From terminal to platform

Challenges on building up a Cloud Science Platform in an HPC world at DZA

Marc Drobek, DZA (German centre for astrophysics), state: 9th Dec 2025

Neutron star merger, AEI Golm

Overview

In this presentation we want to show the technical challenges we are facing when building up a modern science platform for astrophysical projects and potential solutions from the perspective of computer science.
We will use the national German deSRC node from the SRCNet project as an example to illustrate hard challenges.

Agenda:

- SRCNet Intro and Status Quo
- HPC vs Cloud
- Options / solutions
- Summary

DZA – German centre for Astrophysics

- Currently being founded as a country-wide institute to support the German astrophysics research and collaborate with already existing established institutes

- Enable German participation in large astronomical research projects and facilities

- Have a focus on Radio-, Gravitational-Wave and Time-astronomy

- Consists of 3 closely interlinked science pillars:
  - Astro physics
  - Data science
  - Technology

- Mission to catalyze scientific, economical and sociological developments in the former coal mining area of Lusatia
  - “From coal to silicon”

DZA – Some example projects

- ESA Gaia DR4
  - ESA's billion star surveyor
  - https://www.esa.int/Science_Exploration/Space_Science/Gaia
  - DZA will aim to host DR4 as a partner institute
  - Hosting Type:
    - Catalogue data with various search types
    - Think: Storage + Web/TAP/IVOA access

- SKAO – SRCNet
  - https://www.skao.int/en
  - DZA will aim to support setup/installation of the official “deSRC” node
  - Hosting Type:
    - “Science Platform”
    - Think: Storage + Interactive/Batch Compute + Web/TAP/IVOA access

In the talk, we will focus on the technical/engineering challenges for such a “Science Platform” on the example of SRCNet

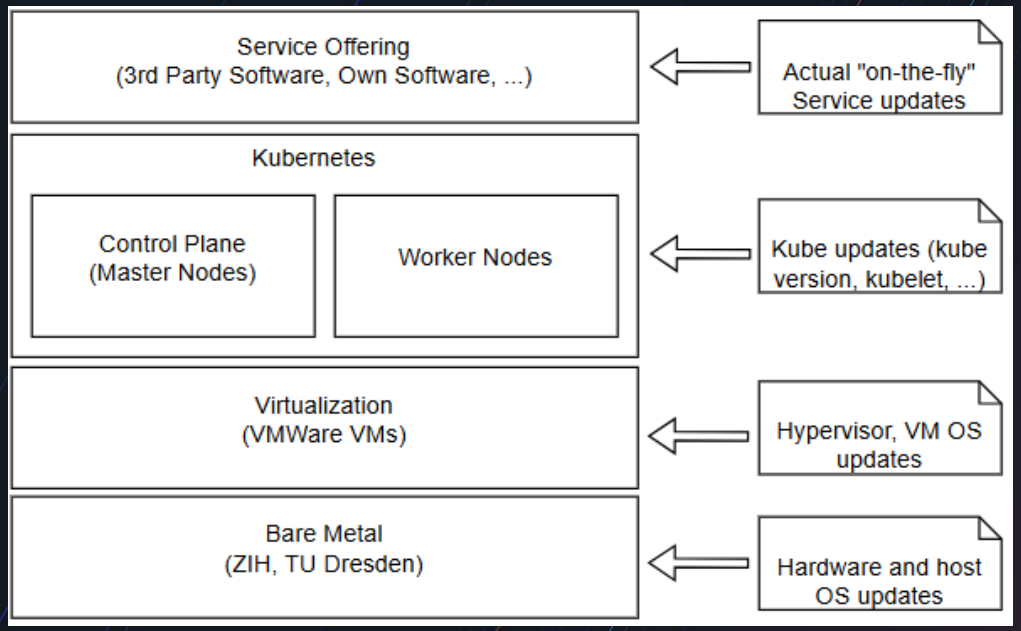
SRCNet Intro and Status Quo

SRCNet basic overview

- Architecture:
  - Distributed node architecture to spread out redundant data storage and compute
  - Follow the example of CERN

- Software Stack:
  - Modern service like development program to follow the industry state of the art
    - Kubernetes (there is no official req. for Kube, but the developed artifacts (images/Helm charts/etc.) imply that req.

- Hardware Stack
  - Heterogenous IT infrastructure at each contributing country/institute

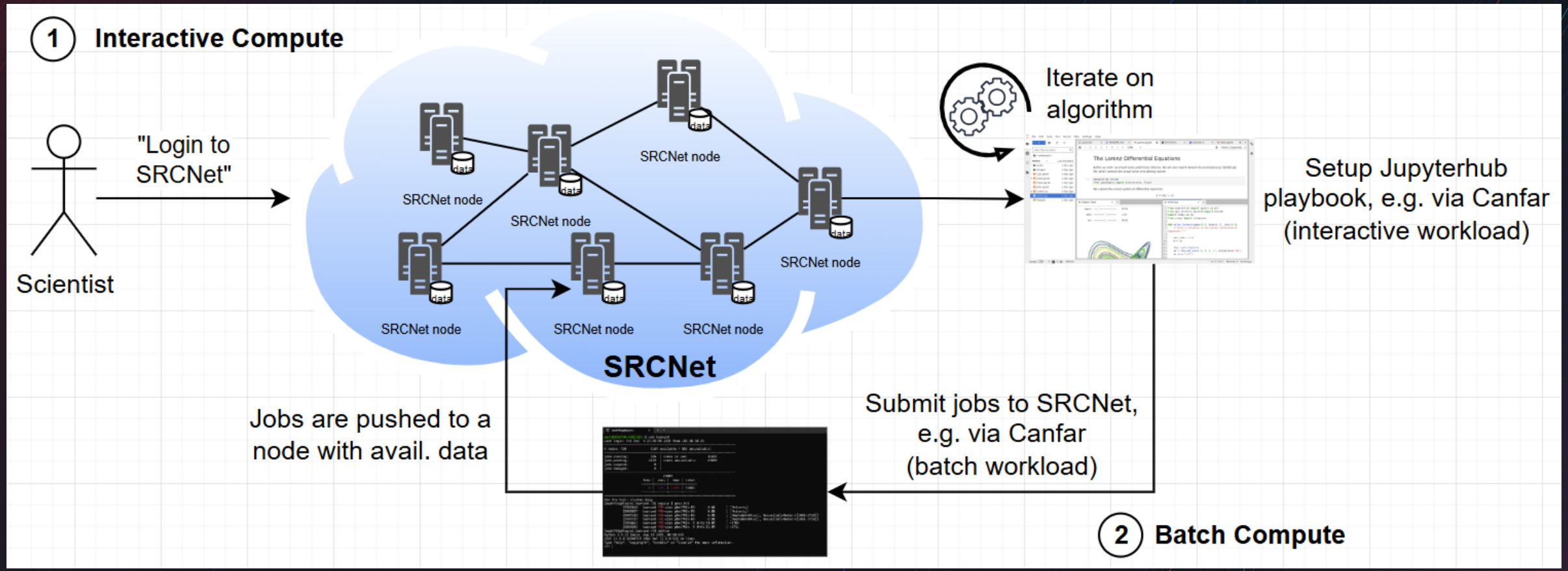


Opinion:

From the perspective of an Industry/Service background -> this is the right way to go

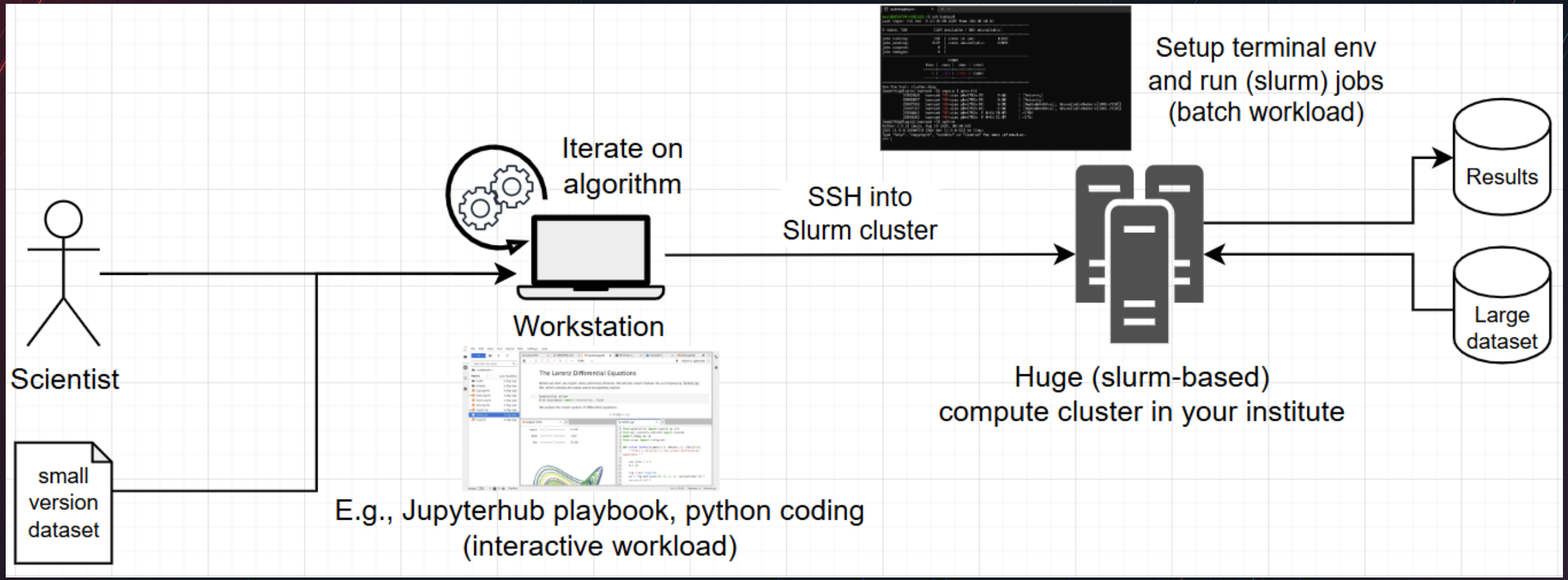
Assumed workflow on a science platform (HTTPS-based)

Or any variation of that



Status Quo: Workflow of a traditional science use-case (SSH-based)

Or any variation of that

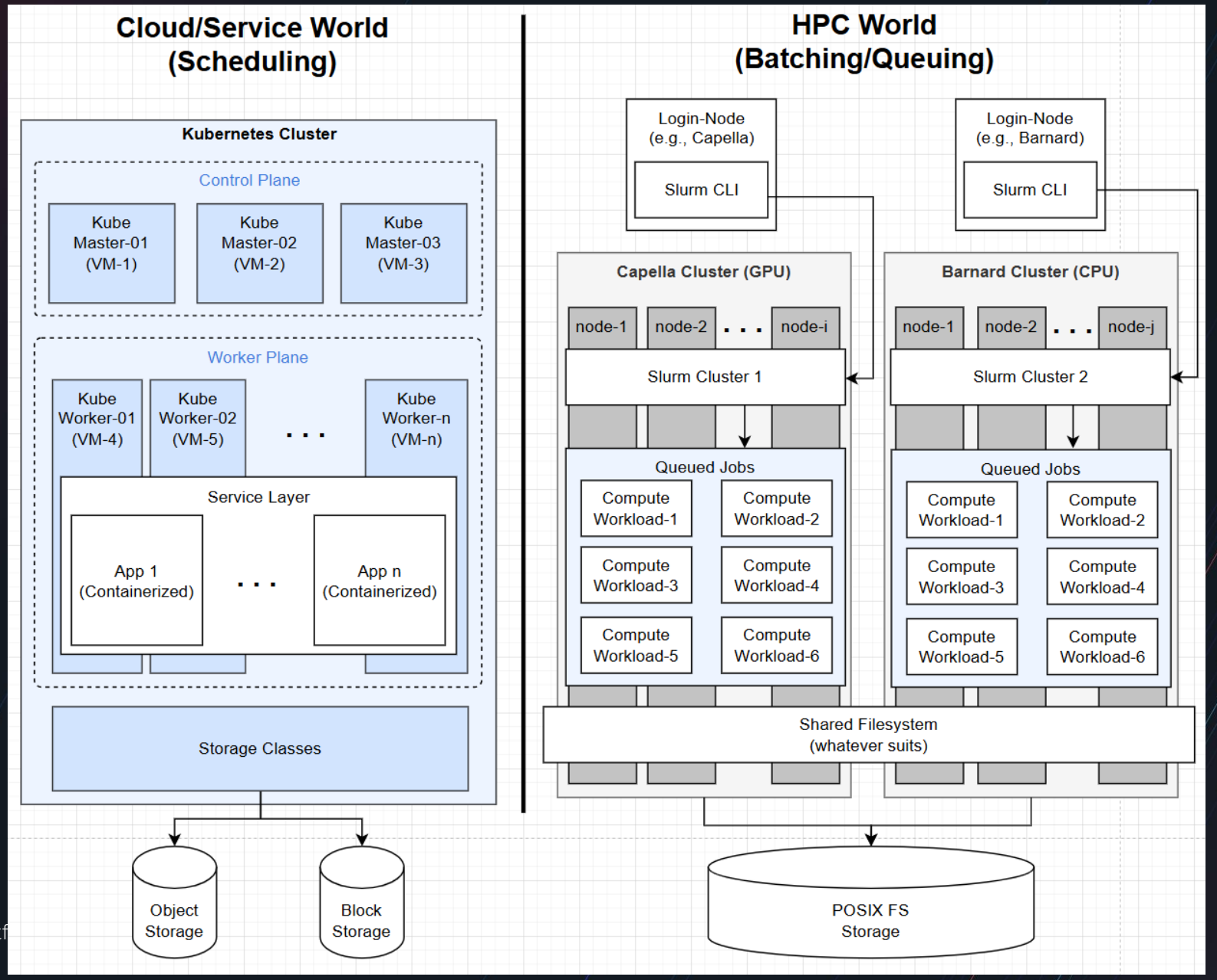


Opinion:

From an academic/batch-job perspective -> this has been an easy and reliable traditional approach

HPC vs Cloud

HPC vs Cloud - Comparison



HPC vs Cloud - Comparison

Category	Slurm (HPC)	Kubernetes (Cloud)
Target use-case	Heavy, but “short-running” batch compute	Never failing, always available, always up-to-date end-user services
High Availability	Platform: not required Jobs: not required	Platform: required (9er rules) Services: required (9er rules)
Environment	Self-managed terminal env – scripting Singularity Containers	Entirely image/container-based
Automation	Mostly manual job triggers (can be scripted though)	GitOps oriented fully automated approach
Queuing/Batching	Native	Scheduling only (but Batching can be added, e.g. Kueue)
Backup/Recovery	Input/Result data: required Jobs: not required	Required for all services
Resource Sharing Model	Between all authorized parties (reservation possible)	Between all authorized parties
Main Adoption	Academia/Research	Industry

So what does that mean for “deSRC”

Options / Solutions

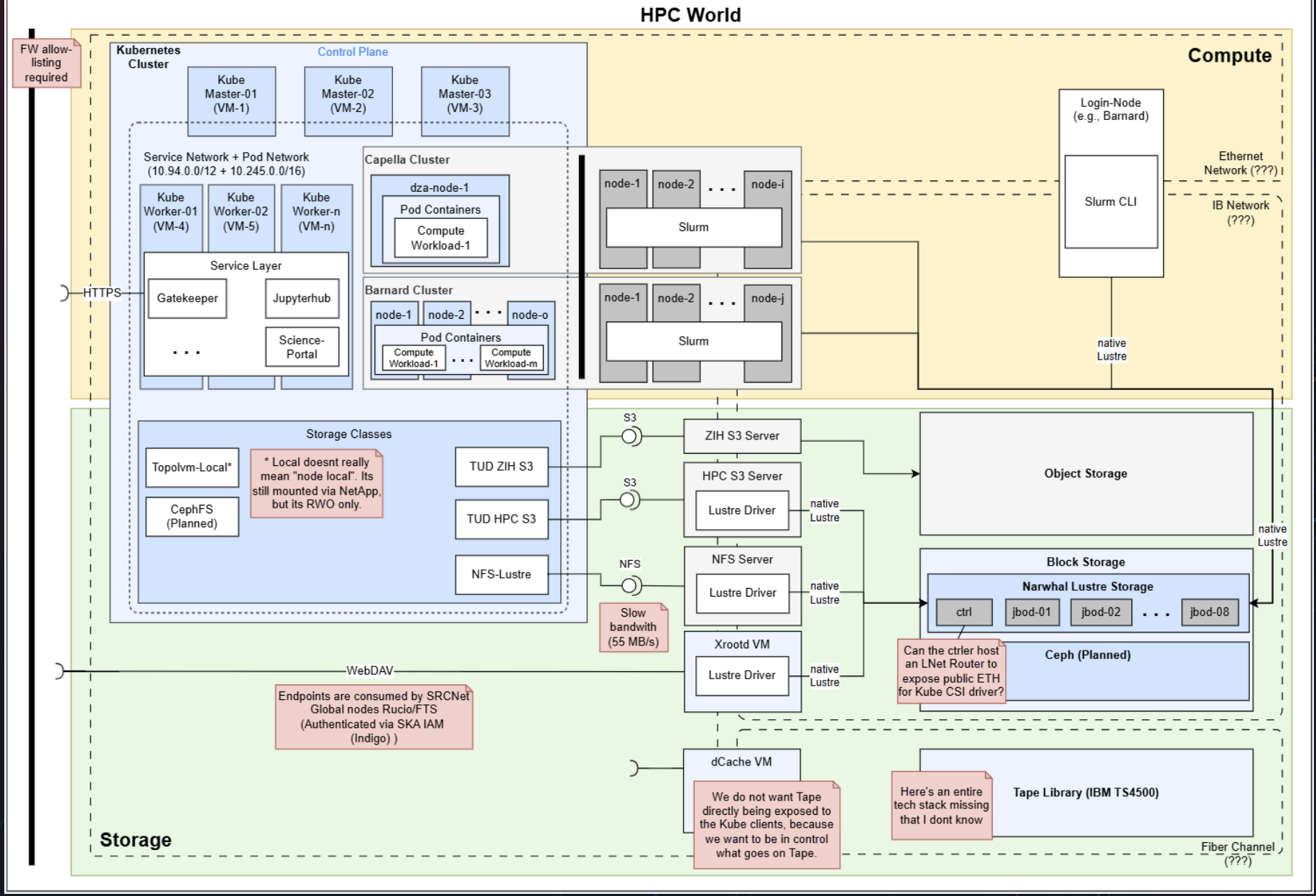
deSRC - constraints

- DZA will host the “deSRC node” from the datacenter at TU Dresden

- All our resources are added into the existing Slurm clusters Barnard/Capella
  - The initial assumption was:
    - This is yet another traditional academic intense compute research project
    - Nobody expected Kubernetes ☺

- We have to somehow make it work!

System Overview - Bare Metal nodes



Compute Gateway options - Interlink

- Comes out of European Project “InterTwin” (https://www.intertwin.eu/)

- Allows to connect together the Kube Control Plane with the `slurm` CLI
- Exposes the HPC resources (cluster) as a new virtual Kubelet (= a node inside the Kube cluster)
- On its way to become a CNCF project

- Overview: https://interlink-project.dev/

- Code: https://github.com/interlink-hq/interLink

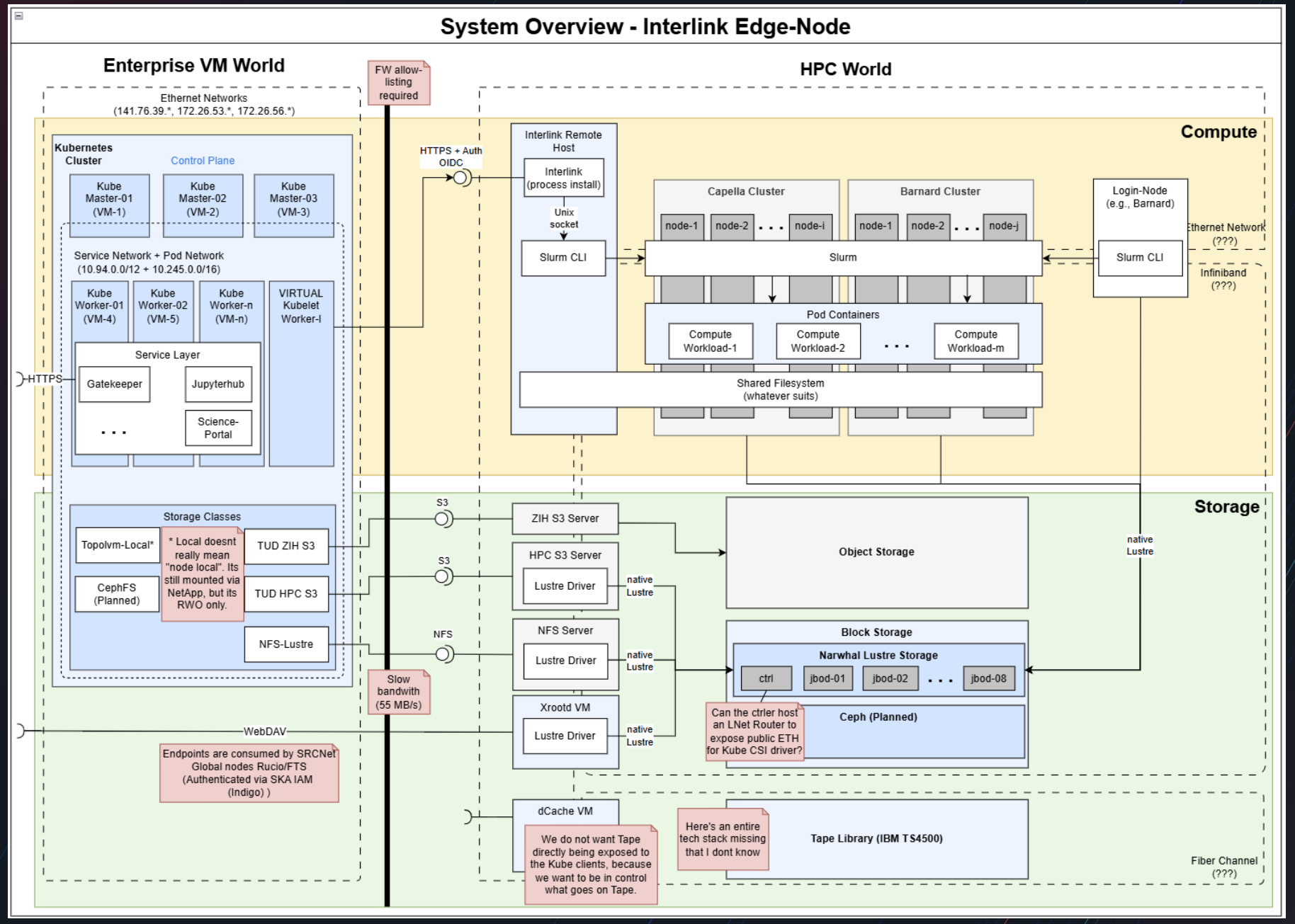
Compute Gateway options - Slinky

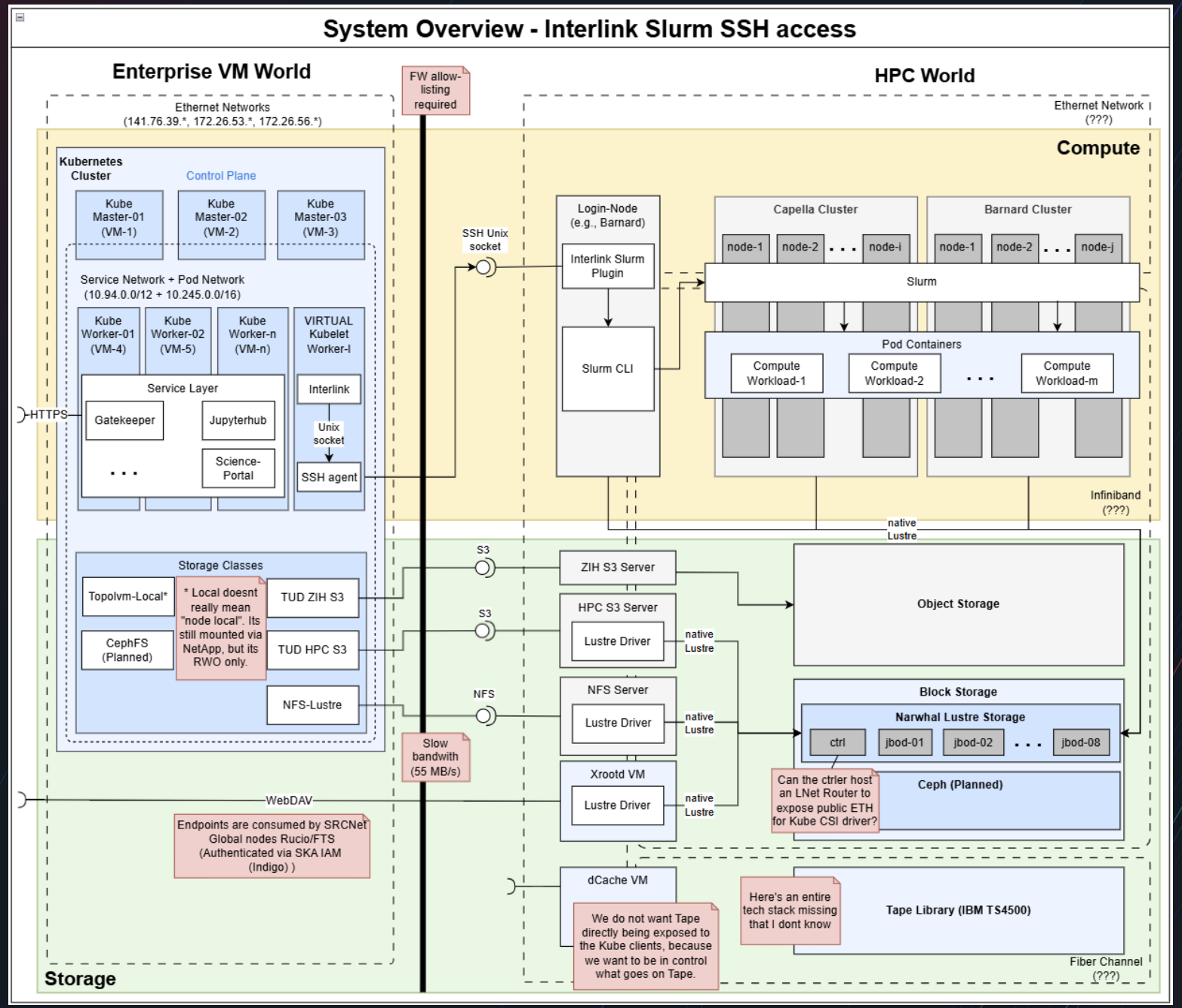
- Developed by SchedMD (the Devs of Slurm)

- Allows to connect together the Kube Control Plane with the `slurmctld` daemon
- Postulates 3 different operation scenarios:
  - Over
  - Adjacent
  - Under
- See Talk by Tim Wickberg at SC 2022
  - https://slurm.schedmd.com/SC22/Slurm-and-or-vs-Kubernetes.pdf

- Overview: https://www.schedmd.com/slinky/why-slinky/

- Code: https://github.com/slinkyproject


System Overview - Interlink Edge-Node
Enterprise VM World
HPC World
FW allow-listing required
Ethernet Networks
(141.76.39.*, 172.26.53.*, 172.26.56.*)
Kubernetes Cluster
Control Plane
Kube Master-01 (VM-1)
Kube Master-02 (VM-2)
Kube Master-03 (VM-3)
Service Network + Pod Network
(10.94.0.0/12 + 10.245.0.0/16)
Kube Worker-01 (VM-4)
Kube Worker-02 (VM-5)
Kube Worker-n (VM-n)
VIRTUAL Kubelet Worker-I
Service Layer
HTTPS
Gatekeeper
Jupyterhub
Science-Portal
HTTPS + Auth OIDC
Interlink Remote Host
Interlink (process install)
Unix socket
Slurm CLI
Compute
Capella Cluster
Barnard Cluster
node-1
node-2
node-i
node-j
Slurm
Pod Containers
Compute Workload-1
Compute Workload-2
Compute Workload-m
Shared Filesystem (whatever suits)
Login-Node (e.g., Barnard)
Slurm CLI
Ethernet Network (???)
Infiniband (???)
Storage
Storage Classes
Topolvm-Local*
* Local doesnt really mean "node local". Its still mounted via NetApp, but its RWO only.
CephFS (Planned)
TUD ZIH S3
TUD HPC S3
NFS-Lustre
S3
NFS
ZIH S3 Server
HPC S3 Server
Lustre Driver
NFS Server
Xrootd VM
native Lustre
Object Storage
Block Storage
Narwhal Lustre Storage
ctrl
jbod-01
jbod-02
jbod-08
Ceph (Planned)
Can the ctrler host an LNet Router to expose public ETH for Kube CSI driver?
Slow bandwith (55 MB/s)
WebDAV
Endpoints are consumed by SRCNet Global nodes Rucio/FTS (Authenticated via SKA IAM (Indigo))
dCache VM
We do not want Tape directly being exposed to the Kube clients, because we want to be in control what goes on Tape.
Here's an entire tech stack missing that I dont know
Tape Library (IBM TS4500)
Fiber Channel (???)


System Overview - Interlink Slurm SSH access
Enterprise VM World
HPC World
FW allow-listing required
Ethernet Networks
(141.76.39.*, 172.26.53.*, 172.26.56.*)
Ethernet Network
(???)
Compute
Kubernetes Cluster
Control Plane
Kube Master-01 (VM-1)
Kube Master-02 (VM-2)
Kube Master-03 (VM-3)
Service Network + Pod Network
(10.94.0.0/12 + 10.245.0.0/16)
Kube Worker-01 (VM-4)
Kube Worker-02 (VM-5)
Kube Worker-n (VM-n)
VIRTUAL Kubelet Worker-I
Service Layer
Gatekeeper
Jupyterhub
Science-Portal
Interlink
Unix socket
SSH agent
HTTPS
SSH Unix socket
Login-Node (e.g., Barnard)
Interlink Slurm Plugin
Slurm CLI
Capella Cluster
Barnard Cluster
node-1
node-2
node-i
node-j
Slurm
Pod Containers
Compute Workload-1
Compute Workload-2
Compute Workload-m
Infiniband
(???)
native Lustre
Storage Classes
Topolvm-Local*
* Local doesnt really mean "node local". Its still mounted via NetApp, but its RWO only.
CephFS (Planned)
TUD ZIH S3
TUD HPC S3
NFS-Lustre
S3
NFS
ZIH S3 Server
HPC S3 Server
Lustre Driver
NFS Server
Xrootd VM
Slow bandwith (55 MB/s)
WebDAV
Endpoints are consumed by SRCNet Global nodes Rucio/FTS (Authenticated via SKA IAM (Indigo))
Object Storage
Block Storage
Narwhal Lustre Storage
ctrl
jbod-01
jbod-02
jbod-08
Ceph (Planned)
Can the ctrler host an LNet Router to expose public ETH for Kube CSI driver?
dCache VM
We do not want Tape directly being exposed to the Kube clients, because we want to be in control what goes on Tape.
Here's an entire tech stack missing that I dont know
Tape Library (IBM TS4500)
Fiber Channel
(???)
Storage

Summary

Looks like a gap between HPC – Cloud?

Summary - Differences

1. The end-user workflows / processes to “do science” look different
    1. We move from “use a terminal” -> “use a webportal”
    2. No need for workstation data copy anymore
    3. “Change management” and “expectation management” for end-users is required
        1. Do they need to learn “kubectl” to interact with Kube or a “webfrontend” to interact with the Science Platform?
2. The underlying technical foundation changes
    1. We move from “use Slurm Job processing” -> “use Kubernetes service layer”
        1. (But do we really? Or will both enter in a marriage?)

Summary – Tech Challenges

1. Is it better to have one single beefy Kube cluster or schedule workload outside?

2. Can we reliably and consistently schedule workload via Kube on an existing Slurm cluster?

3. What is the best solution to make the same data accessible inside Kube for a service/app and inside Slurm for heavy computation (without copying the data)?
    1. Blockstorage vs ObjectStorage vs NFS vs etc.
    2. Can we avoid data copies between “user areas” vs “raw data areas”

4. What will be the “winning” orchestration strategy of the astro algorithm/workload?
    1. Push via packaged container as “standalone” workload in Kube?
    2. Push via “direct code” through Software platform (e.g., Jupyterhub, Canfar, etc.)?

5. And plenty plenty more …