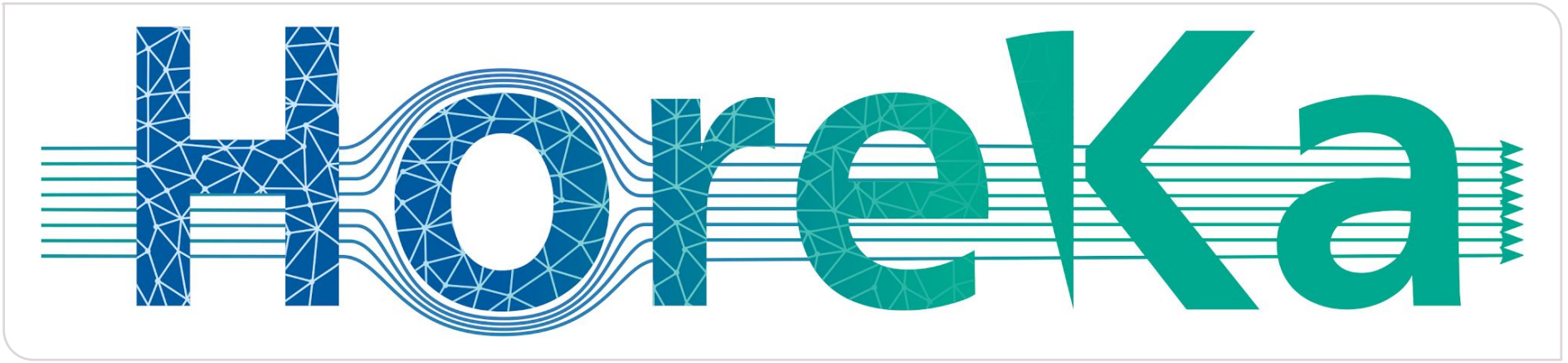
Karlsruhe Institute of Technology

Institute of Experimental Particle Physics

Optimizations for HEP Jobs on HPC with XRootD Caching

Robin Hofsaess for the GridKa R&D team



NATIONALES HOCHLEISTUNGS RECHNEN

Future German HEP Computing Strategy

Plan:

- Transition from dedicated T2 resources at universities to shares on national HPC centers within the NHR computing compound
- Storage at the Helmholtz centers KIT and DESY
- Support from the current T2 groups

Why HPC?

- Use resources that are available anyway
- Increased efficiency and sustainability
- Enhanced collaboration

Challenges:

- No influence on the hardware setup
- From admin to a user among many
- Different compute models
- WLCG **pledges**

Future German HEP Computing Strategy

Plan:

- Transition from dedicated T2 resources at universities to shares on national HPC centers within the NHR computing compound
- Storage at the Helmholtz centers KIT and DESY
- Support from the current T2 groups

Challenges:

- No influence on the hardware setup
- From admin to a user among many
- Different compute models
- WLCG **pledges**

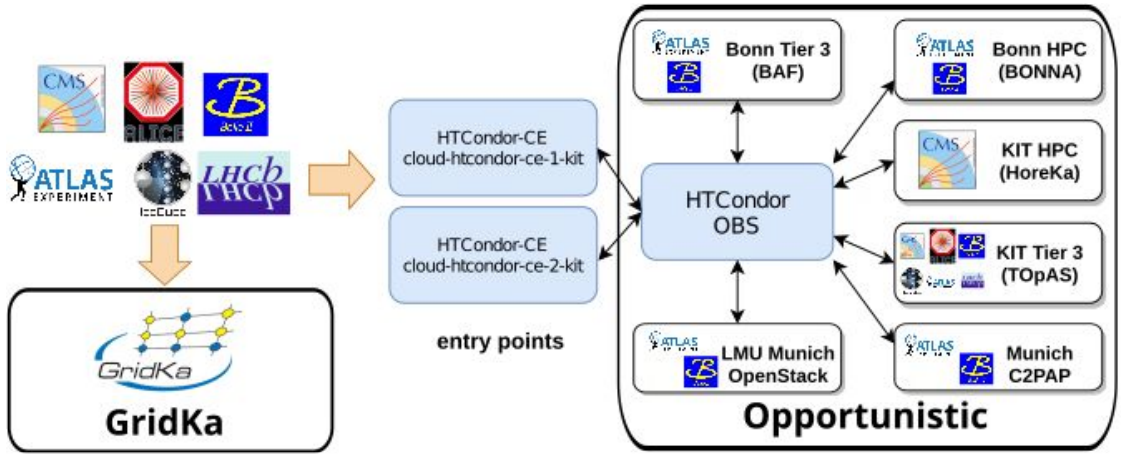
Why HPC?

- Use resources that are available anyway
- Increased efficiency and sustainability
- Enhanced collaboration

A reliable and efficient integration of HPC requires R&D and testing!

Utilization of HPC for WLCG Jobs

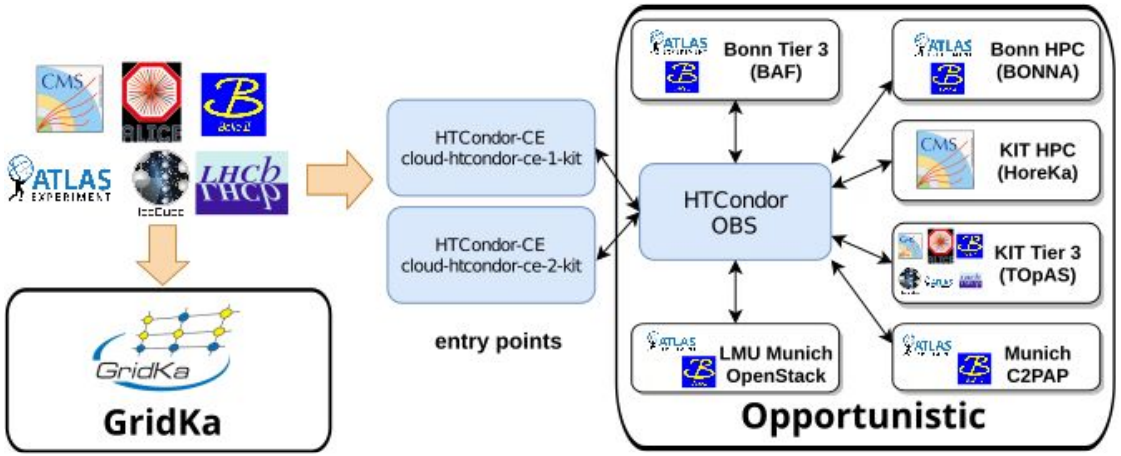
Integration @KIT:



HoreKa is integrated **opportunisti-cally** with [COBaID/TARDIS] since more than **three** years

Utilization of HPC for WLCG Jobs

Integration @KIT:



HoreKa is integrated **opportunisti-cally** with [COBaID/TARDIS] since more than **three** years

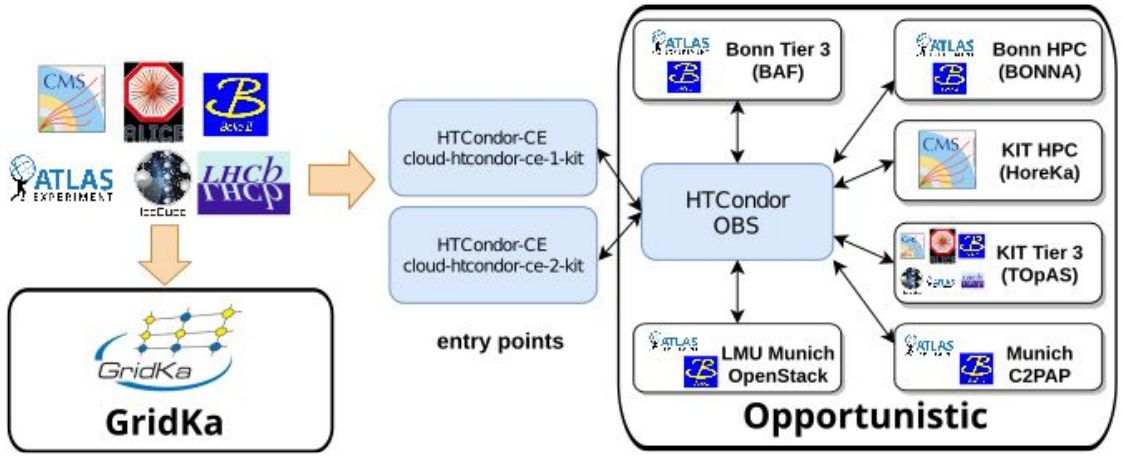
Observations

In comparison to T1/T3:

- Increased failure rate
- lower CPU efficiency

Utilization of HPC for WLCG Jobs

Integration @KIT:



HoreKa is integrated **opportunisti-cally** with [COBaID/TARDIS] since more than **three** years

Observations

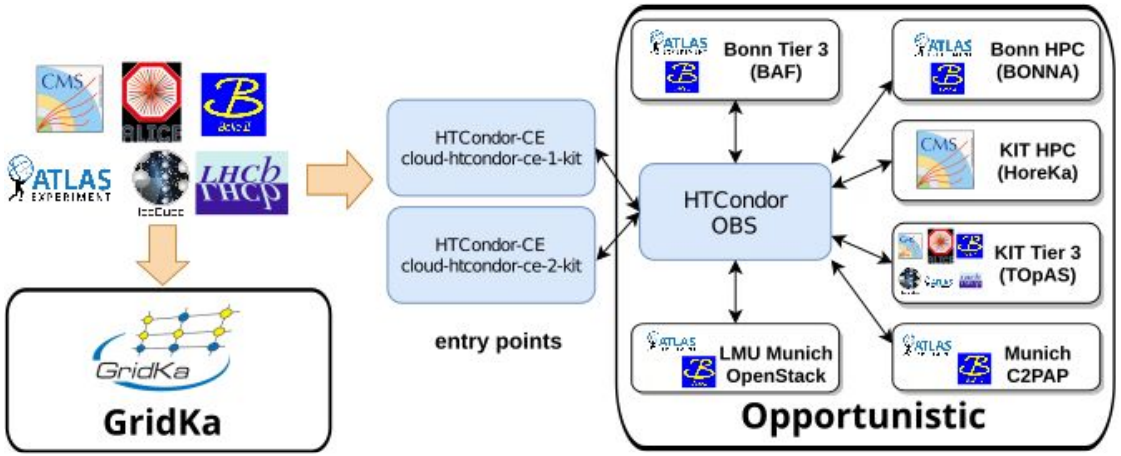
In comparison to T1/T3:

- Increased failure rate
- lower CPU efficiency

Why and how to find out?
ACAT Contribution

Utilization of HPC for WLCG Jobs

Integration @KIT:



HoreKa is integrated **opportunistically** with [COBaID/TARDIS] since more than **three** years

Observations

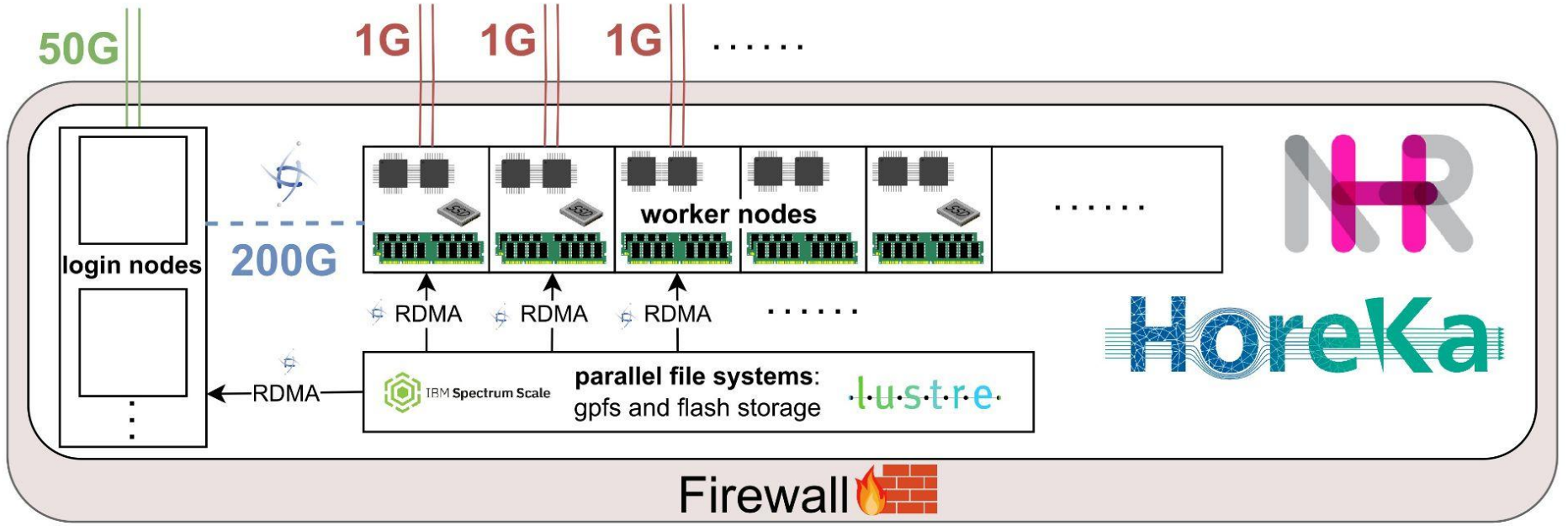
In comparison to T1/T3:

- Increased failure rate
- lower CPU efficiency

Why and how to find out?
ACAT Contribution

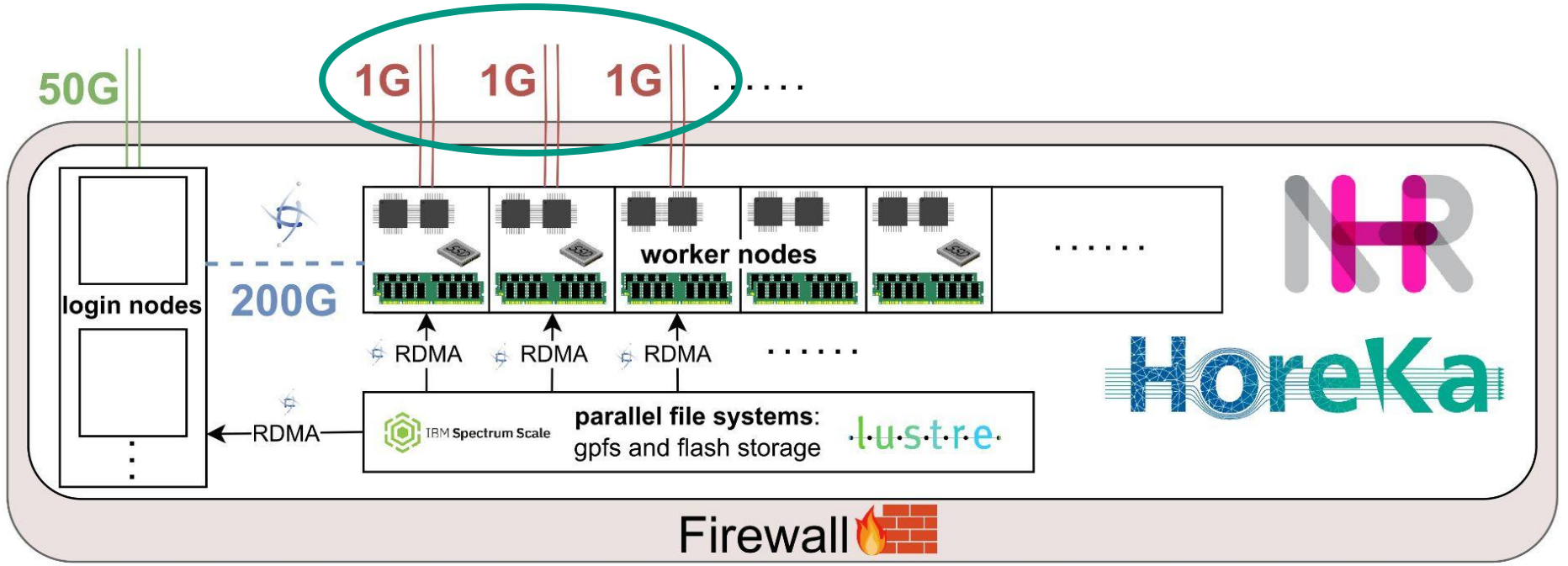
The reasons can be manifold and require a close look at each individual HPC center!

HoreKa Overview



- Separate test projects per VO
- A CMS **test project** got approved recently
- **Additionally:** backfilling of WLCG jobs dependent of cluster utilization

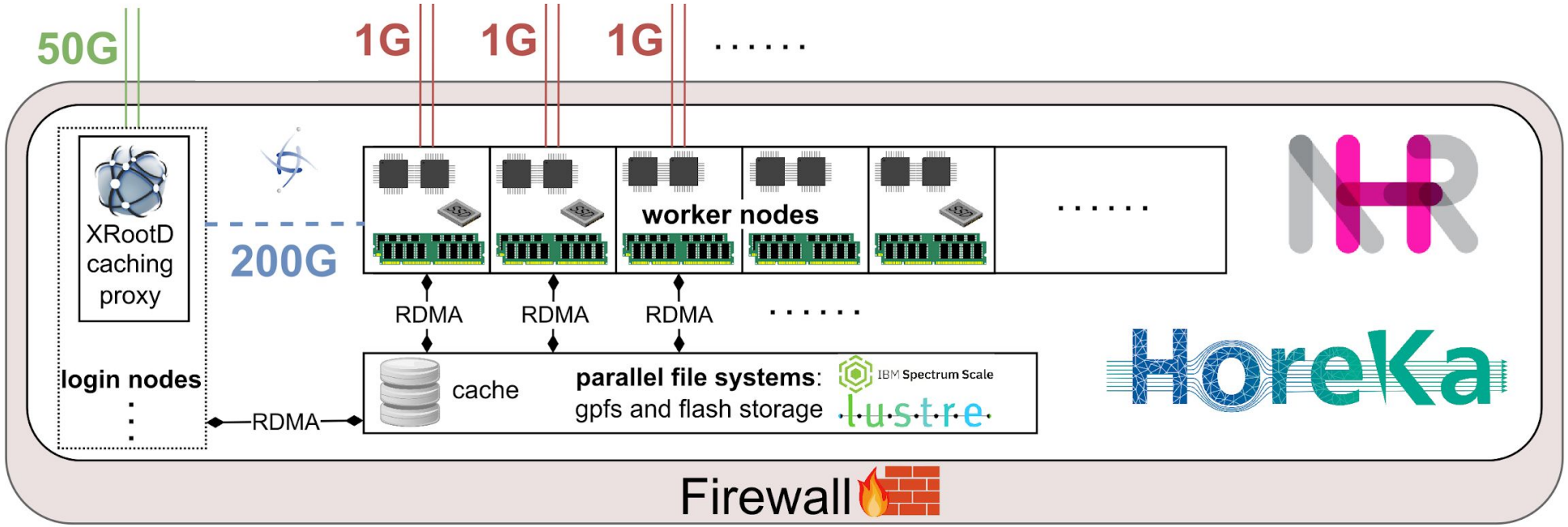
HoreKa Overview



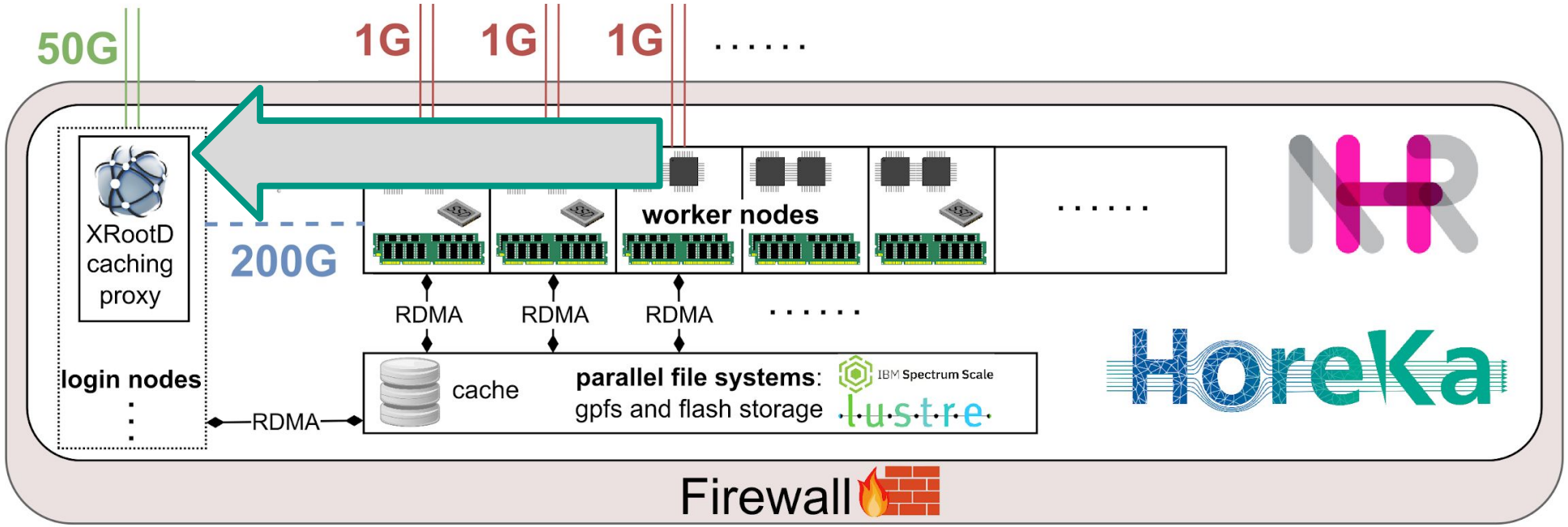
- Separate test projects per VO
- A CMS **test project** got approved recently
- **Additionally:** backfilling of WLCG jobs dependent of cluster utilization

Bandwidth limitations were identified as the **main bottleneck** for **data intensive** workflows

Data-Access Bottleneck Mitigation with XRootD

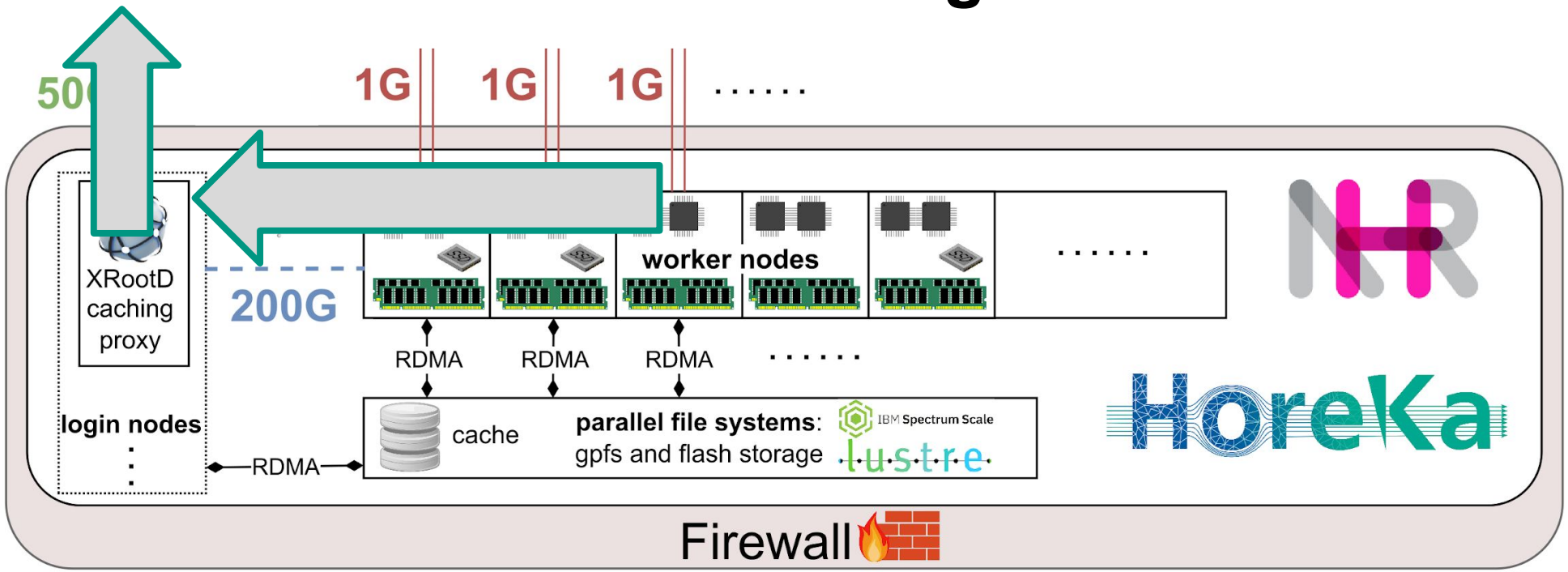


Data-Access Bottleneck Mitigation with XRootD



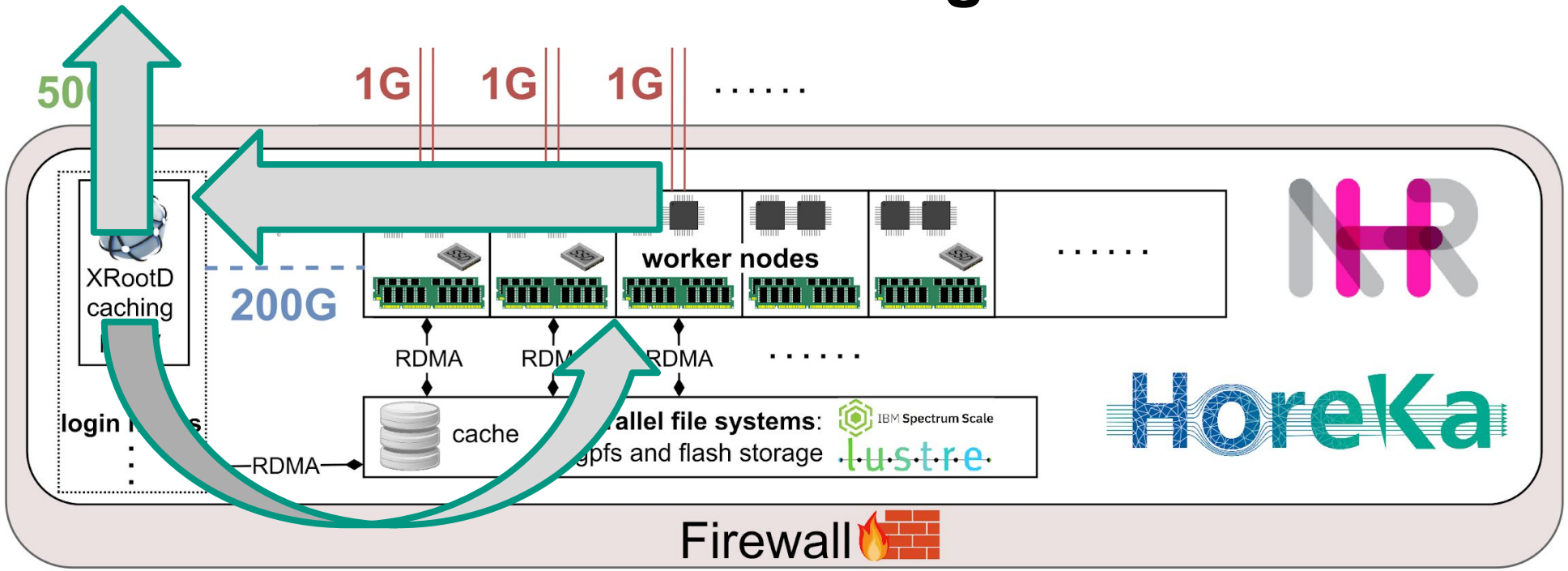
- The traffic is proxied over a login node via XRootD (50G connectivity)
- Remote data can be cached on the fly on the parallel filesystem
- Conceptually, this corresponds to a **buffer**

Data-Access Bottleneck Mitigation with XRootD



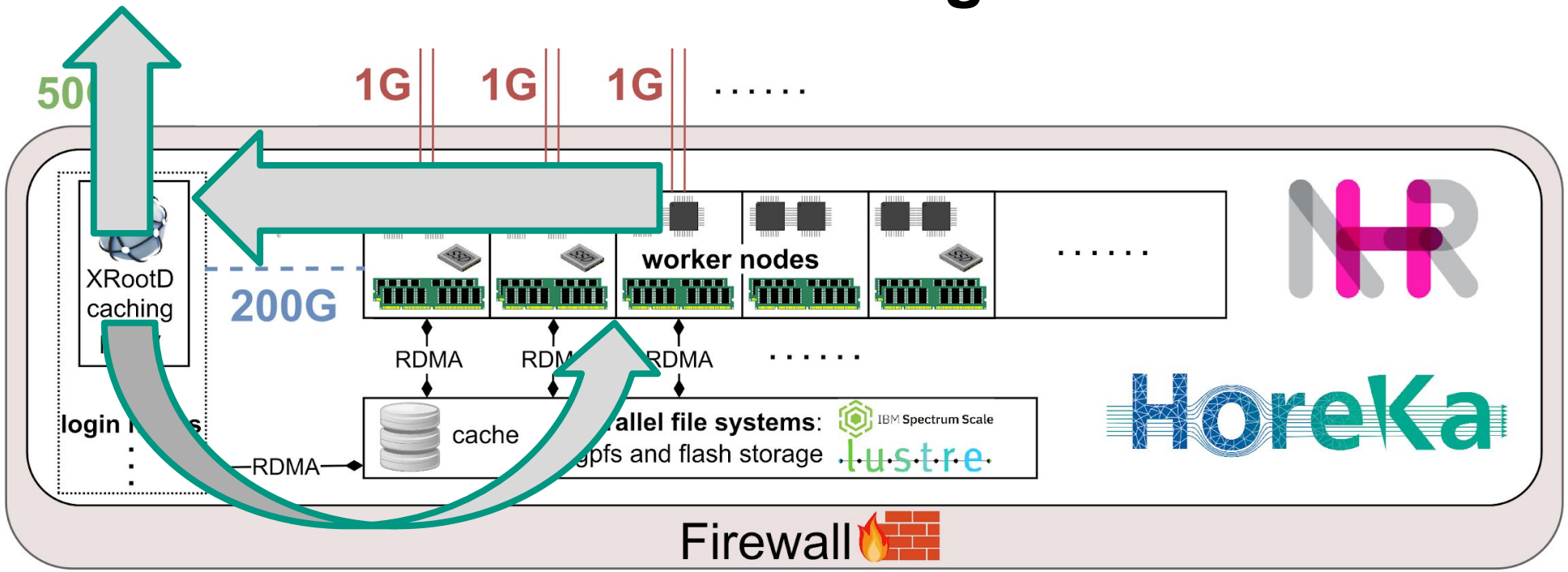
- The traffic is proxied over a login node via XRootD (50G connectivity)
- Remote data can be cached on the fly on the parallel filesystem
- Conceptually, this corresponds to a **buffer**

Data-Access Bottleneck Mitigation with XRootD



- The traffic is proxied over a login node via XRootD (50G connectivity)
- Remote data can be cached on the fly on the parallel filesystem
- Conceptually, this corresponds to a **buffer**

Data-Access Bottleneck Mitigation with XRootD



Now deployed in production for further testing!

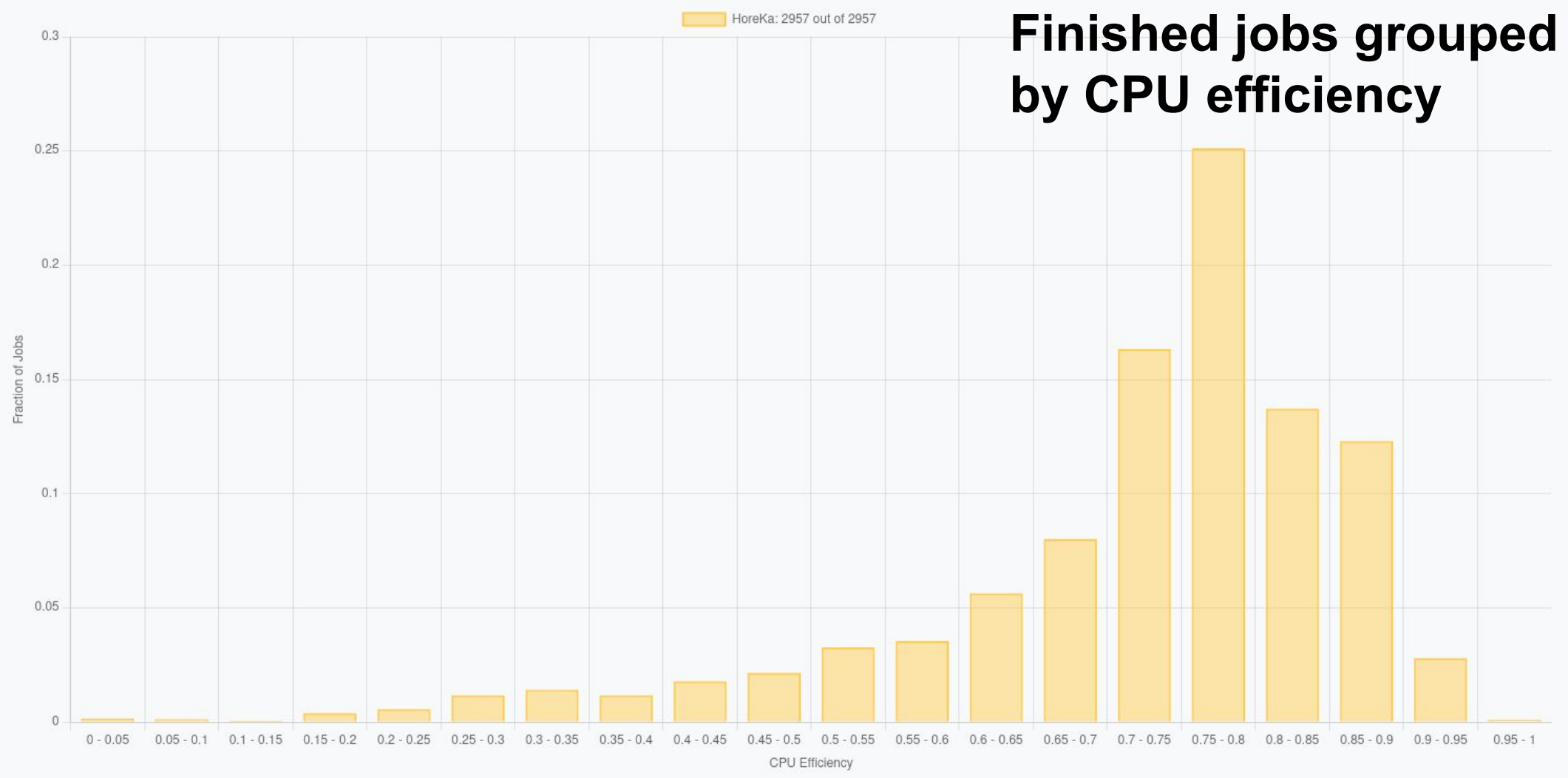
- The traffic is proxied over a login node via XRootD (50G connectivity)
- Remote data can be cached on the fly on the parallel filesystem
- Conceptually, this corresponds to a **buffer**

Advantages of our Concept

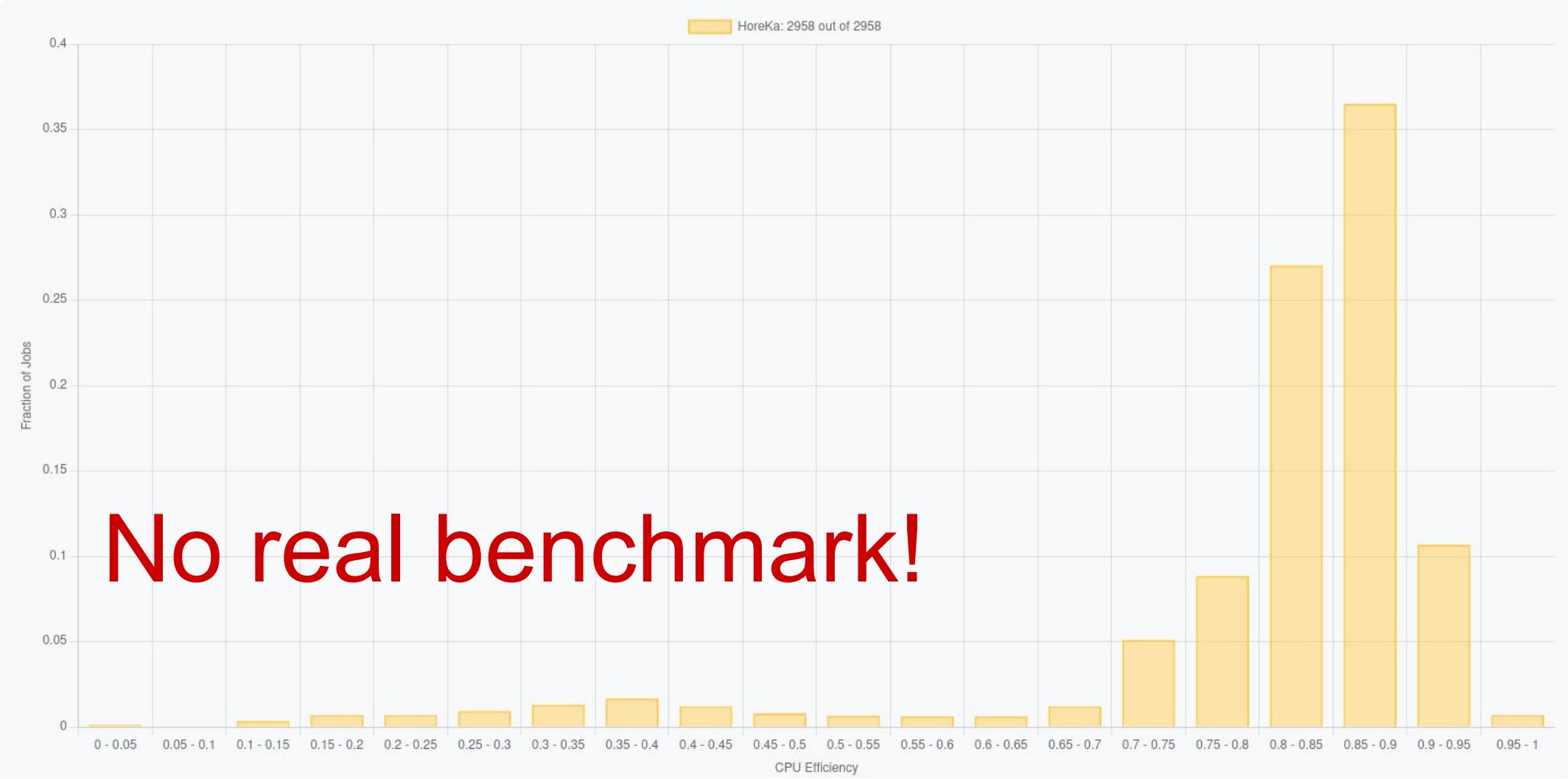
Faster Connectivity:

The **prefetching** of data over the login node accelerates the remote data access.

Preliminary Results: No Caching Proxy



Preliminary Results: With Caching Proxy



No real benchmark!

Advantages of our Concept

Faster Connectivity:

The **prefetching** of data over the login node accelerates the remote data access

Caching:

Cache hits may increase the transfer speed a lot (up to **several GB/s**!)

To cache or not to cache? That is the question!

- This is a **tough** decision:
  - Can be **very useful** with a decent cache **hit rate**:
    - Depends on job mix, data sets, cache size, etc
    - E.g.: very positive experiences with caching of **user analysis** data in Spain (ACAT poster)
  - From our experience on HPC:
    bleeding edge and still **error-prone**
  - **BUT**: Very **fruitful collaboration** with the **XRootD** team for improvements [*1*, *2*]

- Currently **hard to decide**
- Further testing and optimizations are **necessary and planned**
- We are working on additional **studies and benchmark tests**

Advantages of our Concept

Faster Connectivity:

The **prefetching** of data over the login node accelerates the remote data access

Caching:

Cache hits may increase the transfer speed a lot (up to **several GB/s**!)

Additional Benefits:

- Adds **monitoring** (very valuable for site operation!)
- Opens **new ways** of integrating HPC centers

Conclusion and Outlook

- We have developed and deployed a PoC at HoreKa
- The setup runs very smooth and is easily adaptable
- Failure rate and CPU efficiency overall improved and are comparable with limited jobmix
- Also beneficial without a data access bottleneck
- The **XRootD proxy cache** helps a lot in terms of **site operation** by providing **logs/monitoring** data

The **prefetching** over the login node effectively mitigates remote data access bottlenecks

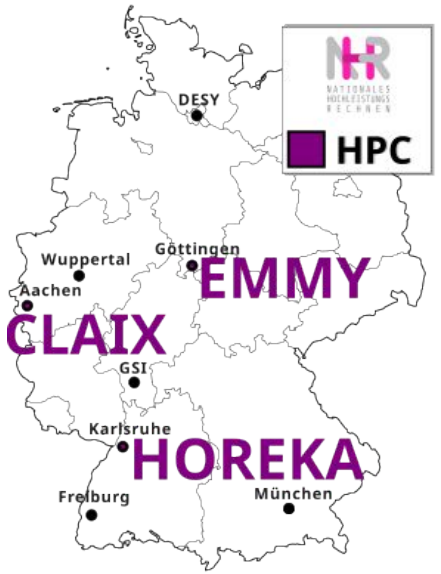
Caching requires more studies but may be useful in a future German data lake infrastructure with further adaptions

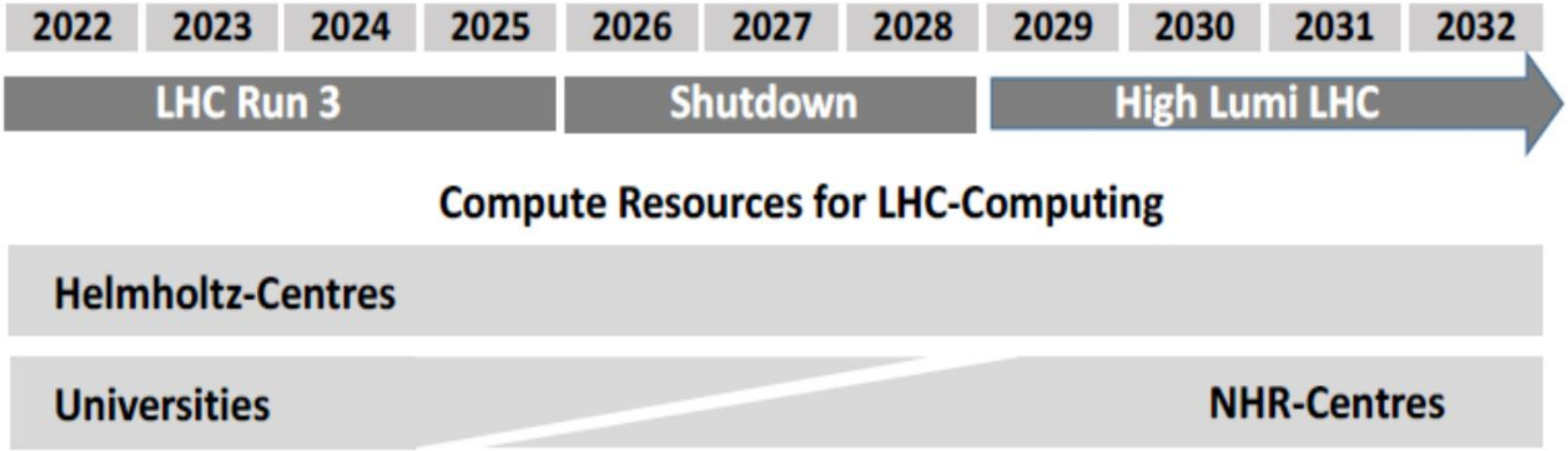
Our concept adds **further benefits and possibilities** to optimize the HPC integration besides the **transfer speed increasement** for data access

Karlsruhe Institute of Technology

THANKS!

German HEP Computing Strategy

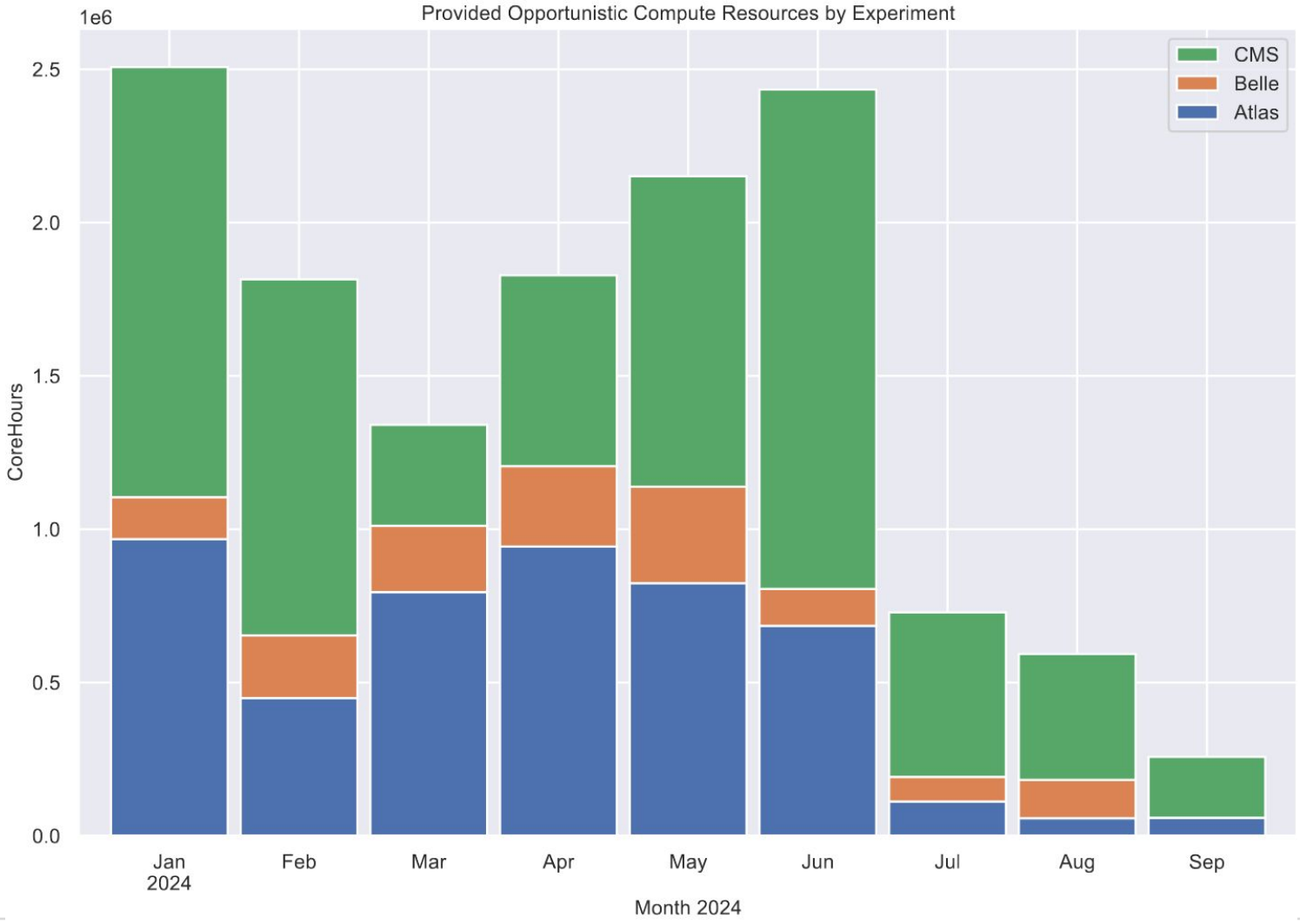




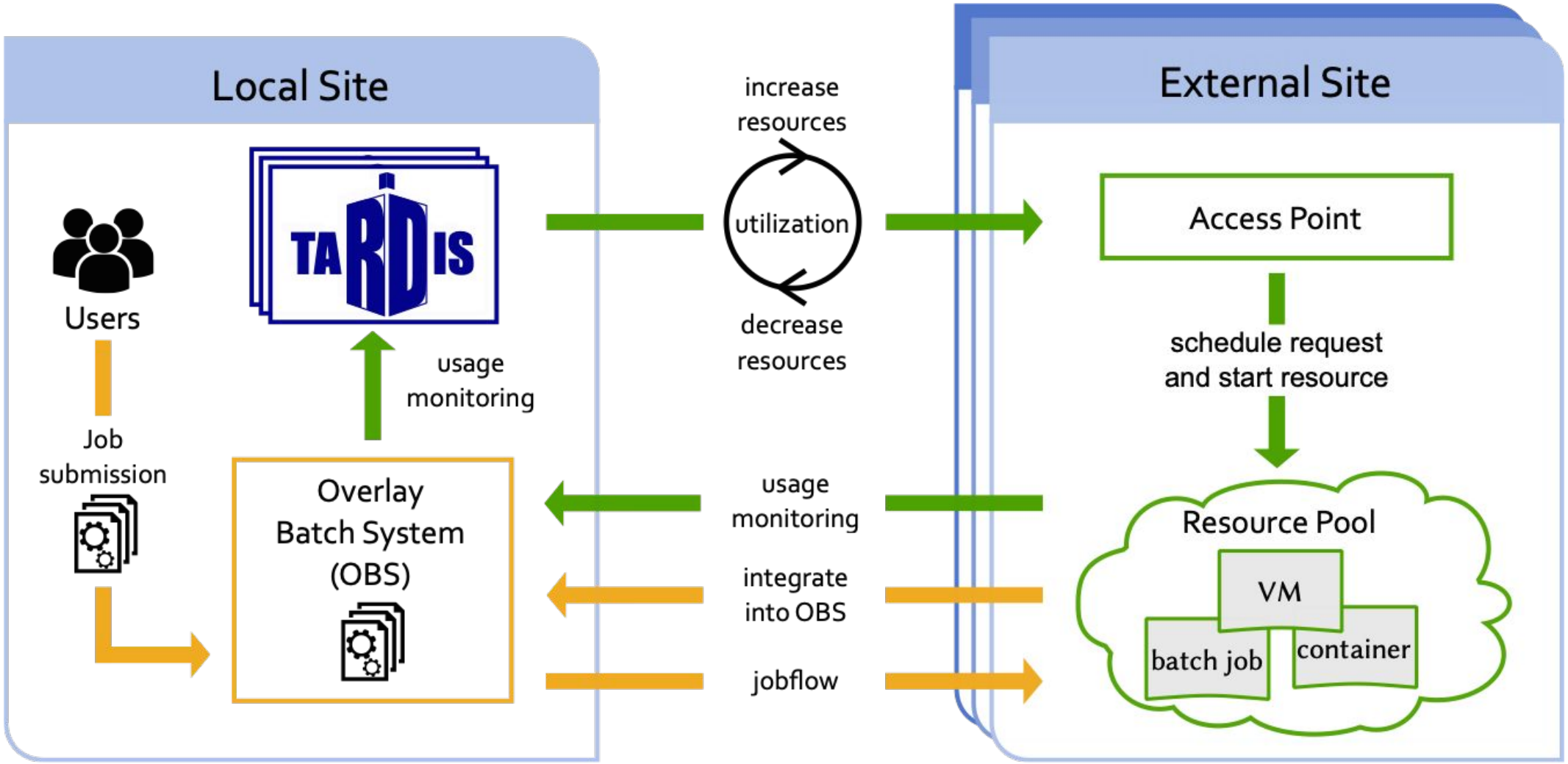
- Transition from dedicated T2 resources at universities to shares on national HPC centers within the NHR computing compound
- Storage at the Helmholtz centers KIT and DESY
- Support from the current T2 groups

strategy paper

Opportunistic Contribution in Germany

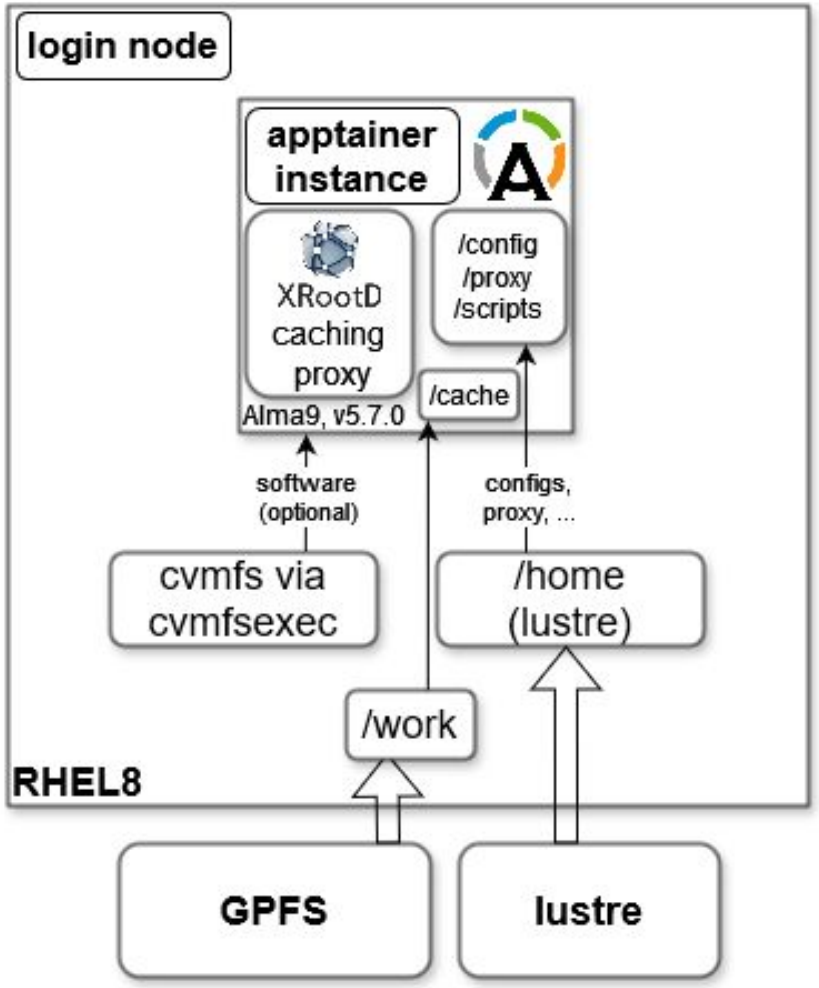


COBaID/TARDIS



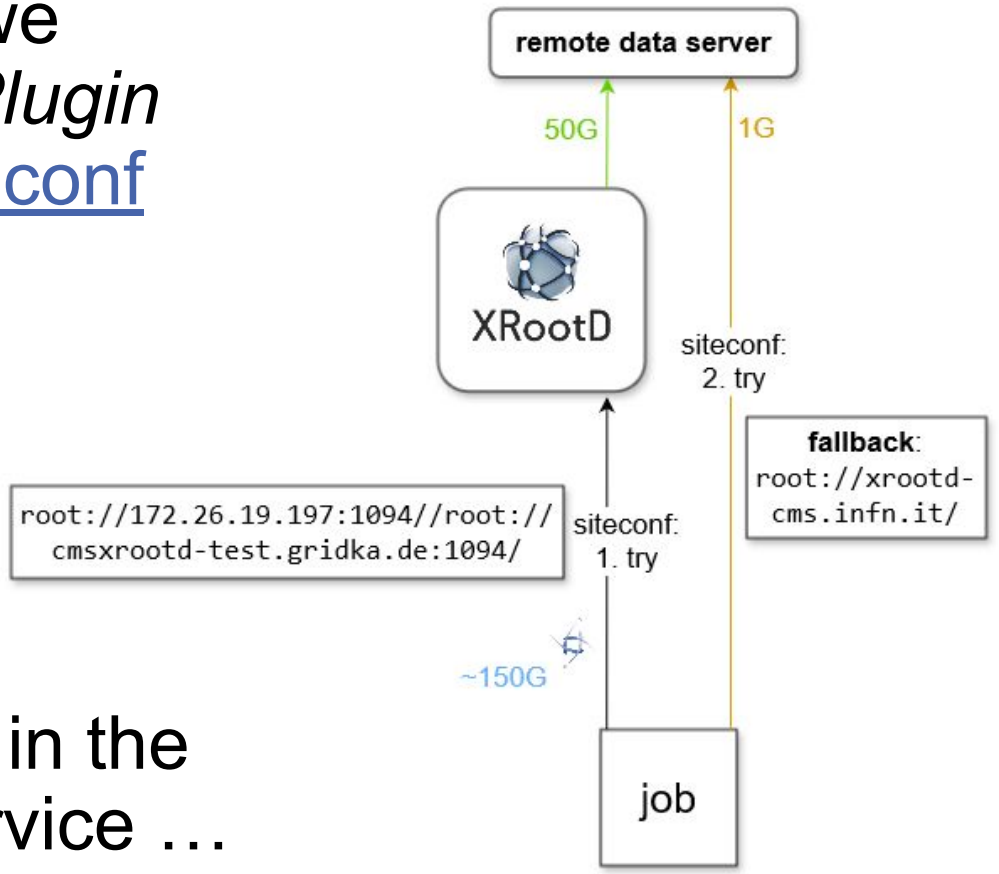
Setup and Configuration: Overview

- Host: RHEL8
- usernamespaces, CGroups v2, systemd user services
- Currently running: XRootD v5.7.1 as Alma9 apptainer instance (image bootstrapped from docker)
- In principle up to 76c and 500GB RAM, 50G WAN
  - But shared with other users (limitation via apptainer instance with CGv2 possible)
  - Usage: 32t, 64GB memory
- 250T quota on gpfs (via IB)



Setup and Configuration: Connection

- For enabling the proxy for transfers, we currently do **not** use the *XrdClProxyPlugin*
- Instead, the proxy is added to the siteconf directly
- Advantages:
  - We can use the intended fallback mechanism if smth fails
  - It is only enabled for file reads
- Disadvantages:
  - Changes require always a change in the repo and a full shutdown of the service …



Setup and Configuration: RDMA

- We currently don't use RDMA natively, but IPoIB for the transfers
- Currently, only IPv4 is possible (no link local v6 addresses in xrootd)
- The cache is also mounted via IPoIB (RDMA)
- We got some complaints from the GPFS team because of the many, many IOPS
- Reason: small blocksizes when caching is enabled
- ideally (FS PoV): pfc.blocksize == FS blocksize, or in general: as big as possible

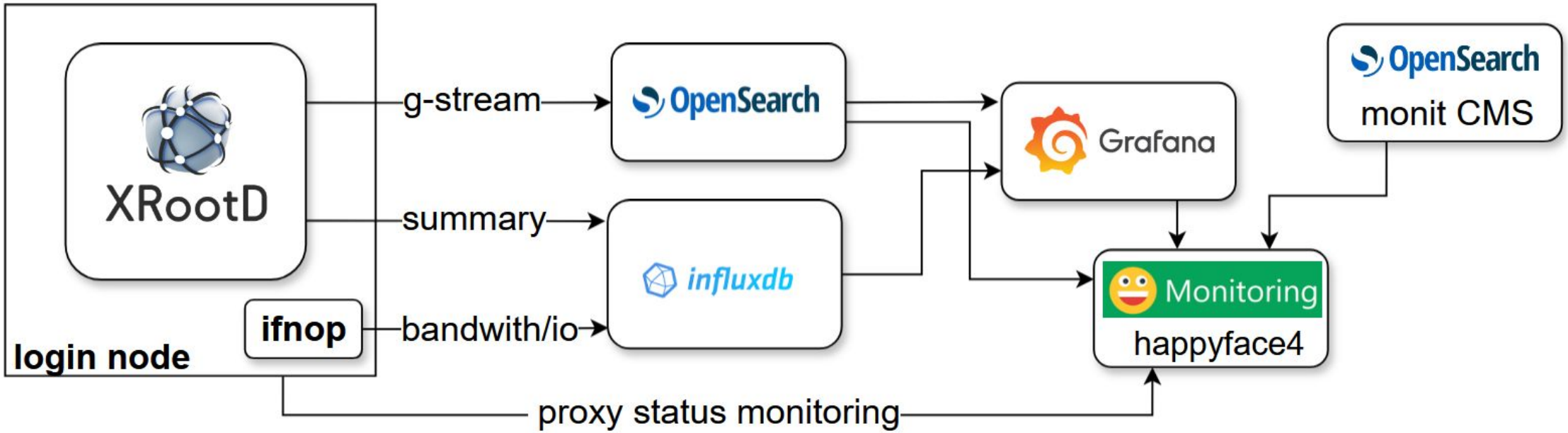
- dca:
  - Tested, but problematic with containerization
  - Dependent of the campaigns/datasets, e.g. premix rarely completely cached -> in production rather pointless
  - *Would be very useful, if made possible for partially cached files – if possible.*

Job Mix

- Currently, we only run a subset of the full job mix
- This works well, even for more data intensive WFs
- A full replacement will require the full mix, including *Analysis*
- Caching may be useful for a subset of the job mix (e.g. not RAW)
  - TBD with more monit/benchmarks

- Our proposal:
  - Even the full replacement should not run jobs like *Merge*, as they are just too inefficient on the expensive HPC hardware
  - To achieve the best possible efficiency, it is crucial to consider that

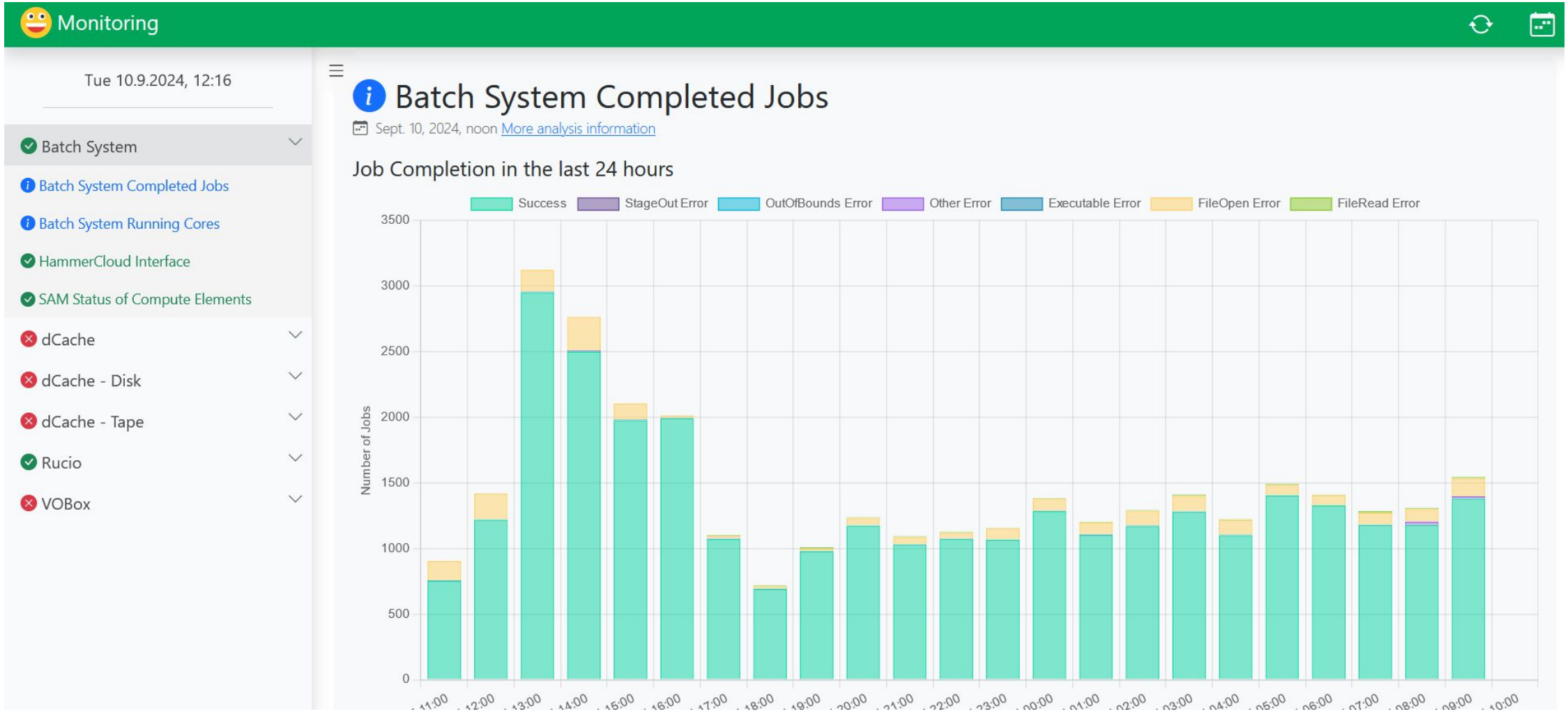
Monitoring



- Many additional monitoring capabilities thanks to XRootD
- Especially cache summary monitoring would be helpful (*on the way*)
- We collect everything and unify it in our meta-monitoring: HappyFace4
- I learned a lot about CMS and created some tools: I, II

Note: not yet fully in production/public

Monitoring


Monitoring
Tue 10.9.2024, 12:16
Batch System
Batch System Completed Jobs
Batch System Running Cores
HammerCloud Interface
SAM Status of Compute Elements
dCache
dCache - Disk
dCache - Tape
Rucio
VOBox
Batch System Completed Jobs
Sept. 10, 2024, noon More analysis information
Job Completion in the last 24 hours
Success
StageOut Error
OutOfBounds Error
Other Error
Executable Error
FileOpen Error
FileRead Error
Number of Jobs
3500
3000
2500
2000
1500
1000
500
0

Benchmark Mechanism

- We are working on requestable benchmarking and debugging WFs
  - Will help for debugging, as it provides a testing scenario that is well understood and can be more verbose (in comparison to production jobs – e.g. no debug flags etc)
  - Will provide a more comparable benchmark mechanism for site comparison – e.g. to answer:
    “Did my recent changes actually work or am I just lucky with the scheduling”
- Oriented on the ARM RelVal workflows

Improvements and Plans

- Further improvements:
  - Caching improvements: E.g. only cache certain sites
  -
- Plans:
  - Make the most out of all monitoring
  - Switch to tokens
  - Investigate feasibility and caching efficiency
  - Develop a benchmark mechanism
  - Further data analysis for comparing HPC with “normal” grid sites
  - Documentation and publishing coming soon