Paving the Way for HPC: An XRootD-Based Approach for Efficiency and Workflow Optimizations for HEP Jobs on HPC Centers

Robin Hofsaess[1,*], Manuel Giffels[1], Artur Gottmann[1], Maximilian Horzela[1], Andreas Petzold[1], Günter Quast[1], Matthias Schnepf[1], Achim Streit[1]

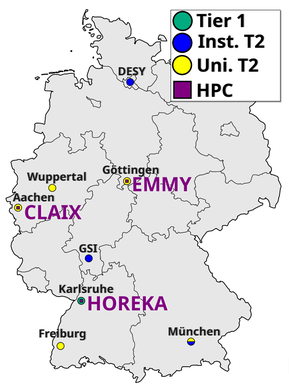
[1] Karlsruhe Institue of Technology (KIT)

The Future of German HEP Computing

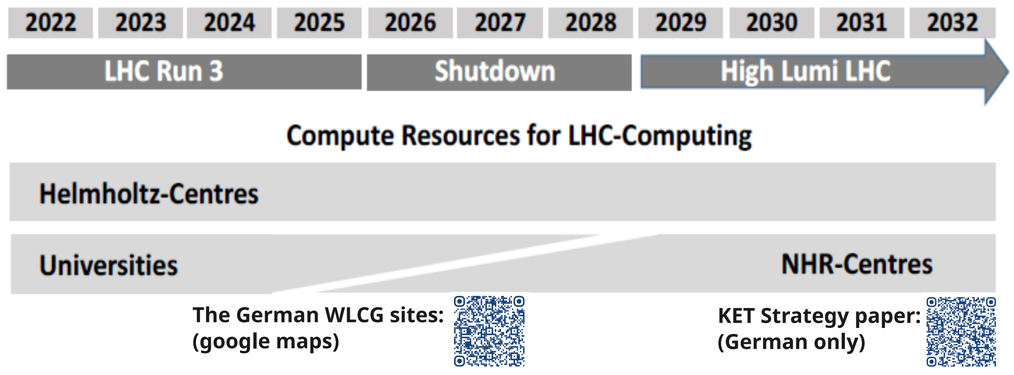
- The University Tier-2s will be **replaced** by shares on three national HPC centers
- All data will be moved to the Helmholtz Centers (**KIT** and **DESY**)

Challenges:

- HPC centers are comparably **heterogeneous** (different HW, SW, policies, permissions, monitoring, ...)
- **Data intensive workflows** are not inherently suited for all HPC centers



https://nhr-verein.de/en



The German WLCG sites: (google maps)

KET Strategy paper: (German only)

Prototype Integration of HoreKa @ KIT

- HoreKa is integrated as **opportunistic resource** since over 3 years into GridKa, the German Tier-1 center
- For the dynamic integration, we use COBaID/TARDIS, developed at KIT
- HoreKa is one of the **R&D sites** used to test and develop concepts for a smooth transition away from the dedicated university Tier-2 centers

Observations at HoreKa:

- Higher job **failure rates** compared to the Tier-1 (GridKa) and Tier-3 (TOpAS) center at KIT
- Comparably **low CPU efficiency** and high I/O wait

→ Both are attributable to **remote transfers** and slow external bandwidth of the worker nodes

Solution: Data Access Bottleneck Mitigation with an XRootD Cache as *Buffer*

Idea

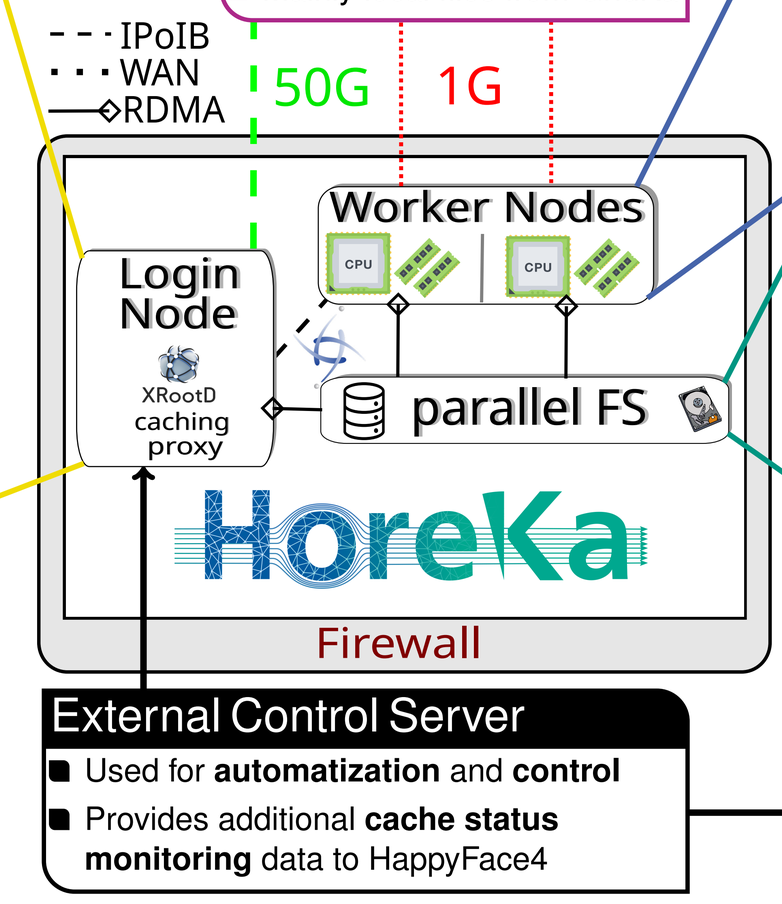
- Redirect transfer requests to an **XRootD caching proxy** on the login node (PoC)
- The proxy then streams the data, exploiting the **faster external bandwidth** to mitigate data access bottlenecks
- With a moderate **prefetching**, the XRootD proxy acts as a sort of *buffer* for the data transfers
- Additionally: **fully** cached files are provided from the **local** filesystem via RDMA

Benefits:

- Faster remote transfers (up to 50G)
- caching accelerate recurrent transfers

Remote Data Servers

- Connected via the internet
- Mainly local files from GridKa



Worker Nodes

- 2021 Intel Xeon 76c/152t machines
- 1G Ethernet (**external**) connection
- 200G **internal** IPoIB
- **parallel filesystem** with RDMA
- Running grid jobs in **Apptainer** pilots

Parallel Filesystem

- GPFS with RDMA over Infiniband
- Up to **150G** for data transfers from local cache
- **250TB** share for (volatile) XRootD **cache**
- Mounted on login and worker nodes

Results

- Greatly **reduced failure rates** (now comparable to the Tier-1 and Tier-3)
- Comparable **CPU efficiency** distribution of jobs
- However, still **dependent** on the job mix
- ***Remark:*** *Currently, only tested with CMS jobs*

External Control Server

- Used for **automatization** and **control**
- Provides additional **cache status monitoring** data to HappyFace4

Monitoring

- We develop a **meta monitoring** tool at KIT, ideal for enhanced monitoring: HappyFace4
- The **experiments** provide job monitoring and statistics

→ Monitoring is **essential** for reliable operation and benchmarking

Conclusion and Outlook

- Our **XRootD-based** Proof of Concept increases the **reliability** and **efficiency** of CMS jobs running at **HoreKa**, our local HPC center
- The job execution benefits from the **faster bandwidth**
- **Cache hits** on the parallel filesystem accelerate the transfers significantly

For the **future**, further optimizations are planned:

- Tweaking of XRootD **config parameters** for the caching proxy performance, like RAM, or the prefetching level
- Dedicated transfer nodes with 100G+ access to LHCOne or a firewall bypass directly to GridKa

Code release coming soon:
https://github.com/RHofsaess

* Robin.Hofsaess@cern.ch https://github.com/RHofsaess

ACAT 2024, Stony Brook University