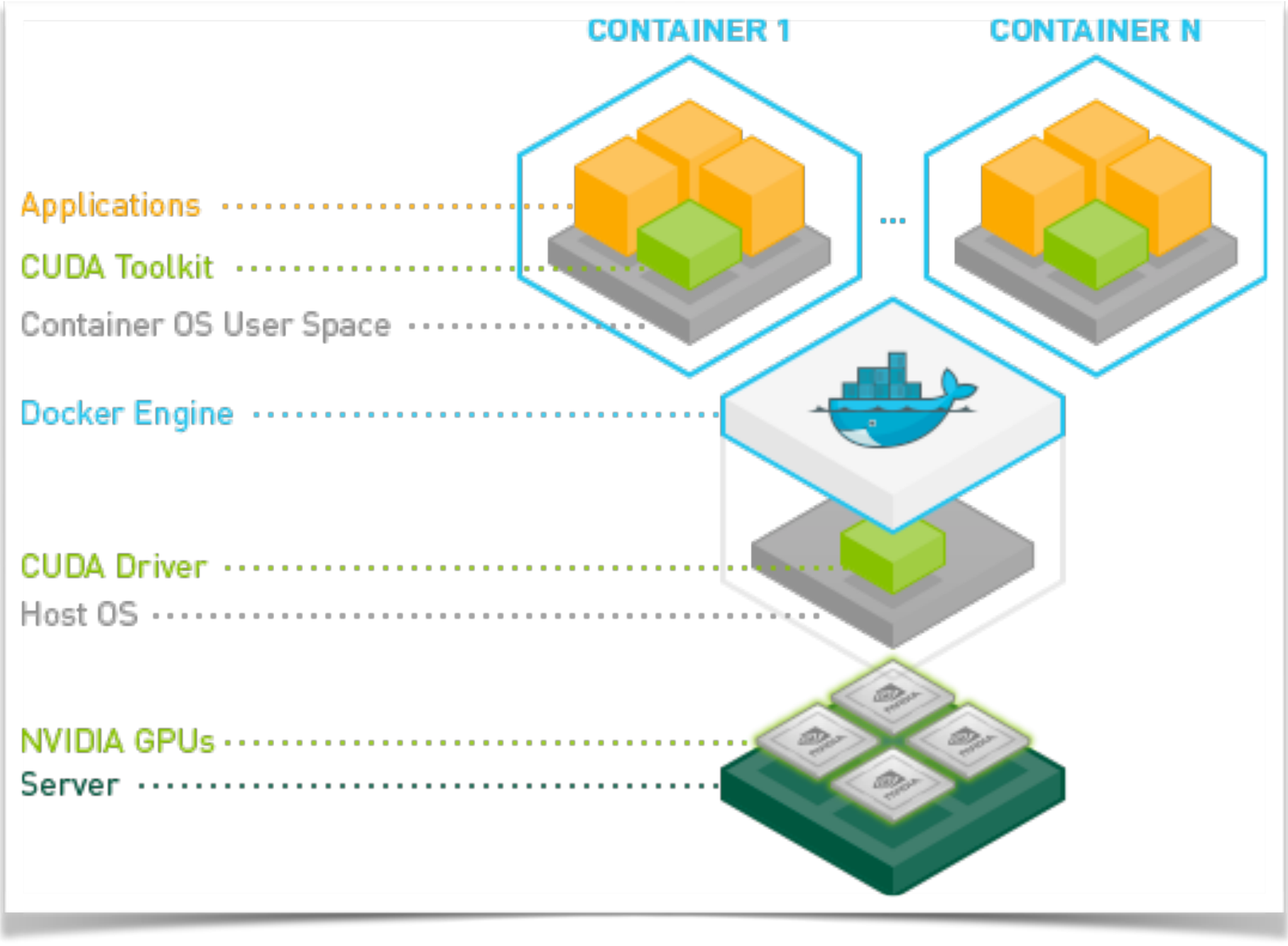
Datexis Software

- As usual: Python, R, TensorFlow, PyTorch, CUDA, …
- Docker everywhere in this field





Example: Nvidia Docker

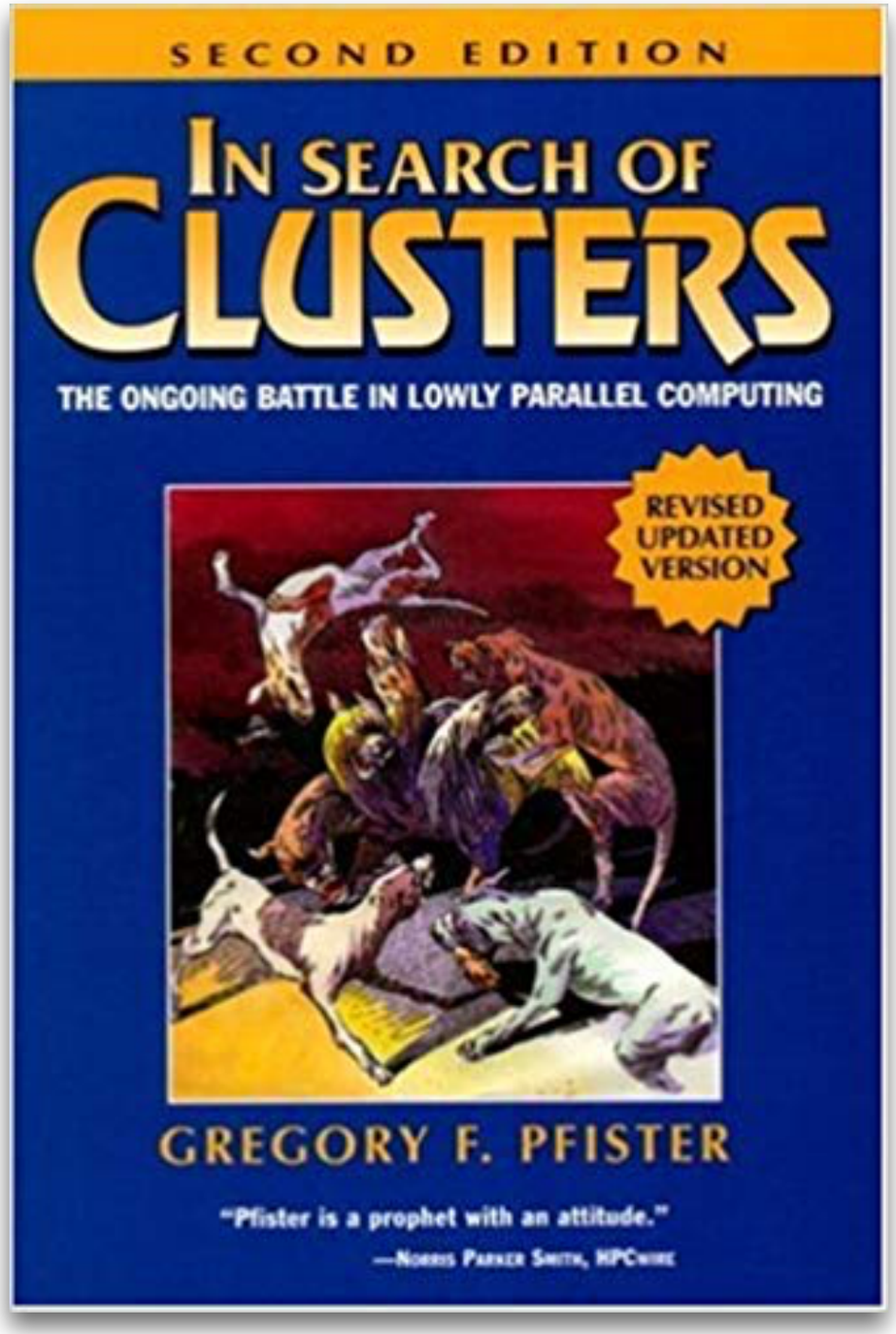


Starting Point

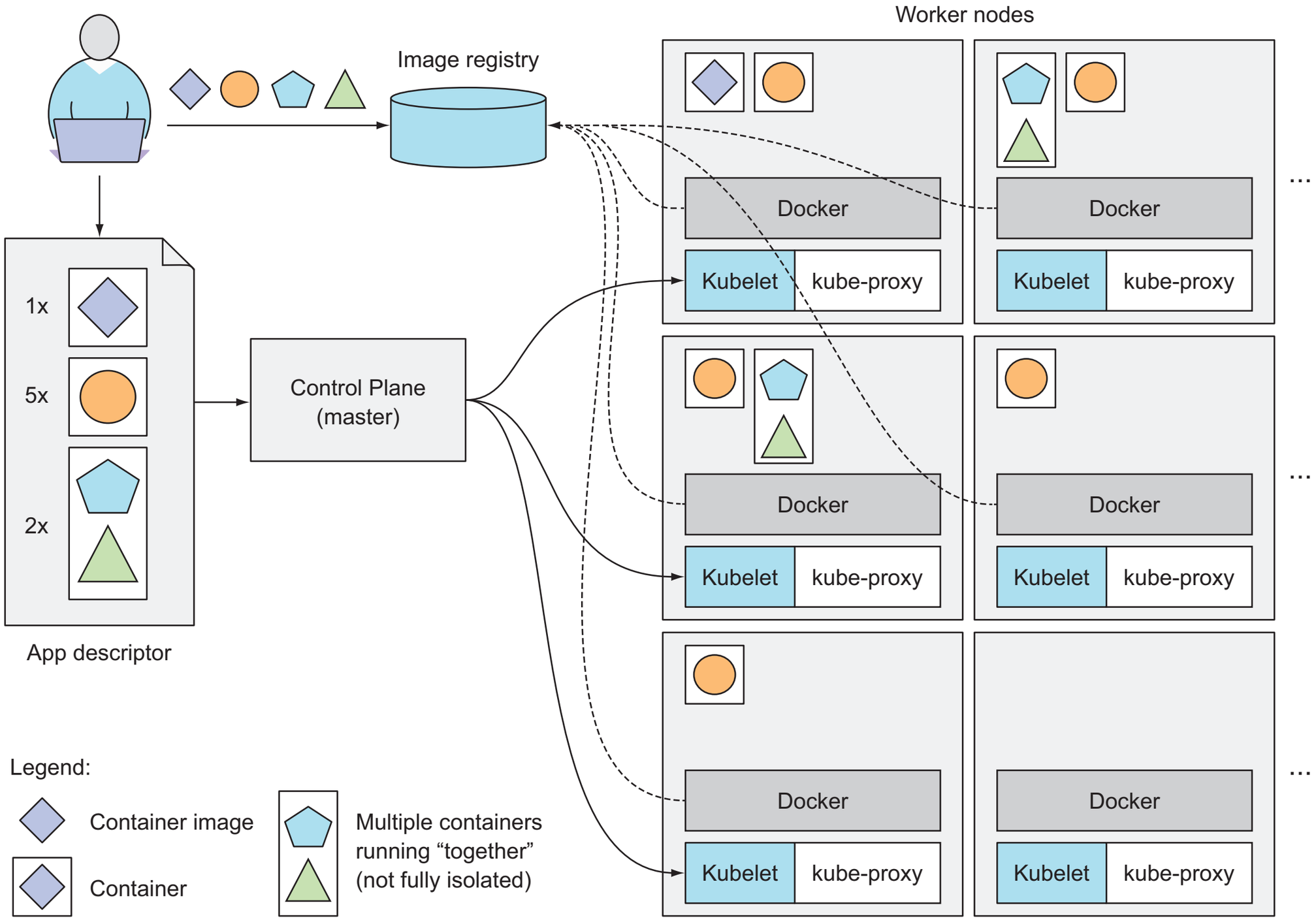
- 1 year ago:
  - SSH into physical servers, shared NFS
  - Pulling (tailored) Docker images for ML
- Calendar - based scheduling of resources
- Issues with library updates / dependencies
- Long-running computations, lack of resiliency

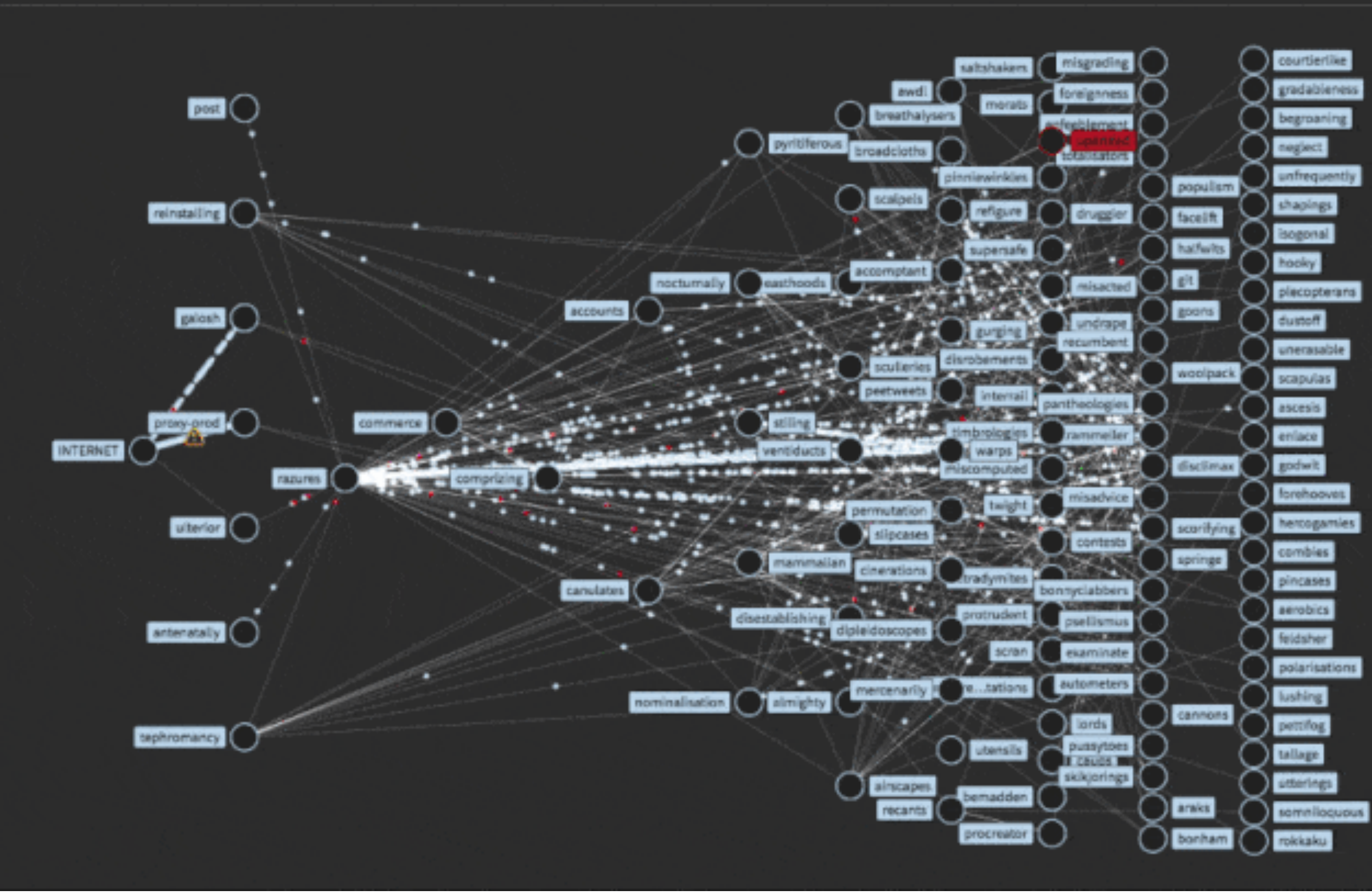
The Plan

- Introduce scheduler for containers on heterogeneous CPU / GPU resources
- Reproducible and isolated deployments
- More automation for batch processing
- Replace HDFS
- Real-world infrastructures in teaching
- Prepare for growth



Kubernetes


Worker nodes
Image registry
1x
5x
2x
App descriptor
Control Plane
(master)
Docker
Kubelet
kube-proxy
...
Legend:
Container image
Container
Multiple containers
running “together”
(not fully isolated)

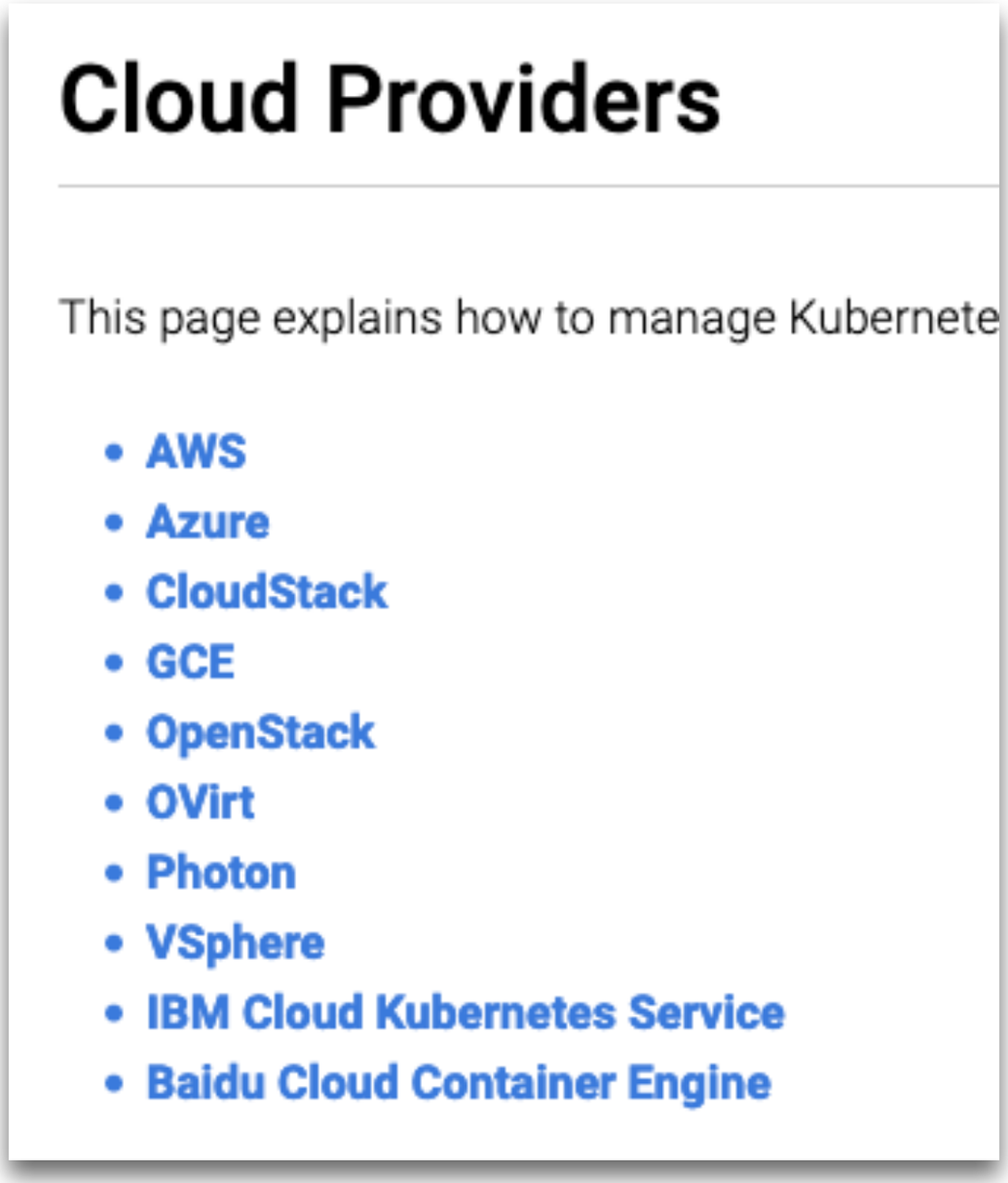


https://opensource.com/article/18/4/how-netflix-does-failovers-7-minutes-flat

How hard can it be …

- Many installation support packages and distributions available (kubespray, kubeadm, krib, KOPS, Canonical MaaS, …)

- But:
  - Main intention is a scale-out infrastructure for containers
  - GCE / AWS hosting as default
  - On premise often a special case



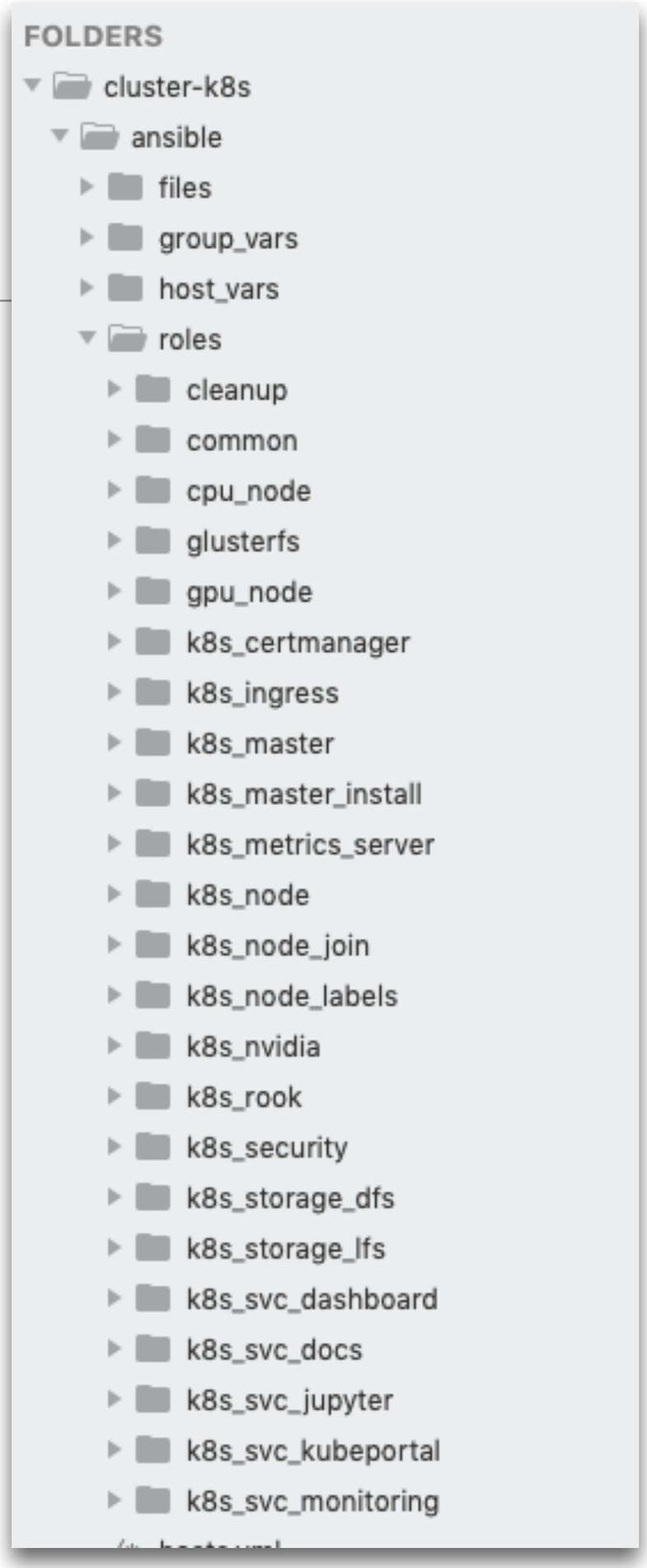
Cloud Providers

This page explains how to manage Kubernete

- AWS
- Azure
- CloudStack
- GCE
- OpenStack
- OVirt
- Photon
- VSphere
- IBM Cloud Kubernetes Service
- Baidu Cloud Container Engine

Basic installation

- Chose your favorite distribution
- Pick some container run-time
- Choose your level of installation ‚magic‘
- Follow the usual cluster node rules
  - NTP and DNS are critical
  - Automate everything
- Configure NVidia Docker for GPU nodes
- …



ptroeger — root@datexis-master2: ~ — ssh root@cluster.datexis.com — 80×24

```
root@datexis-master2:~# kubectl get nodes
NAME                STATUS                        ROLES    AGE     VERSION
cl-worker10         Ready                         <none>   3d22h   v1.12.7
cl-worker12         Ready                         <none>   69d     v1.12.7
cl-worker14         Ready                         <none>   165d    v1.12.7
cl-worker16         Ready                         <none>   215d    v1.12.7
cl-worker17         Ready                         <none>   203d    v1.12.7
cl-worker18         Ready                         <none>   145d    v1.12.7
cl-worker19         Ready                         <none>   115d    v1.12.7
cl-worker20         Ready                         <none>   21d     v1.12.7
cl-worker5          Ready                         <none>   166d    v1.12.7
cl-worker6          Ready                         <none>   10d     v1.12.7
cl-worker8          Ready                         <none>   165d    v1.12.7
datexis-master2     Ready,SchedulingDisabled      master   369d    v1.12.7
datexis-worker15    Ready,SchedulingDisabled      <none>   311d    v1.12.7
root@datexis-master2:~#
```

Users?

- Data science research works on tight schedules
- No time for long infrastructure learning curve
- Solution: Small manual with YML example snippets
- StackOverflow does the rest
- Command-Line tool `kubectl` and Kubernetes Dashboard GUI

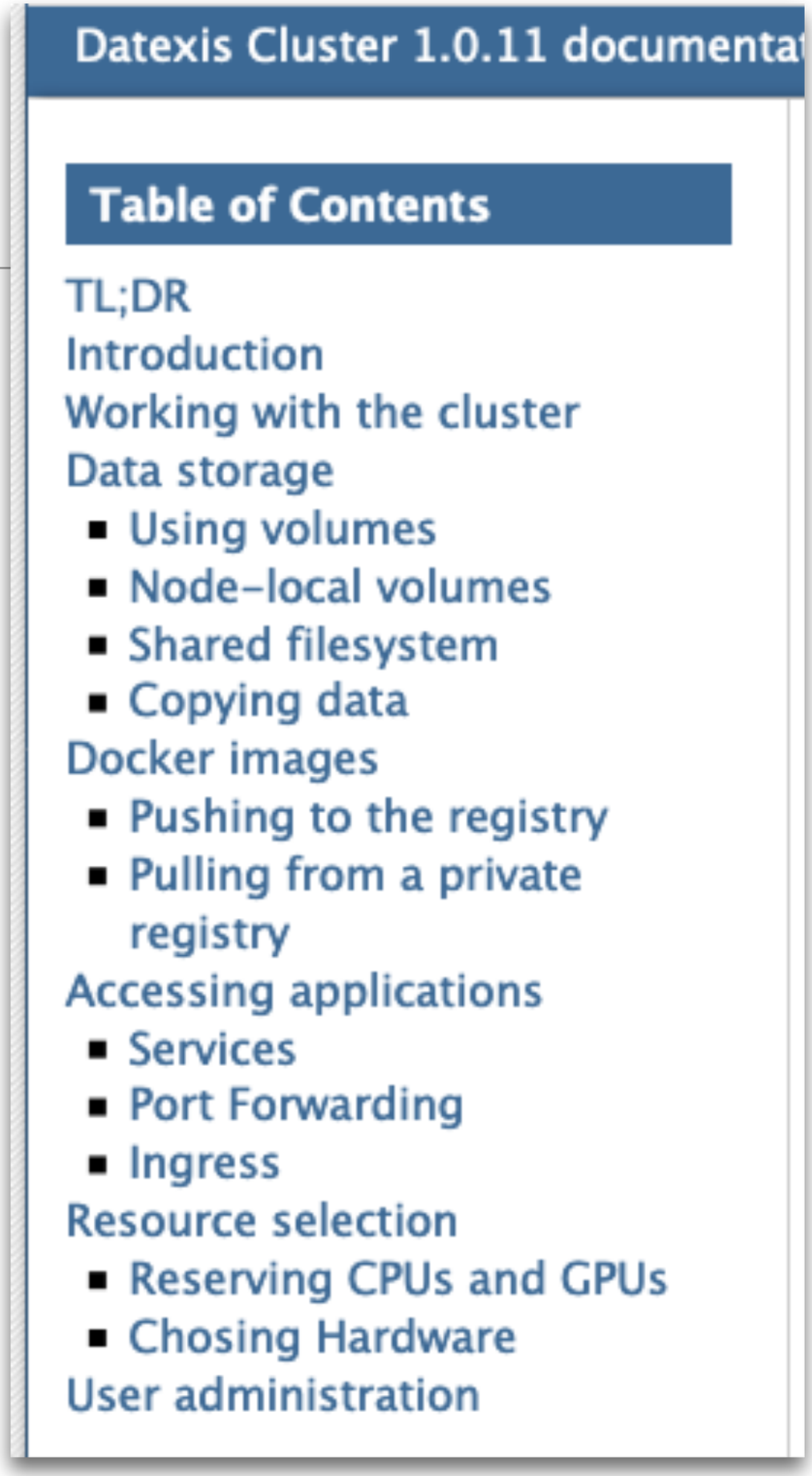
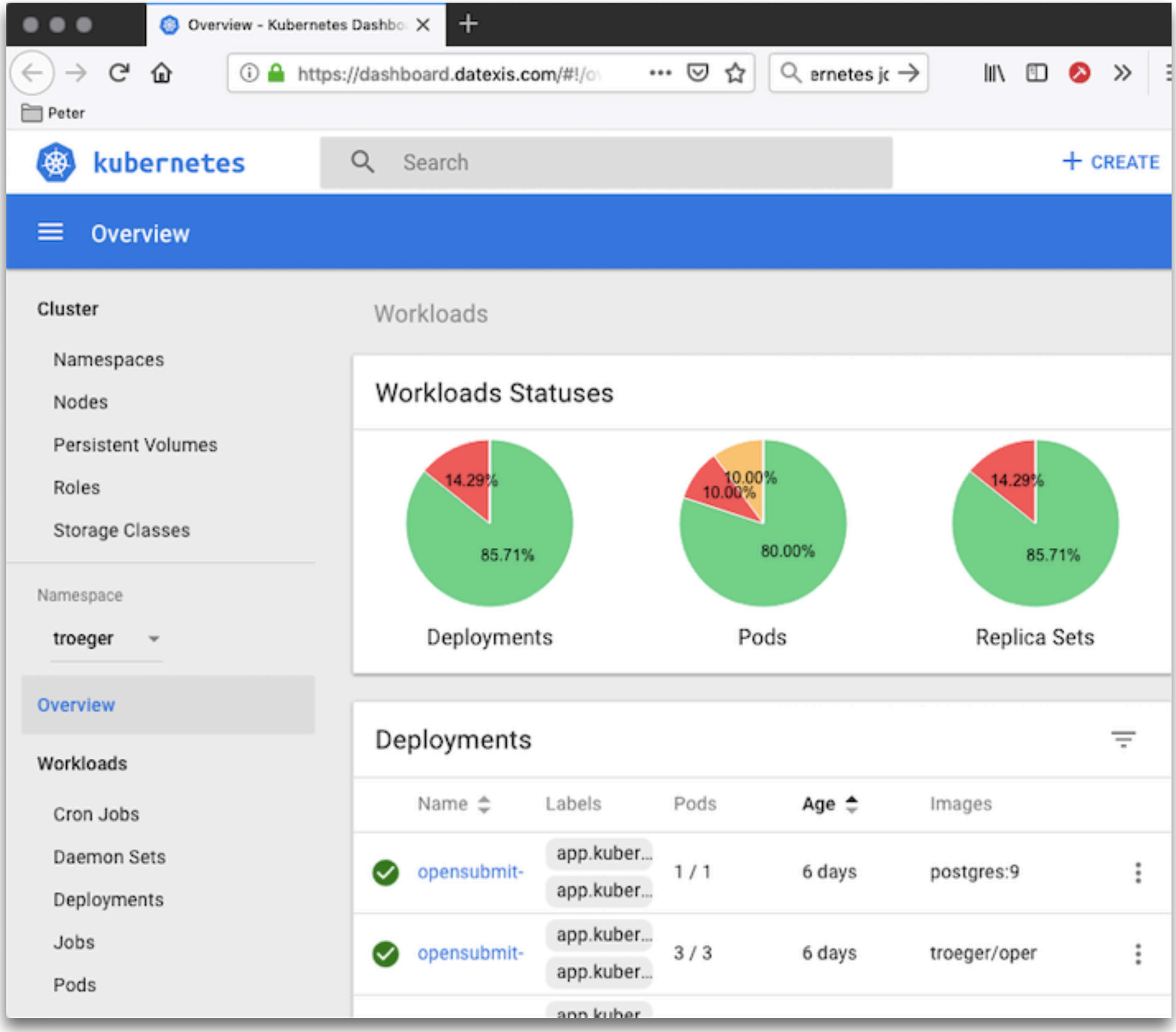

Datexis Cluster 1.0.11 documentat

Table of Contents

Example: GPU Allocation

Example:

```
apiVersion: apps/v1
kind: Deployment
...
spec:
  ...
    spec:
      containers:
        - name: cuda-vector-add
          image: "k8s.gcr.io/cuda-vector-add:v0.1"
          resources:
            limits:
              nvidia.com/gpu: 1 # requesting 1 GPU
      nodeSelector:
        gpu: k80
```


Overview - Kubernetes Dashbo
https://dashboard.datexis.com/#!/o
ernetes jc
Peter
kubernetes
Search
+ CREATE
Overview
Cluster
Namespaces
Nodes
Persistent Volumes
Roles
Storage Classes
Namespace
troeger
Overview
Workloads
Cron Jobs
Daemon Sets
Deployments
Jobs
Pods
Workloads
Workloads Statuses
14.29%
85.71%
Deployments
10.00%
10.00%
80.00%
Pods
14.29%
85.71%
Replica Sets
Deployments
Name
Labels
Pods
Age
Images
opensubmit-
app.kuber...
app.kuber...
1 / 1
6 days
postgres:9
opensubmit-
app.kuber...
app.kuber...
3 / 3
6 days
troeger/oper

Basic installation

- **Everything** in Kubernetes is extensible and pluggable
  - Container overlay network
  - Dynamic storage provisioning
  - Schedulers, schema for cluster resources
  - Quota and eviction handling
  - Security policy handing
- After initial cluster setup, the real fun begins …

Example: Container Networking

CNI	INSTALL	SECURITY	PERFS	RESOURCES
Calico	mtu	NetPol	95% bare metal	441MB RAM 5.9% CPU
Canal	mtu	NetPol	95% bare metal	443MB RAM 5.8% CPU
Cilium	no config	NetPol	94% bare metal	781MB RAM 11.1% CPU
Flannel	no config	none	95% bare metal	427MB RAM 5.7% CPU
Kube-router	mtu	ingress NetPol	96% bare metal	420MB RAM 5.7% CPU
WeaveNet	mtu	NetPol	89% bare metal	501MB RAM 8.9% CPU
Cilium encrypted	crypt + mtu	crypt + NetPol	7% bare metal	765MB RAM 12.5% CPU
WeaveNet encrypted	crypt + mtu	crypt + NetPol	10% bare metal	495MB RAM 9.2% CPU

https://itnext.io/benchmark-results-of-kubernetes-network-plugins-cni-over-10gbit-s-network-updated-april-2019-4a9886efe9c4

Example: CPU Partitioning

Tuesday, July 24, 2018

Feature Highlight: CPU Manager

Authors: Balaji Subramaniam (Intel), Connor Doyle (Intel)

This blog post describes the CPU Manager, a beta feature in Kubernetes. The CPU manager feature enables better placement of workloads in the Kubelet, the Kubernetes node agent, by allocating exclusive CPUs to certain pod containers.

```
apiVersion: v1
kind: Pod
metadata:
  name: exclusive-2
spec:
  containers:
  - image: quay.io/connordoyle/cpuset-visualizer
    name: exclusive-2
    resources:
      # Pod is in the Guaranteed QoS class because requests == limits
      requests:
        # CPU request is an integer
        cpu: 2
        memory: "256M"
      limits:
        cpu: 2
        memory: "256M"
```

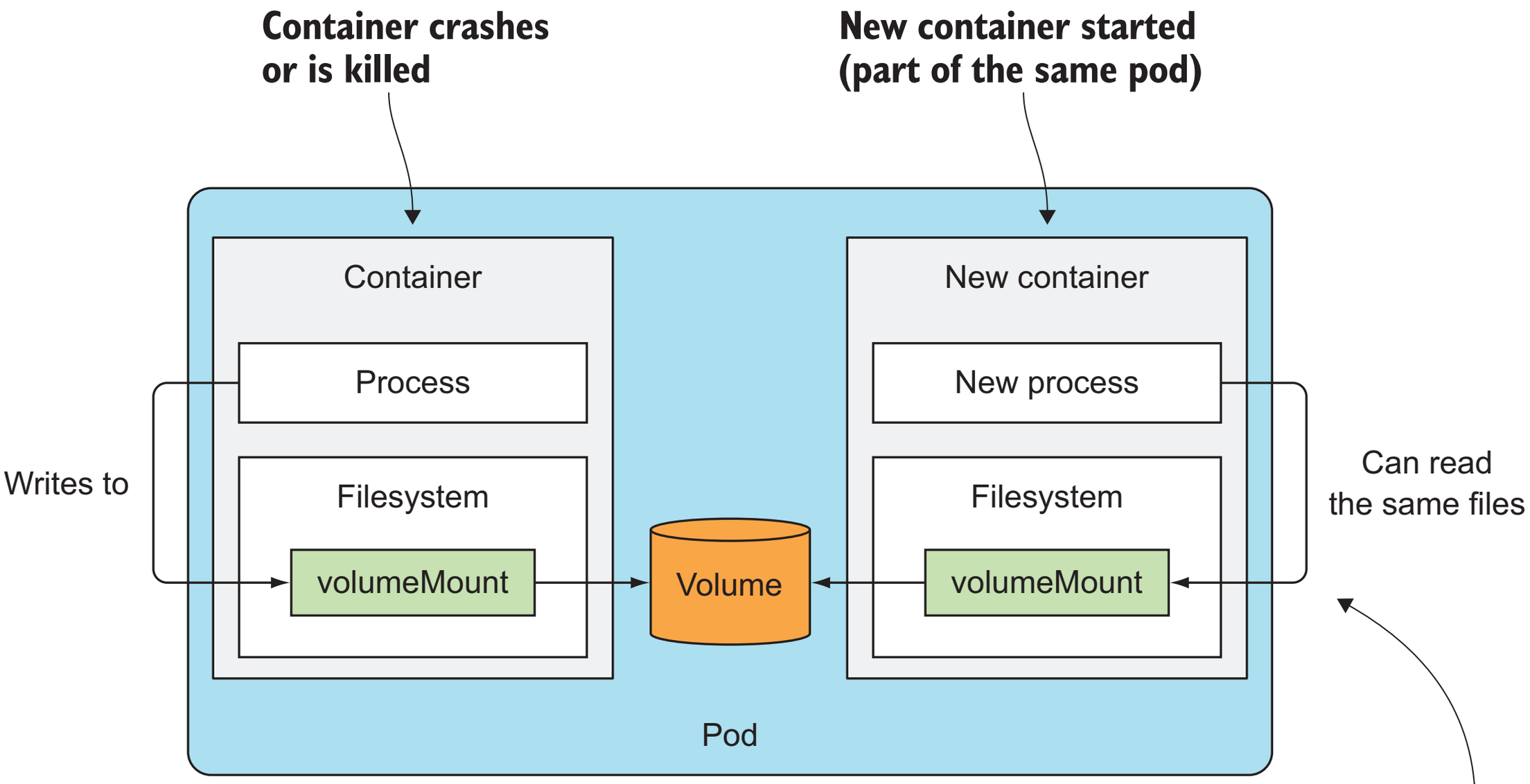
First user experiences

- Containers (alone) are supposed to be stateless
- Killing containers is a normal and regular activity
- Initially very irritating
  - Things no longer run on „your“ host
  - Checkpointing mentality
- ML Docker images are not really ready for that
- Quick demand for custom images

First problems

- `/var/lib/docker` for the overlay file system
- `/var/lib/docker` for images being pulled
- Swap is disabled on Kubernetes nodes
- Users where easily able to fill the disk
  - Data analysis tasks in containers filling `/tmp`
  - Private Docker images for large data transfer

Volumes


Container crashes or is killed
New container started (part of the same pod)
Container
Process
Filesystem
volumeMount
New container
New process
Filesystem
volumeMount
Volume
Pod
Writes to
Can read the same files
New process can use data preserved in the volume

Dynamic Volume Provisioning

- Users describe demand for storage (*persistent volume claim*)

- Kubernetes forwards request to provisioner
  - Some storage technology + API + Kubernetes integration
  - Examples: GCE disk, AWS EBS device, Azure disk, Fibre Channel, NFS mount, iSCSI mount, VSphere volume, Portworx, ScaleIO, …

- Provisioner creates a mountable *persistent volume*

Example: Datexis storage

- 120 disks over 7 nodes, 200TB raw storage
- GlusterFS
  - First attempt, looked manageable
  - Broken support for relational databases in volumes
- Current attempt: Rook
  - Ceph deployment as containers
  - Scaling becomes easier, network filesystem included

More problems

- Special provisioner for *local persistent volumes*
  - Real physical disk for caching latency-critical data
  - One user stored 12 million input files on the disk
    - *Locate daemon* was still running on the physical node
- Everything inside Kubernetes uses TLS
  - Certificate roll-over does not happen automatically
- Killing the cluster with one line:

```
kubectl -n foo delete pods, ns -all
```

Different mindset

federation: kubectl --cascade should be false by default for federation #38897

Closed **nikhiljindal** opened this issue on 16 Dec 2016 · 34 comments

nikhiljindal commented on 16 Dec 2016 — Member

Ref #33612.

Cascading deletion is true by default for kubectl.

For federation, we want it to be false since deleting a federation resource with cascade=true will delete the resource from all clusters as well which can lead to a global outage. So we want --cascade to be false by default but users can pass --cascade=true if they do want cascading deletion.

Assignees

irfanu

nikhil

Labels

area/fed

area/kub

priority/

Comment by nikhiljindal

Monday Mar 20, 2017 at 05:48 GMT

As per the discussion above, defaults for cascading deletion in federation will be same as in kubernetes i.e `--cascade=true` by default for `kubectl delete`.
Closing this issue.

The missing landing page

- Kubernetes in-built authentication is for software,
  not for humans (bearer tokens, X.509 certificates, OIDC)

- On-boarding of users is your problem

- Project „Dex“: Extension of auth options, but still no frontend

- Project „K8S Dashboard“: No single sign-on

- Nice user frontends seem to be only available as part of
  commercial products

- So we built something for ourselves …

cluster.datexis.com/

https://cluster.datexis.com

Suchen

Peter

Datexis Cluster

User name

Password

Sign In

Status: *beuth-hochschule.de* login is **available**

Sign in with Google

cluster.datexis.com/welcome/

https://cluster.datexis.com/welcome/

Suchen

Peter

Datexis Cluster

Welcome

Config

Admin

Logout

Hello Peter!

Full name: Peter Tröger

User name: ptroeger

Kubernetes Service Account: troeger:default

Datexis Kubernetes Dashboard

Start

Datexis Monitoring

Start

Datexis Docker Registry

Start

Datexis Cluster Manual

Start

cluster.datexis.com/config/

https://cluster.datexis.com/

Suchen

Peter

Datexis Cluster

- Welcome
- Config
- Admin
- Logout

config (Download)

```
apiVersion: v1
clusters:
- cluster:
    insecure-skip-tls-verify: true
    server: https://datexis-master2.beuth
  name: datexis_cluster
contexts:
- context:
    cluster: datexis_cluster
    namespace: troeger
    user: ptroeger
  name: ptroeger
current-context: ptroeger
kind: Config
preferences: {}
users:
- name: ptroeger
  user:
    token: eyJhbGciOiJSUzI1NiIsImtpZCI6Ii
```

This configuration file is needed for Kubernetes client tools on your computer.
It contains your personal access token.

Using kubectl

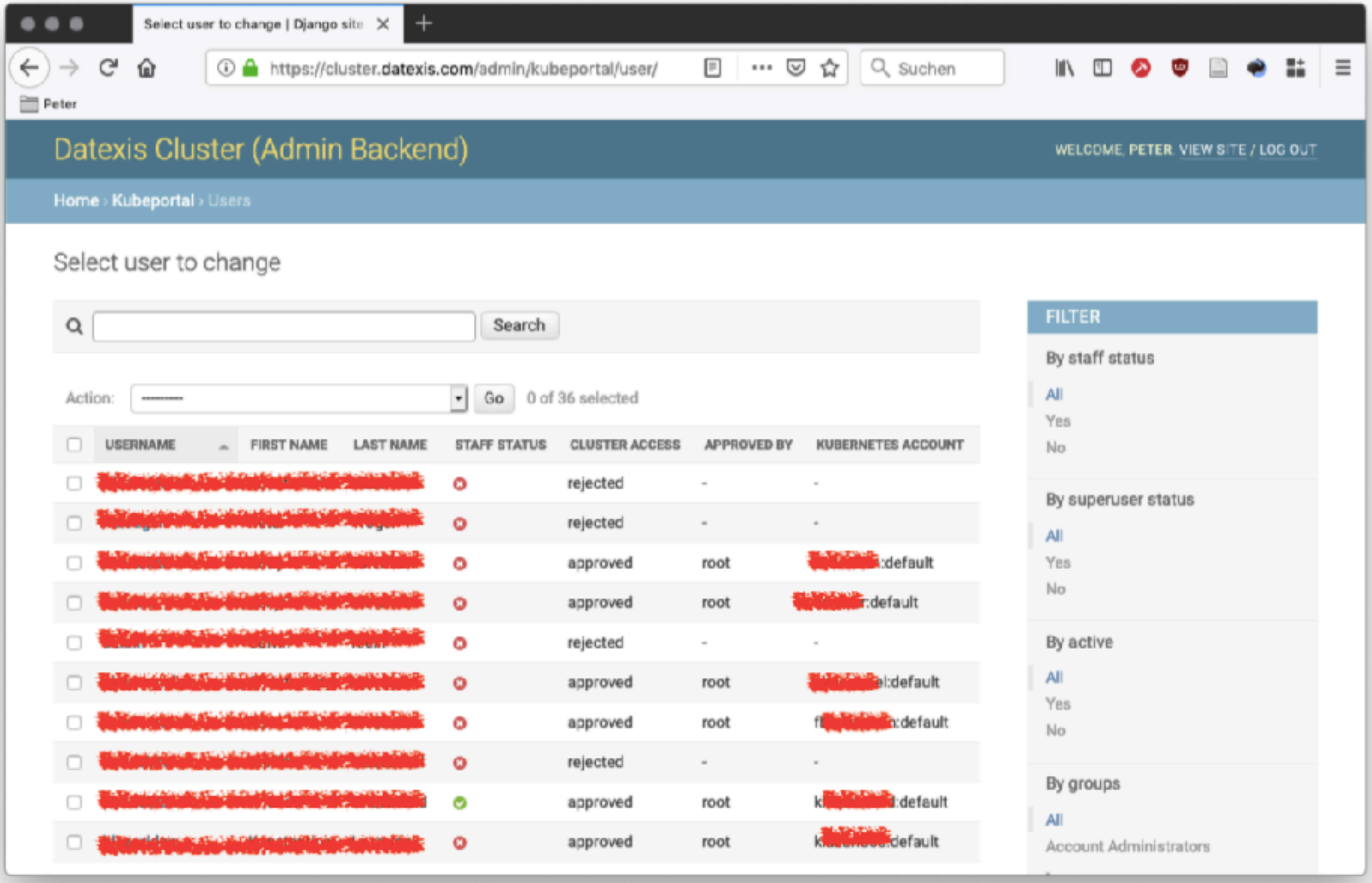
Windows

- Install kubectl for Windows
- Navigate to your home directory: `cd %USERPROFILE%`
- Create the `.kube` directory: `mkdir .kube`
- Store the config file as `.kube/config`

MacOS X / Linux / Unix

- Install kubectl with your package manager
- Navigate to your home directory: `cd ~`
- Create the `.kube` directory: `mkdir .kube`
- Store the config file as `.kube/config`

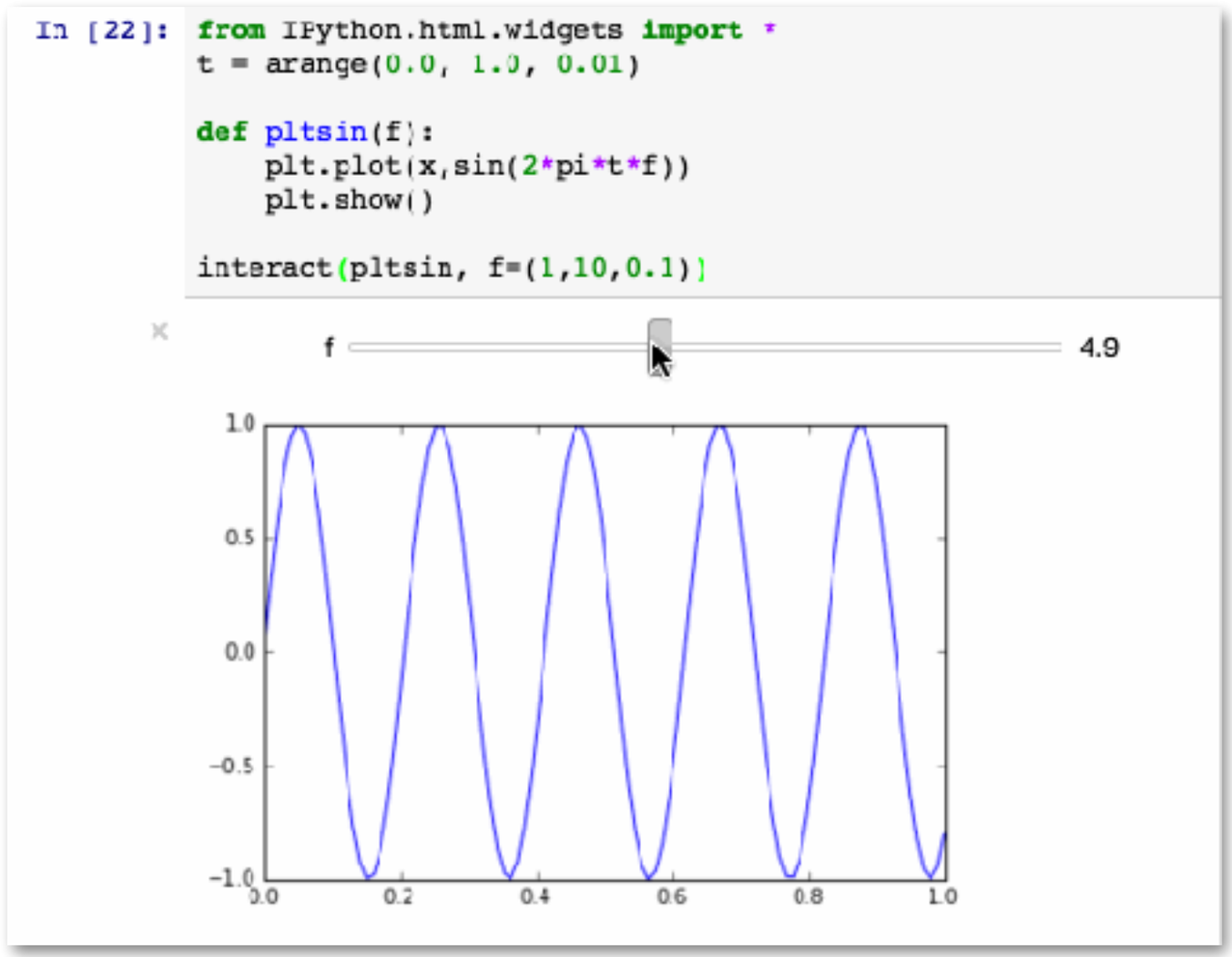
You can test your installation by calling `kubectl cluster-info`.



https://github.com/troeger/kubeportal

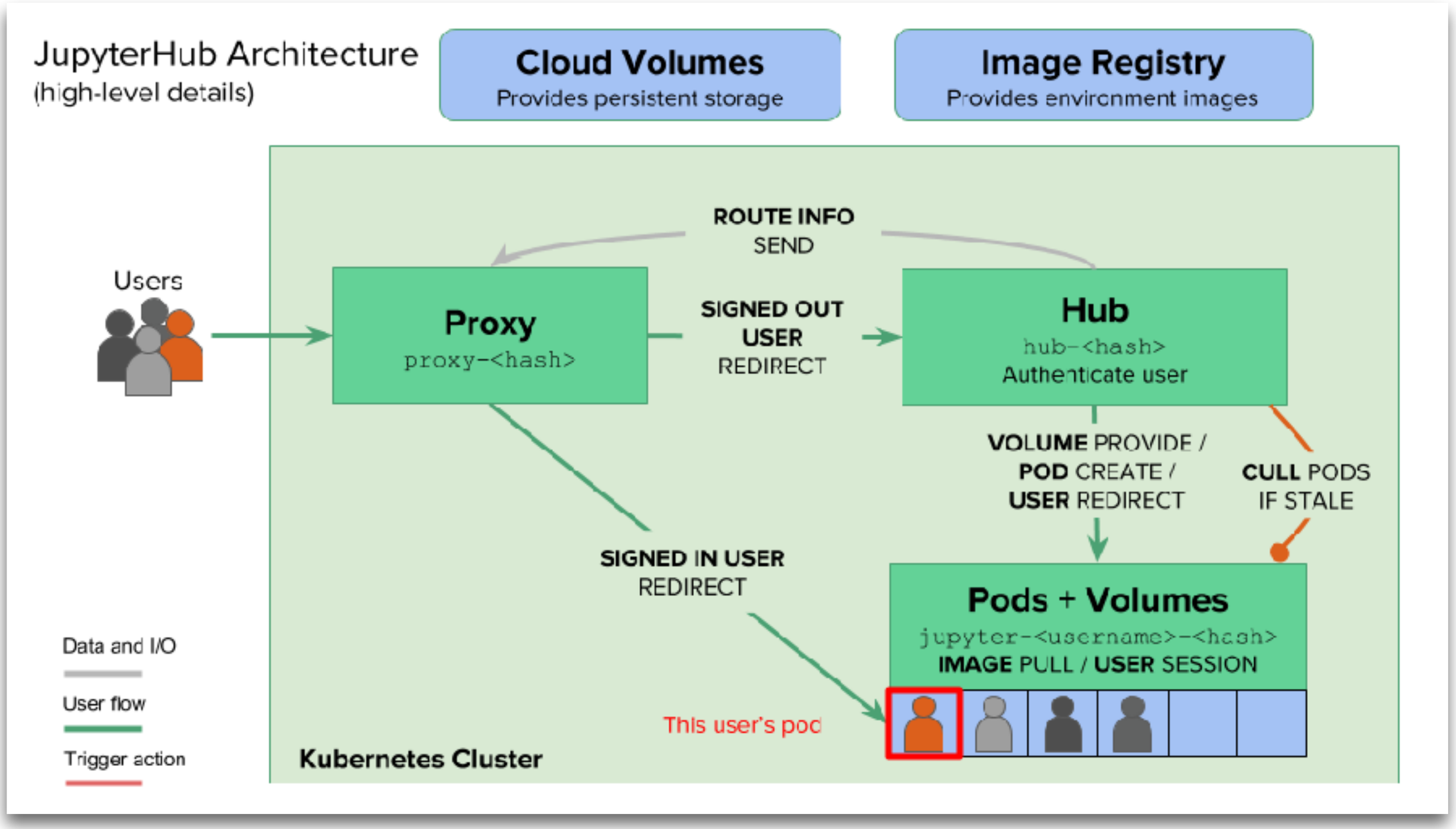
Status

- Shared GPU usage with Kubernetes works smoothly
- Several benefits (batch processing, improved storage, single sign-on) already taken for granted
- Still fighting with low-level management of resources (images, YML, mounts)
- What they really want is Jupyter notebooks …



https://blog.dominodatalab.com/interactive-dashboards-in-jupyter/

JupyterHub



https://zero-to-jupyterhub.readthedocs.io/en/latest/architecture.html

Conclusion

- Kubernetes as container orchestration engine
  - Fits perfectly to Docker-based software packaging
  - Endless flexibility, hard choices for admins
  - Vibrant community
- Great possibilities for knowledge transfer from the grid world (portal technology, federated user authentication, scalable storage, job pipelines, resource partitioning, resource accounting, security, …)
- The story continues.