RESILIENCE REALIZED

KubeCon | CloudNativeCon

North America 2021

RESILIENCE
REALIZED

Implementation challenges: From HPC to Containers in the Academy

Viktória Spišaková & Lukáš Hejtmánek, Masaryk University

Who we are

Lukáš Hejtmánek

Masaryk University

IT architect

Viktória Spišaková

Masaryk University

IT specialist

Overview

- Czech NREN e-Infra CZ operates HPC environment
- ~20k CPU cores, 200 GPUs, 60PB storage
- Computational resources accessible mostly through PBSPro
- Storage accessible through Kerberized NFSv4
  - Minority usage through S3, CEPH RBD
- ~1000 active users

HPC: Resources

- Compute resources
  - Users create shell scrips and run them via PBSPro
  - ssh experience required, no GUI
  - Open OnDemand — attempt to provide GUI
- Storage resources
  - Directly available on worker nodes, many storage locations!
  - Accessible from user’s computer

HPC: Troubles

- No straightforward way to monitor running computations
- Heritage of old unsupported scripts not working on updated/upgraded nodes
- Average UNIX skills required
- Access to storage is time limited
- Setting up a NFS client is a hard task

HPC: Containers

- Common existing containers
  - NGC containers
  - Biocontainers
- How to use them?
  - Docker mostly prohibited
  - Singularity tool
  - Podman
- Why not to use native container infrastructure?

Containers: Infrastructure

- Building shared container infrastructure
  - No need for users to deploy and maintain own infrastructure
  - Users focus on research and their work
- Alternate approach — run your own container infrastructure
  - OpenStack Magnum

Containers: e-Infra CZ

- Operating several Kubernetes clusters
  - Rancher + RKE 2
- User perspective
  - Native K8s access
  - Pre-deployed applications and frameworks
  - Rancher GUI

User Perspective

Native K8s access

- Own/shared project
- Namespace
- Persistent storage
    - NFS
    - CEPH RBD
    - S3
- GPU
- InfiniBand

Pre-deployed applications

- Jupyter Hub + Binder
- Galaxy
- Kubeflow
- 3D accelerated desktop

Frameworks

- GA4GH TES/WES
- Nextflow
- Snakemake

Containers: Benefits for Users

- No required knowledge of
    - Shell scripts, ssh, and CLI tools
    - Kerberos and NFS
    - NREN topology
    - Software modules and their dependencies
    - Way to run HPC containers

Containers: Challenges

- K8s — HPC integration
- Queueing and fairness
- Scheduling
- User trust

Challenges: K8s — HPC Integration

- How to integrate existing HPC infrastructure with K8s?
    - AAI
    - Compute Nodes
    - Storage

Challenges: K8s — HPC Integration

- AAI can be shared
- Worker nodes are easily shared between PBSPro and K8s
    - PBSPro K8s connector as an option
- Storage — real challenge

HPC Storage

- HPC usually utilises NFS, AFS
- How to access HPC storage from K8s?
- User authentication
    - Access tokens — do not understand namespace
        - How to renew the token?
    - UID only — most containers run as user 1000
        - UID remapping

HPC Storage

- NFS — UID remap ❌
    - Fast
    - Many CSI drivers
- sshfs — UID remap ✅
    - Slow
    - CSI driver must not restart
- CIFS — UID remap ✅
    - Acceptable performance
    - Not widely supported in HPC

Challenges: Queueing and Fairness

- Currently not present in vanilla K8s
- Do we need queuing system?
- We need fairness
  - Force fair use policy
- Resource quotas, priorities, is it all we need?

Challenges: Scheduling

- PBSPro contains complex scheduler
- K8s contains rather simple scheduler
- Should avoid pod starvation
- Pod eviction is a problem for HPC
- K8s resources without time limit

Challenges: User Trust

- Users are afraid of changes
  - Will it work?
  - Is it stable?
  - Will it survive next year?
- Build better portals

Future Plans

- Continue transition from PBSPro to K8s
- Experimental setup
    - Worker nodes with large SSD
    - Build fast shared storage
    - Provide reasonable data redundancy

Conclusion

- Providing unified container infrastructure in e-Infra CZ
  - Multi-tenancy
  - Suitable for web services and HPC
  - Already running HPC workloads

Thank you for your attention