Sebastian Oeste
Center for Information Services and High Performance Computing (ZIH)

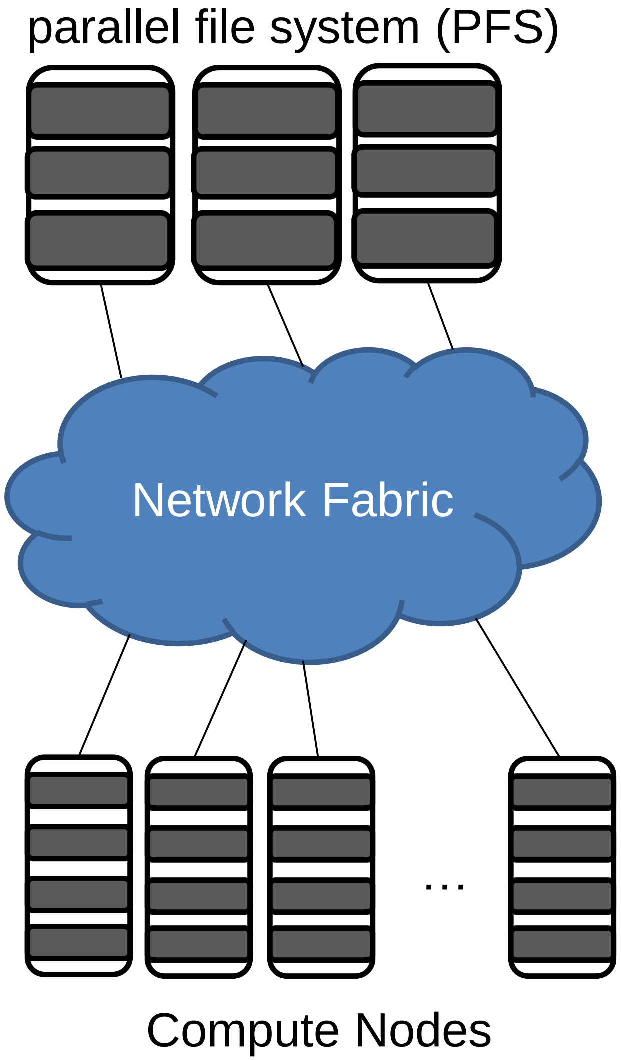
Exclusive file systems for power users with BeeGFS and network NVME storage

NHR-PerfLab

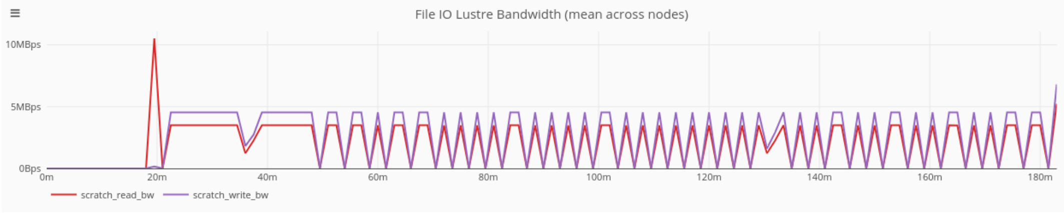
20.07.2021

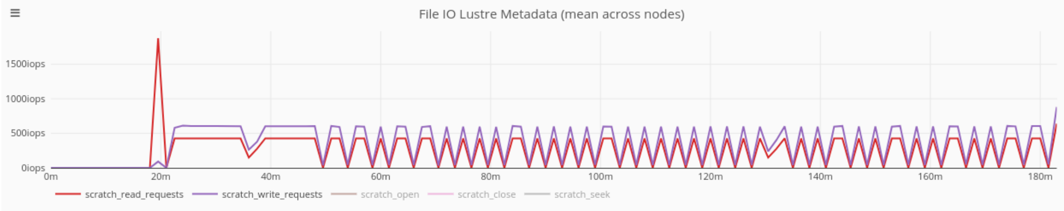
Center for Information Services & High Performance Computing

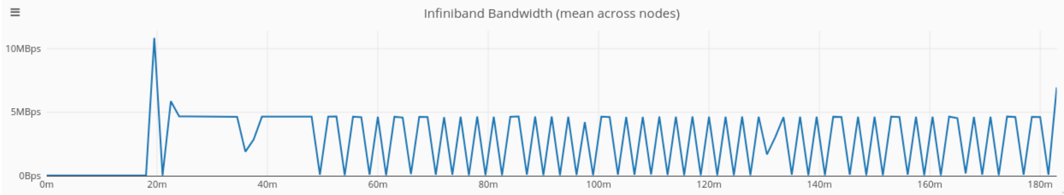
I/O is global



- Parallel I/O affects the whole HPC-Cluster.
- Different workloads access shared resources at the same time.
- Strong correlation of file system and network performance

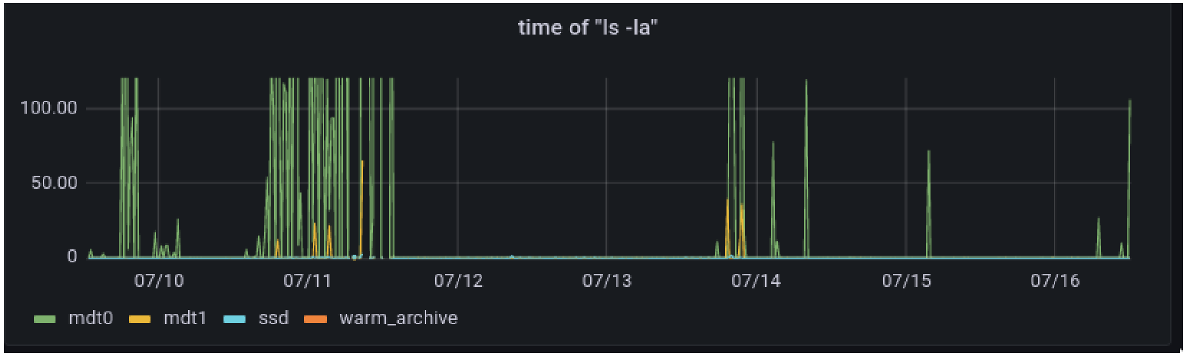


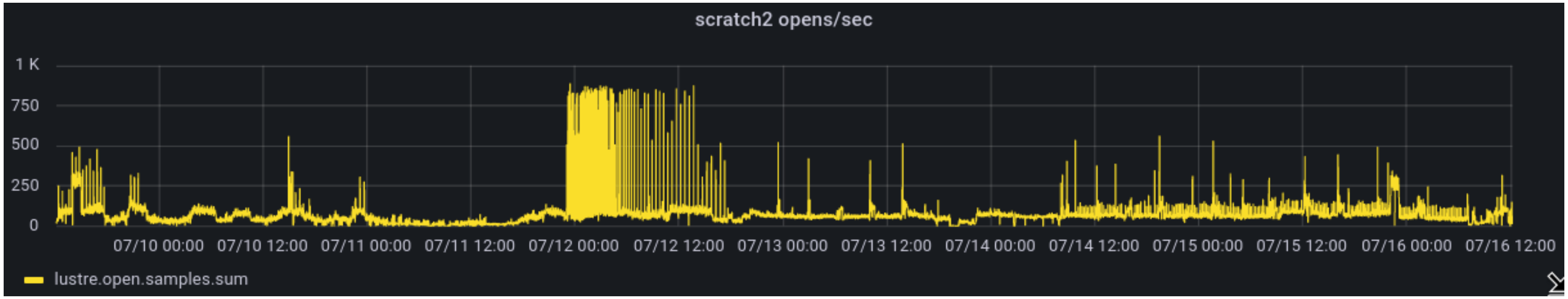




The parallel file system as a shared resource

- All Users / Jobs suffer from a stressed PFS
- E.g. high metadata load from a single or a few users can slow down the PFS for all



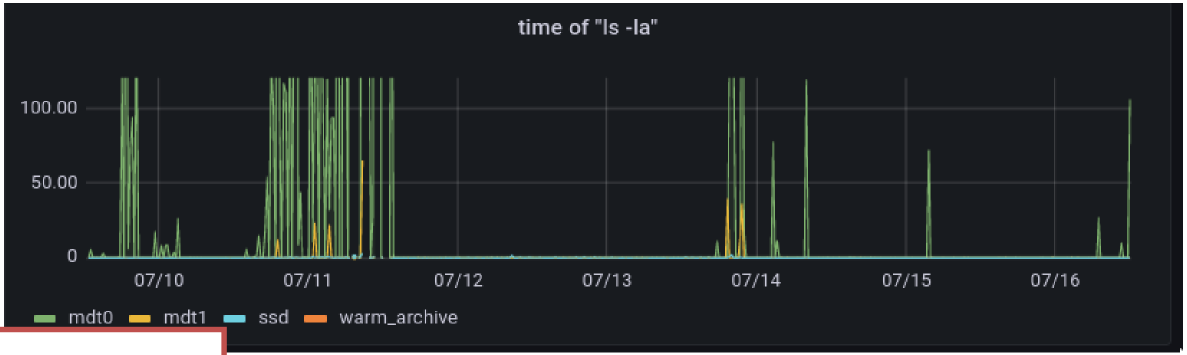


The parallel file system as a shared resource

- All Users / Jobs suffer from a stressed PFS
- E.g. high metadata load from a single or a few users can slow down the PFS for all

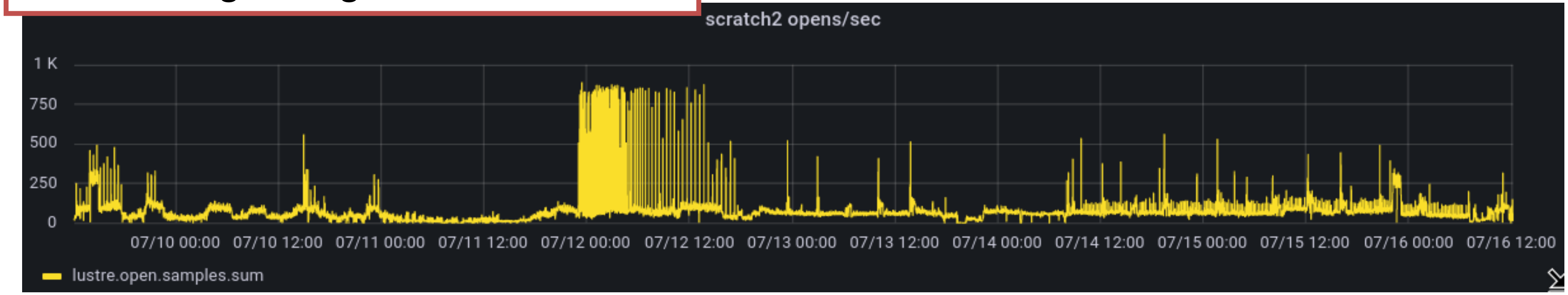
Symptom:

- high `ls –la` latency

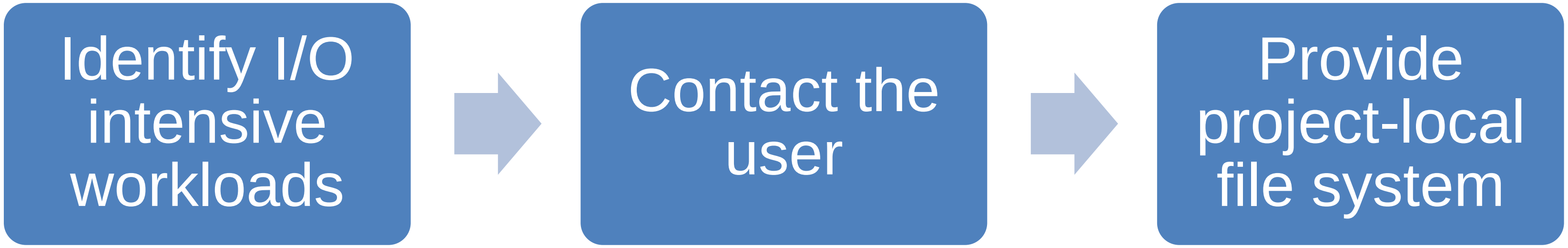


Possible Root-cause:

- Metadata intensive application
- touching a huge amount of files



Basic Idea - Isolate I/O intensive workloads



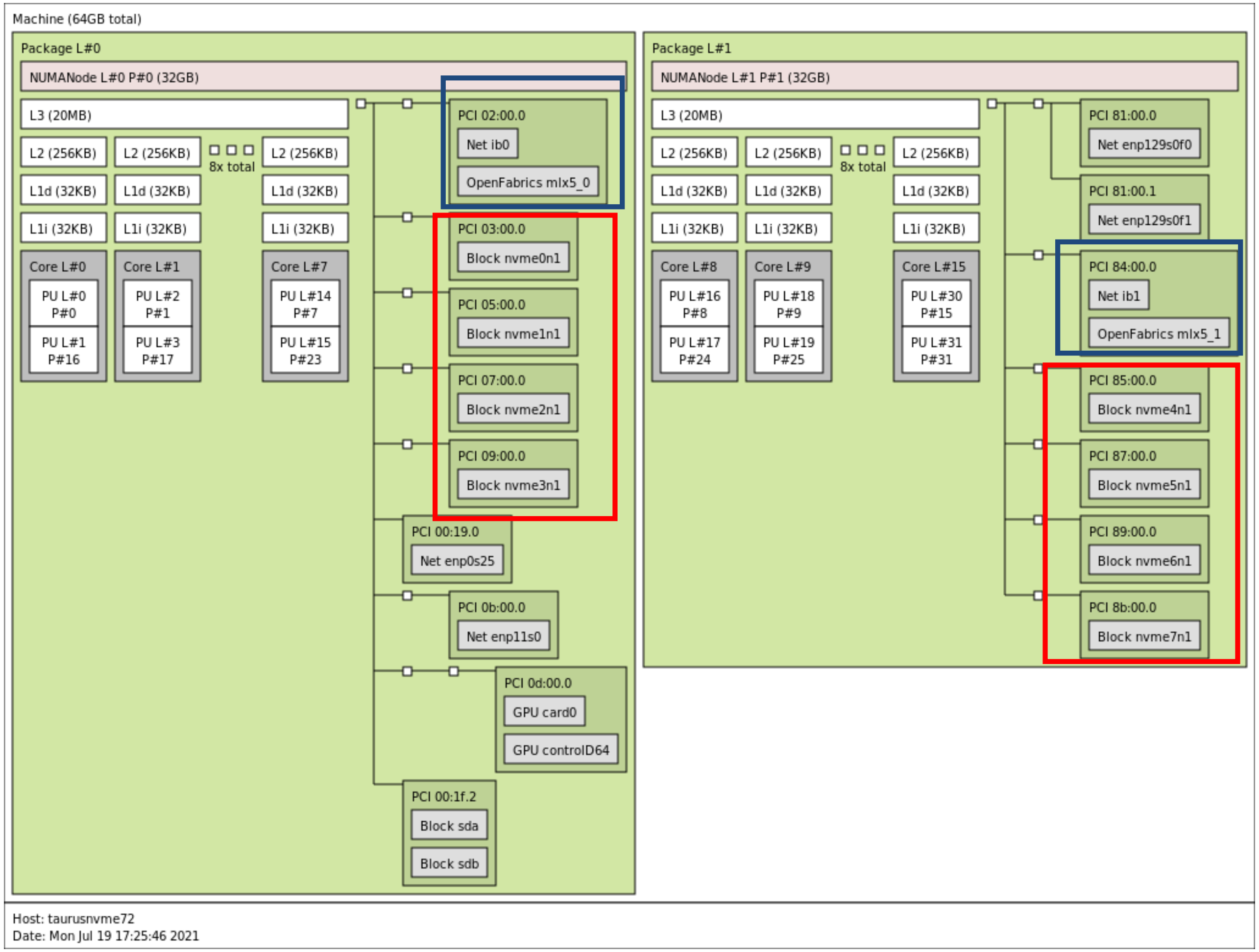
Identify I/O intensive workloads

Contact the user

Provide project-local file system

NVME Nodes – taurusnvme[1-90]

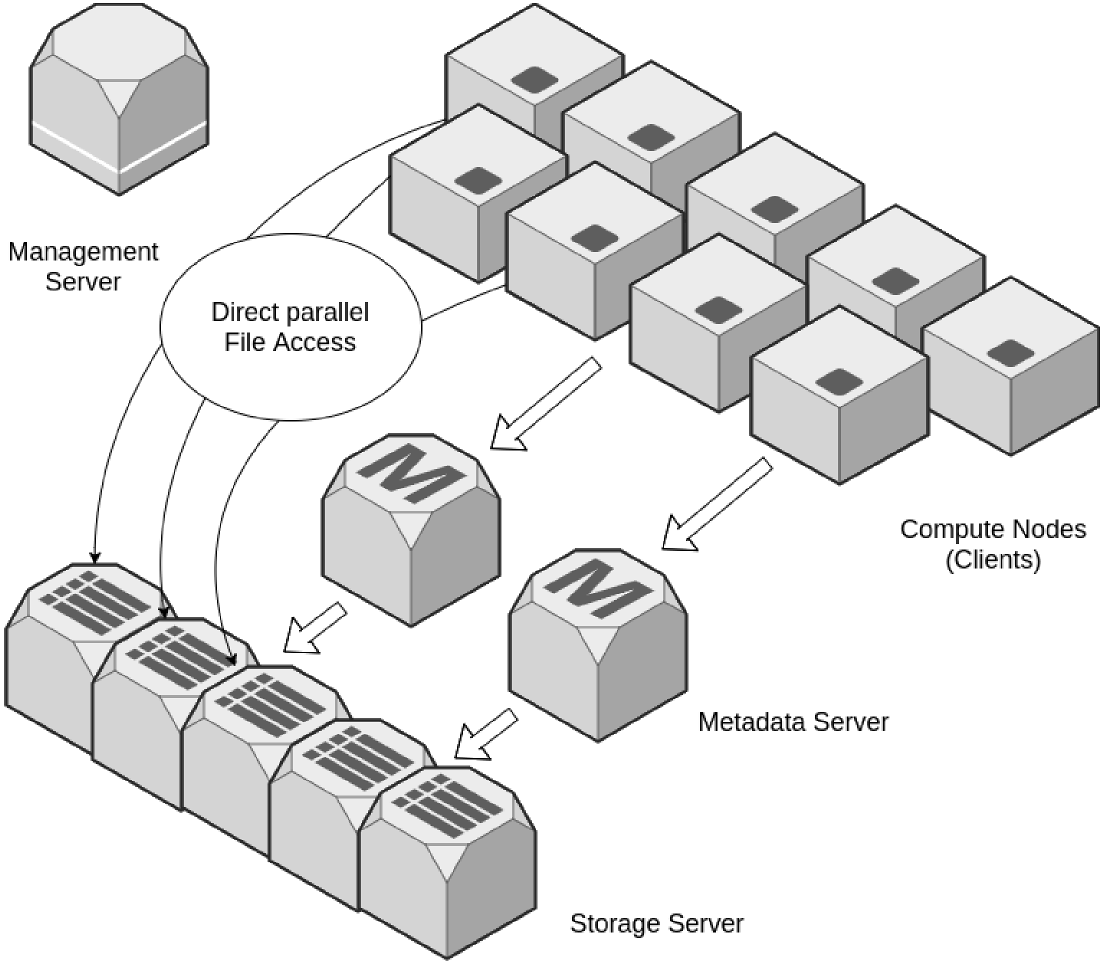
- 90 NVME-Nodes
- 2x EDR Infiniband (100 Gbit/s)
- 8x NVME SSD with 3TB capacity and ~3GiB/s read/write Bandwidth



Host: taurusnvme72
Date: Mon Jul 19 17:25:46 2021

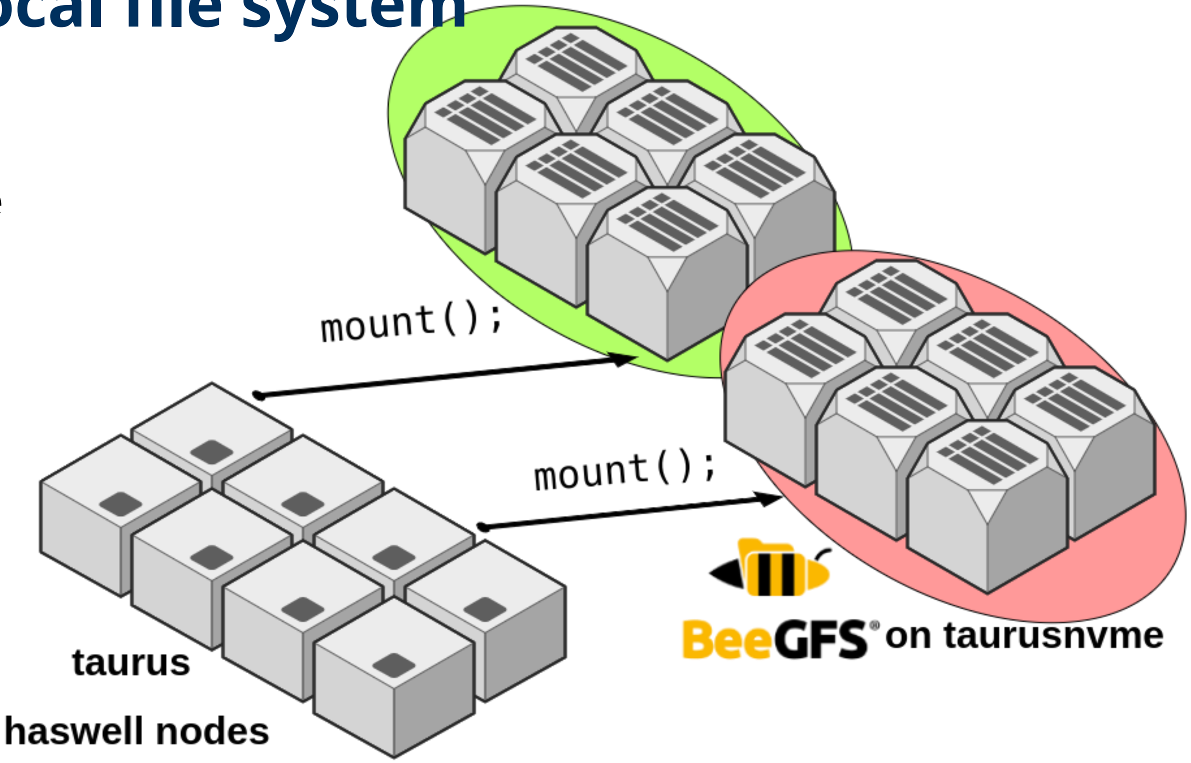
BeeGFS

- Origin from Frauenhofer Institut (FhGFS)
- Since 2014 developed by Thinkparq
- Since last year Peter Braam (CTO)
- Classical parallel file system
- Focusing on performance instead of on features
- Full POSIX-compliant
- Implemented as kernel modules with userspace tools
- “Easy” to deploy



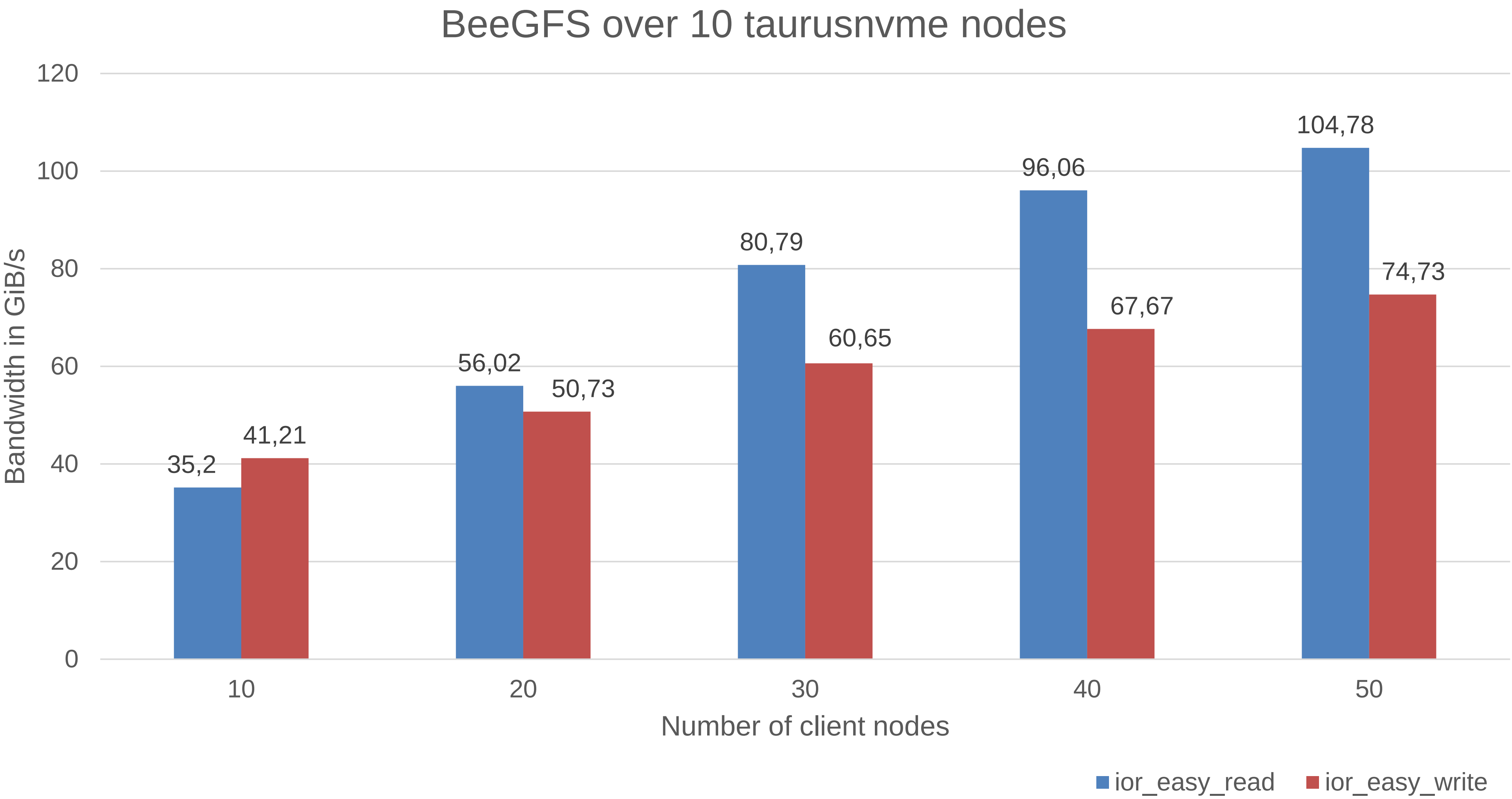
BeeGFS as project-local file system

- Own PFS for I/O intensive projects / users
- File systems of individual sizes for a period of time
- Access restrictions with Unix group rights
- In production at ZIH



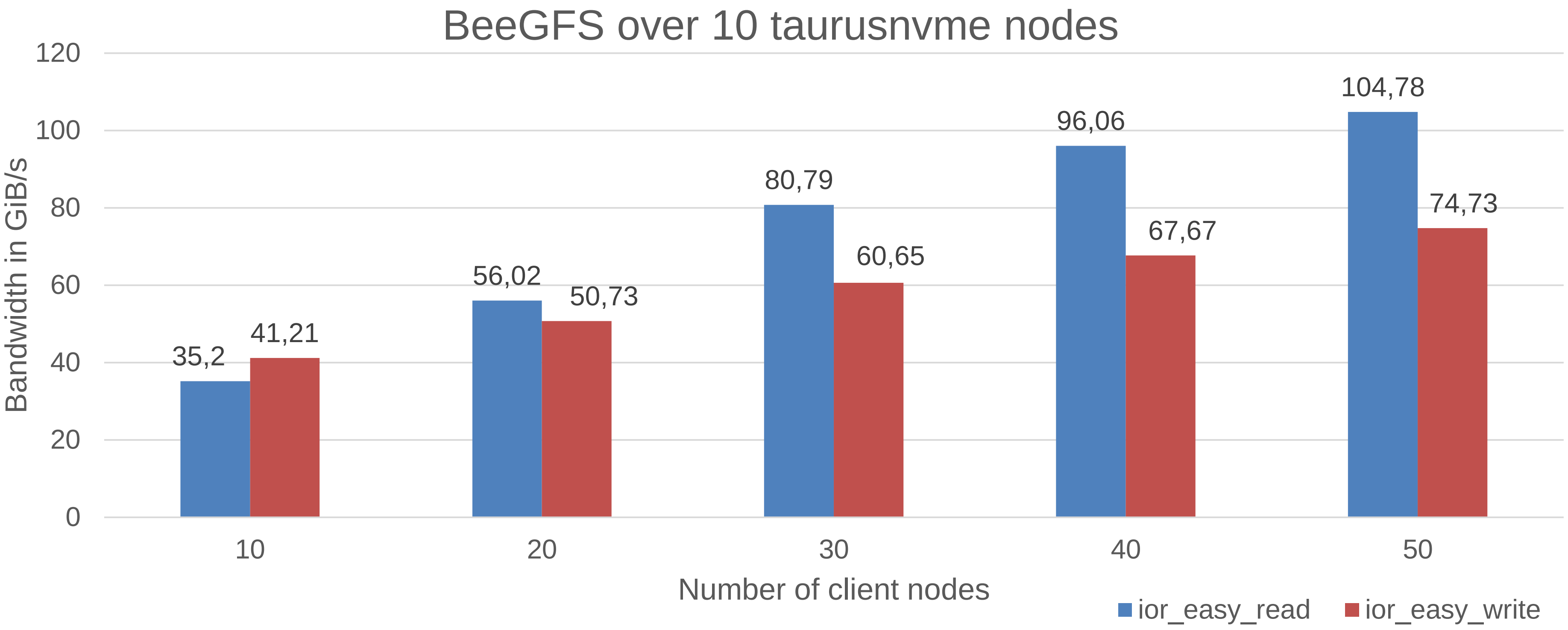
Identify user with high I/O needs → Create file system mount on client nodes → User uses Slurm features for node selection

BeeGFS as parallel file system



BeeGFS as parallel file system

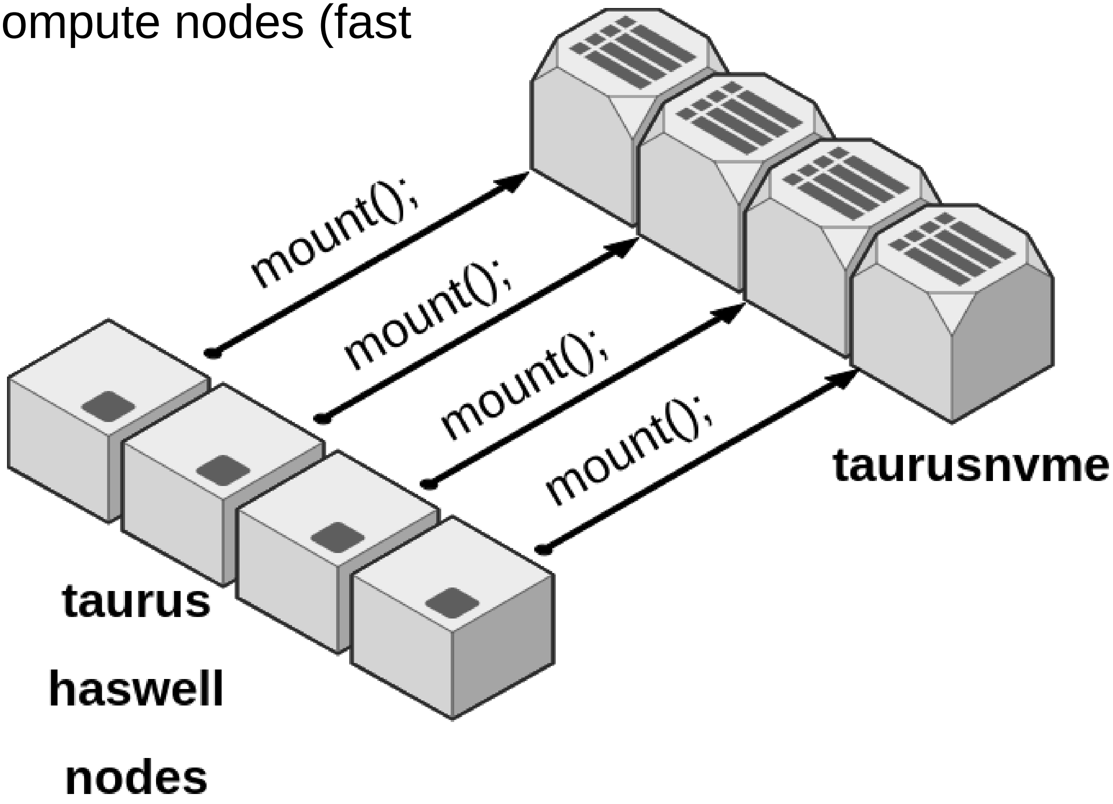
- Aggregated SSD peak performance for 78 storage targets is 234 GiB/s.
- BeeGFS storagebench reports 124 GiB/s.
- BeeGFS storagebench run on storage targets → no Networking!



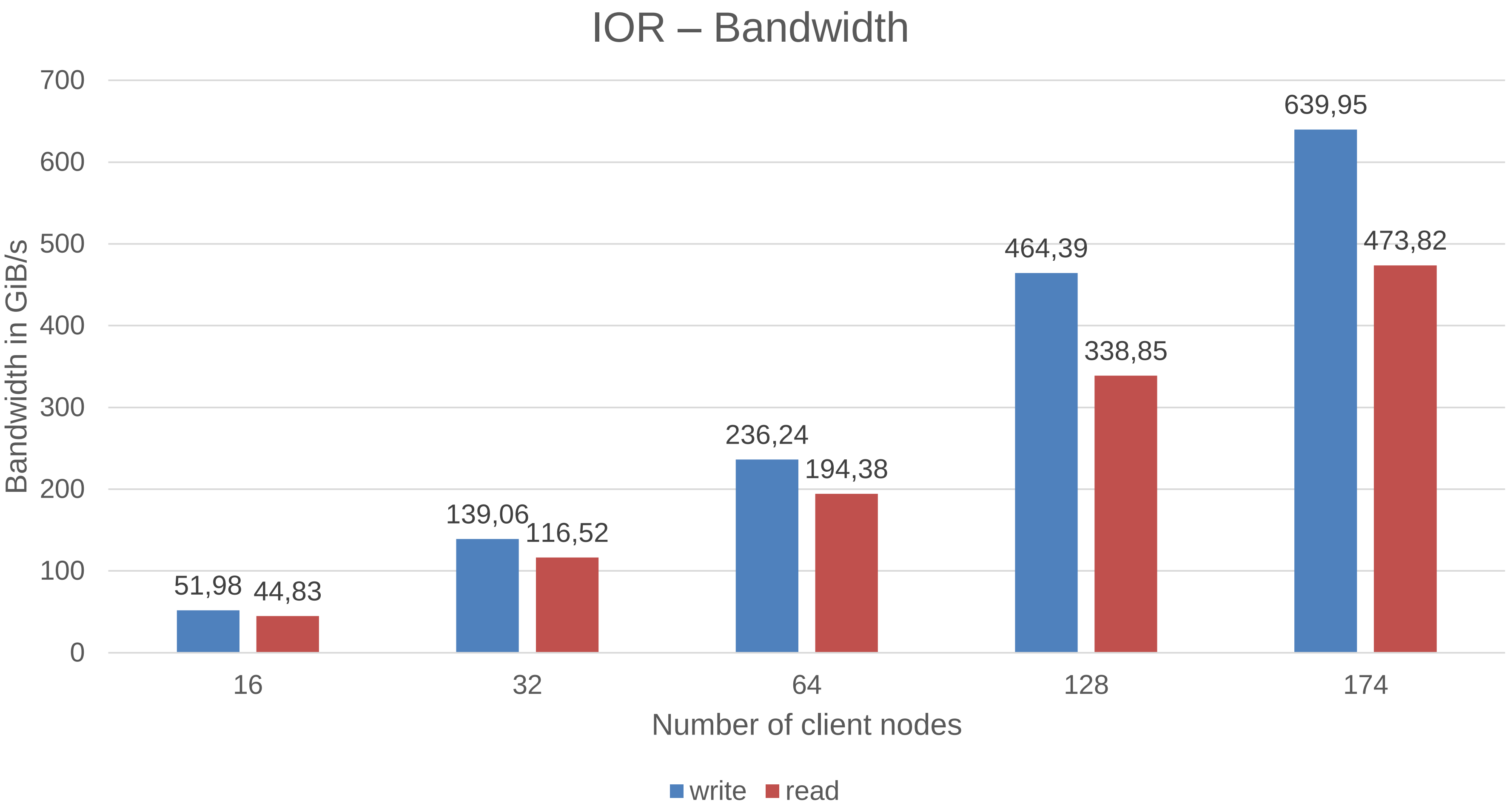
NVME over Fabrics

Compute node connects directly to NVME

- Sever-side SSD appears as block device on compute node
- Use a local file system (ext4, xfs, …)
- No shared view across compute nodes (fast /tmp)
- rw only 1:1 ro also 1:n

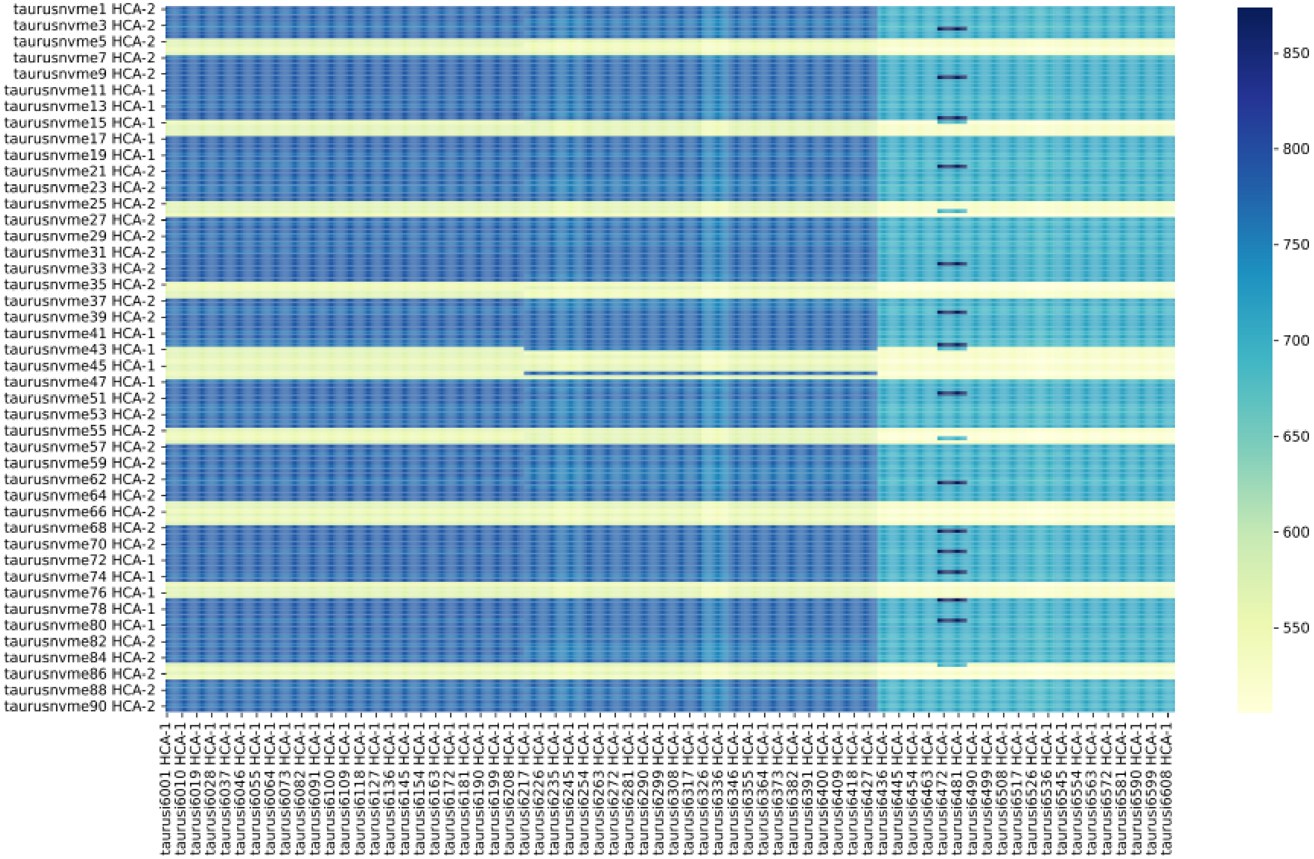


NVME over Fabrics

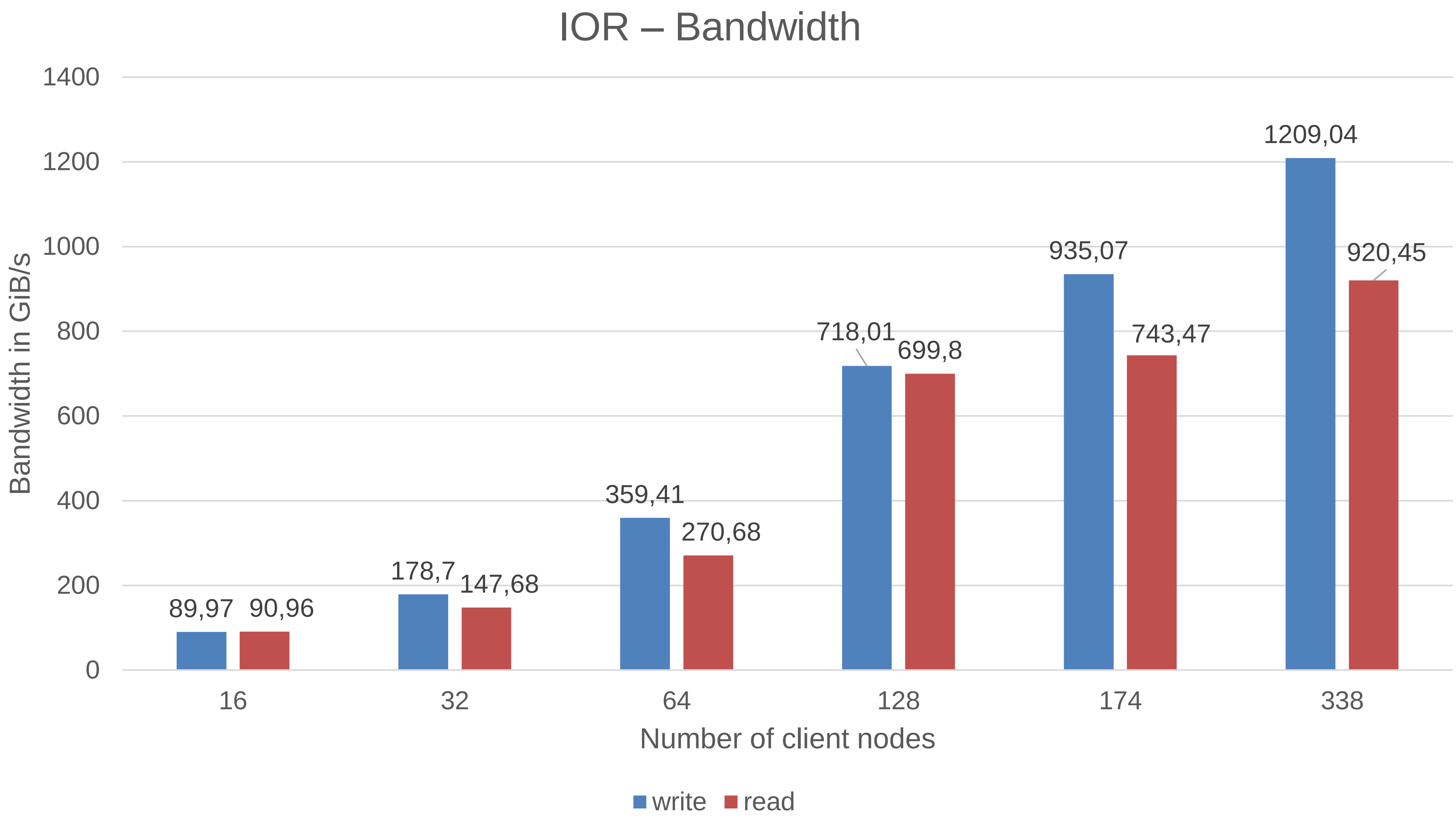


Select your own path!

- Build paths through the network
- Count weight of each hop
- Sum hop-weight for each route
- Select route with lowest weights.

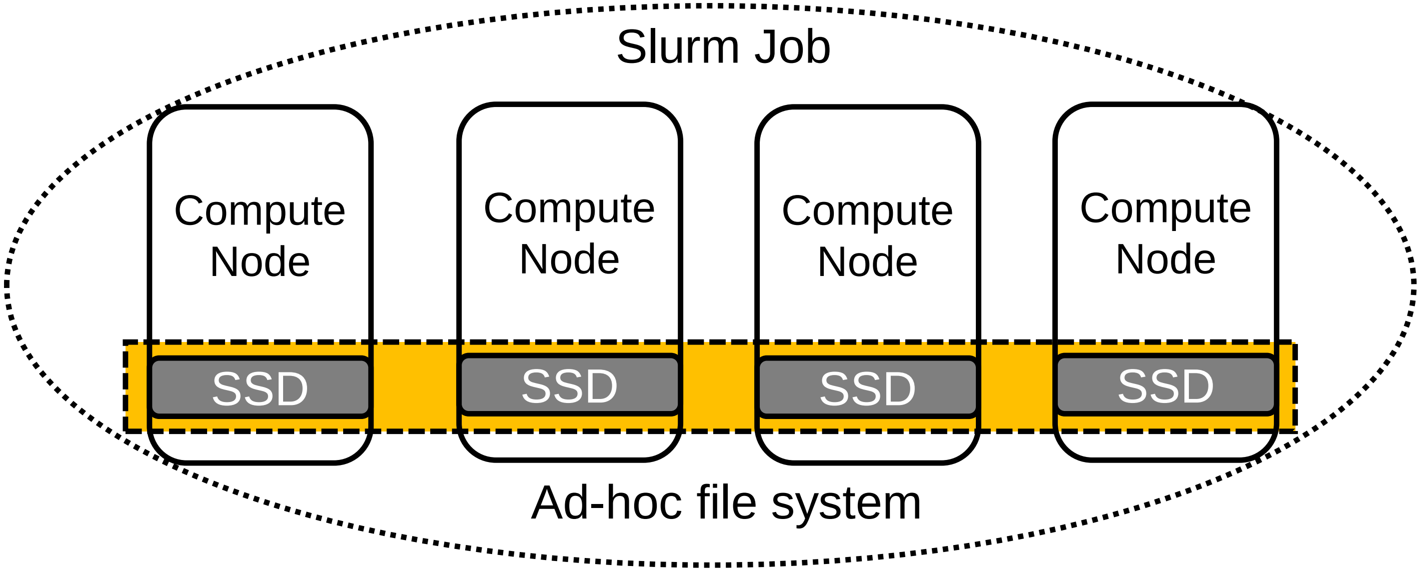


NVME over Fabrics – with hand-picked connections



Ad-hoc file systems for HPC*

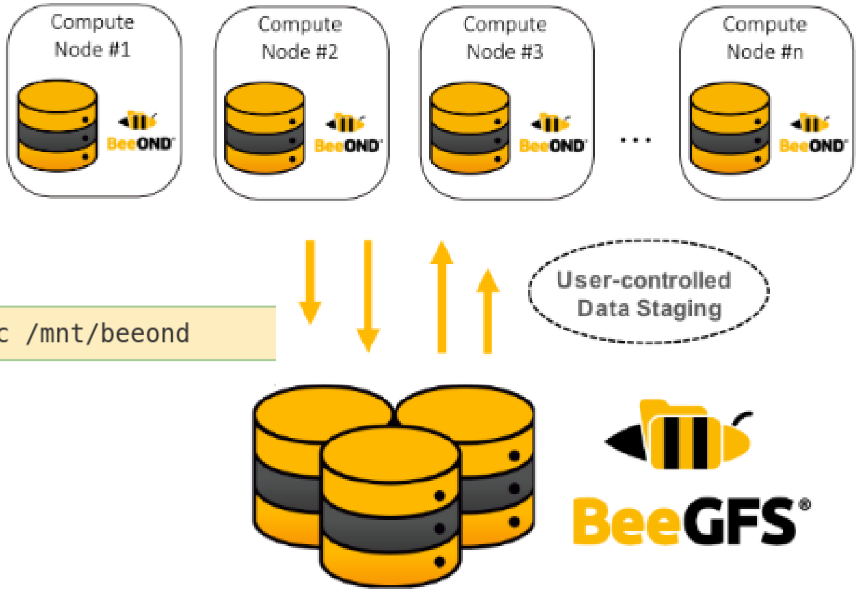
- Isolation of challenging I/O from PFS and the Network
- Using node local fast storages (e.g. SSDs, NVRAM, …)
- Provide a global file system view in a shared namespace
- Job-temporal life time → requires Data Staging



* Brinkmann, André, Mohror, Kathryn, Yu, Weikuan, Carns, Philip, Cortes, Toni, Klasky, Scott A., Miranda, Alberto, Pfreundt, Franz-Josef, Ross, Robert B., and Vef, Marc-André. Ad Hoc File Systems for High-Performance Computing. United States: N. p., 2020. Web. https://doi.org/10.1007/s11390-020-9801-1.

BeeOND – BeeGFS on demand

- BeeGFS as an ad-hoc file system on compute nodes
- Wrapper around BeeGFS
- Run BeeGFS instances on all compute nodes
- Can be built on any underlying POSIX-compliant local file system
- BeeOND clients are implemented as kernel module
- Production ready

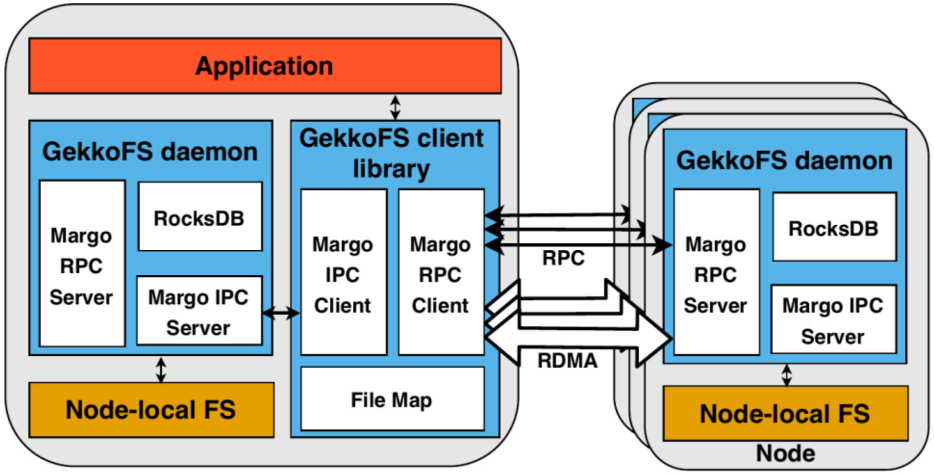


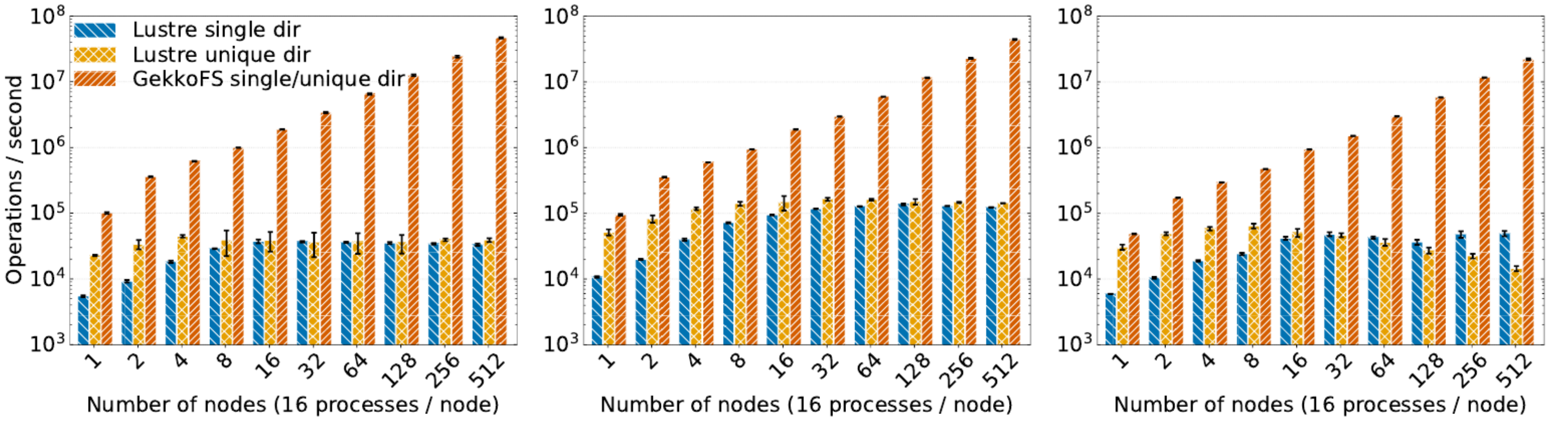
```
$ beeond start -n nodefile -d /data/beeond -c /mnt/beeond
```

https://doc.beegfs.io/latest/_images/beeond-overview.png

GekkoFS

- Developed within ADA-FS DFG Project
- Relaxes POSIX directory semantics
- Distributes Metadata across all nodes
- No locking, no permissions
- 100% in userspace





(a) Create throughput (b) Stat throughput (c) Remove throughput

Vef, MA., Moti, N., Süß, T. et al. GekkoFS — A Temporary Burst Buffer File System for HPC Applications. J. Comput. Sci. Technol. 35, 72–91 (2020). https://doi.org/10.1007

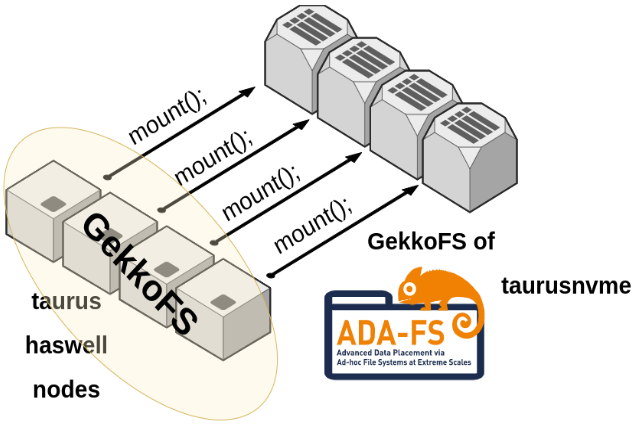
TECHNISCHE UNIVERSITÄT DRESDEN

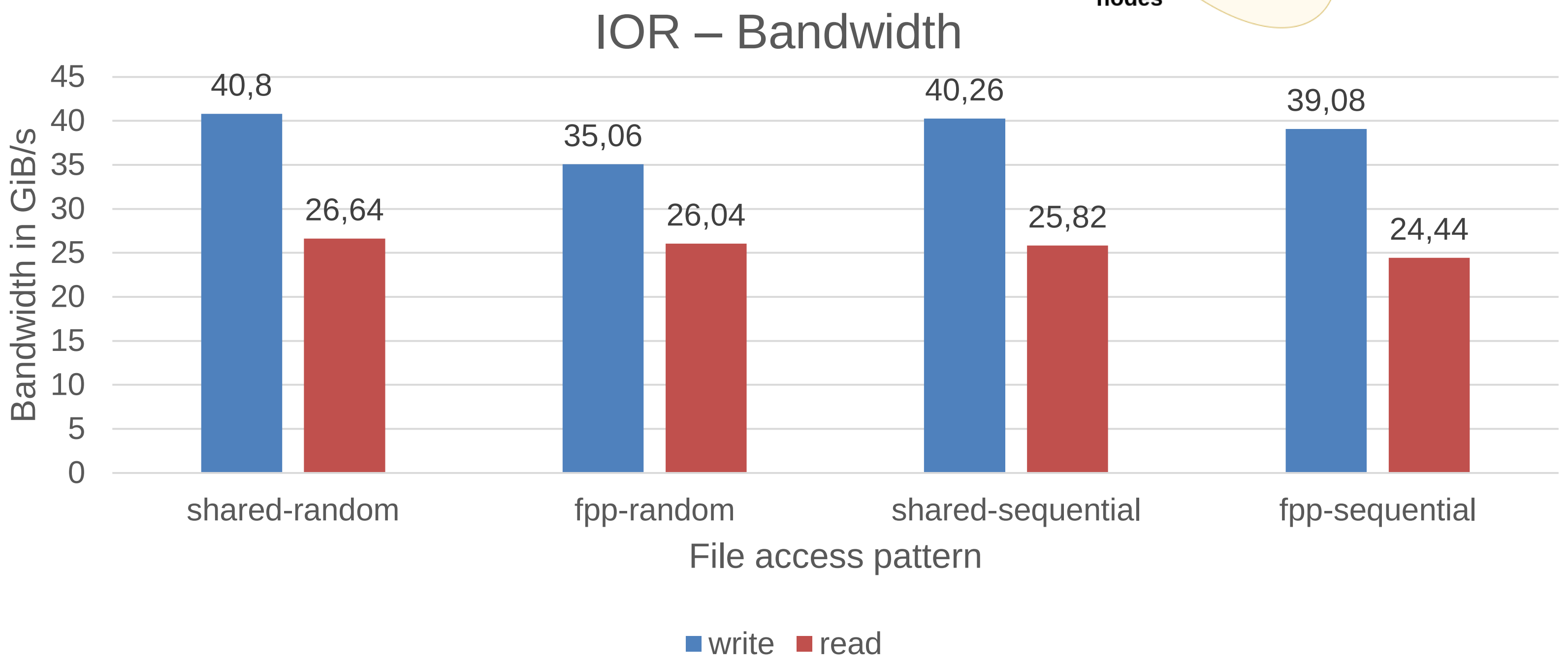
ZIH Center for Information Services & High Performance Computing

DRESDEN concept

GekkoFS Experiment with NVME over Fabrics

- 8 taurus haswell client nodes
- 2 NVMEoF SSD per Client
- In this case, no isolation of network

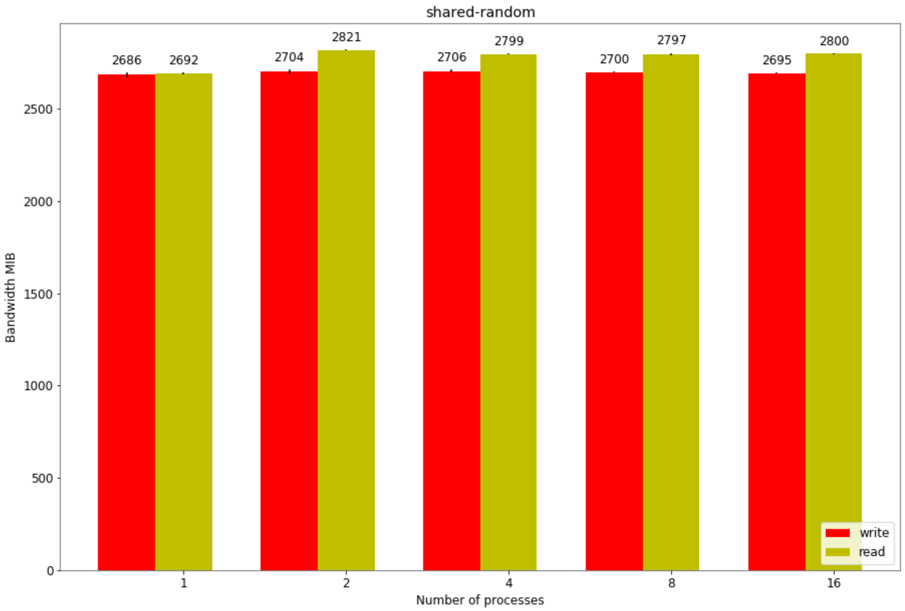


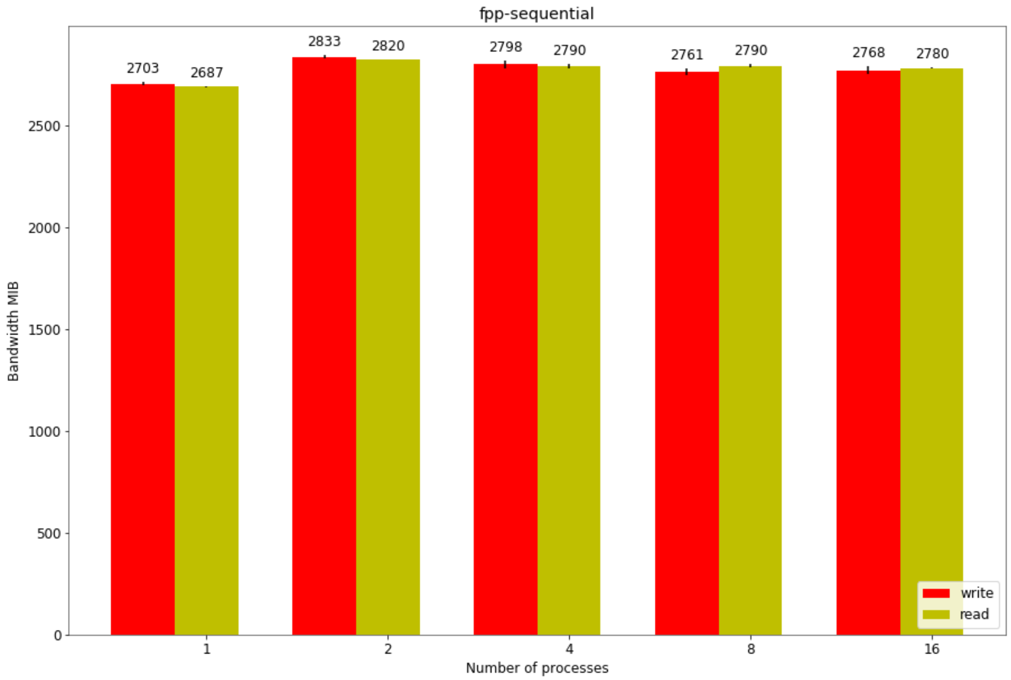


Conclusion

- Every shared ressource can be a bottleneck.
- Providing project-local PFS
  - works with adminitration overhead
  - not reaching peak performance
- Ad-hoc file systems can be an alternative especially for metadata intensive or latency sensitive applications
  - Isolated file system and less network contention
  - Integration in Job-Environment and HPC-Workflows is a todo

Single-NVME SSD’s works well.


shared-random
Bandwidth MIB
Number of processes
🟥 write
🟩 read


fpp-sequential
Bandwidth MIB
Number of processes
🟥 write
🟩 read