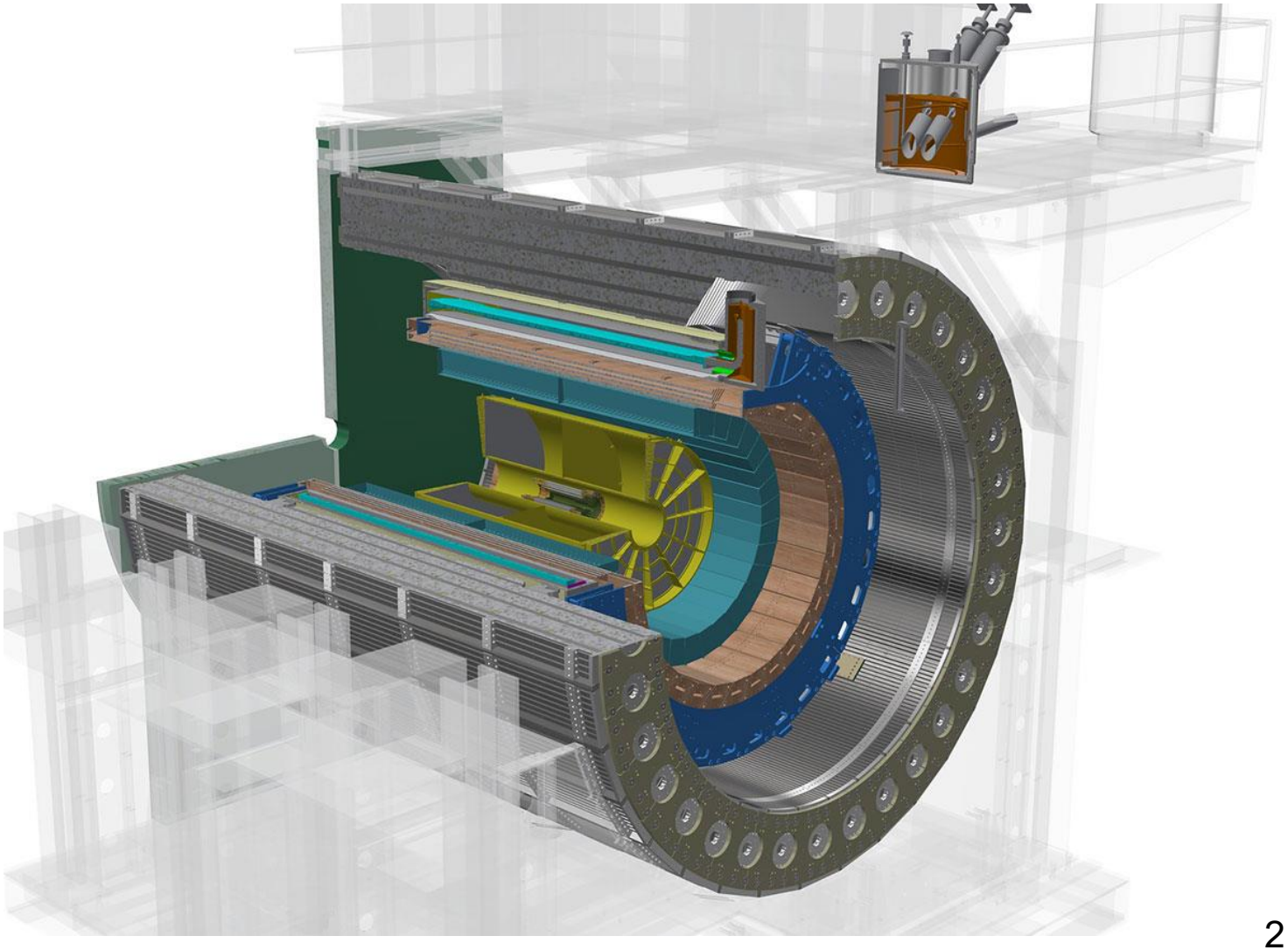
Lustre@sPHENIX

Chris Pinkenburg/BNL

sPHENIX computing

- Second generation experiment at RHIC
- Designed to take data at 15kHz
  - Peak data rates 40GB/s
- Daq is rate limited – same rate for pp and AuAu
  - Extended readout for pp → For this year event sizes larger than AuAu
- No online event building → each event is spread over 60 files
  - Very limited monitoring at the experiment
- First pass data reconstruction with the incoming data
  - Verification with fully reconstructed events
- Second pass during in RHIC off periods



Storage Requirements (2019 computing plan)

- 6 PB (now 11PB) buffer in the sPHENIX counting house
  - On its own Lustre system
- Sustained transfers at 20GB/s to Lustre (and HPSS)

  →Event building reading 20GB/s, writing 30GB/s (intermediate files optimized for random access)

  → Reconstruction reading 30GB/s, writing 10GB/s

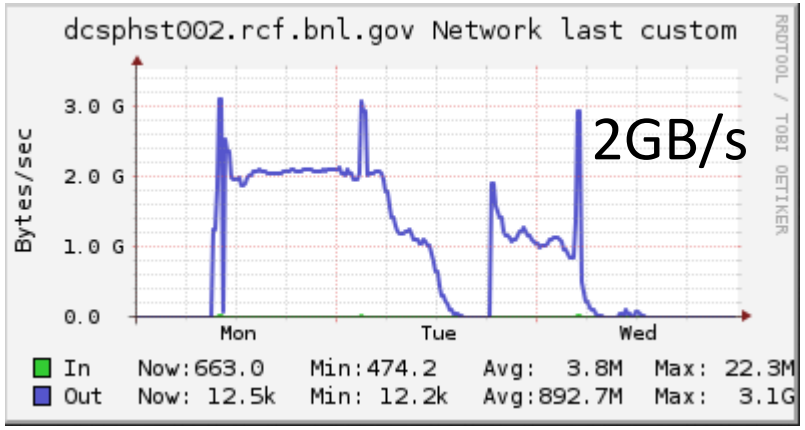
  → Skimming,filtering,… reading 10GB/s

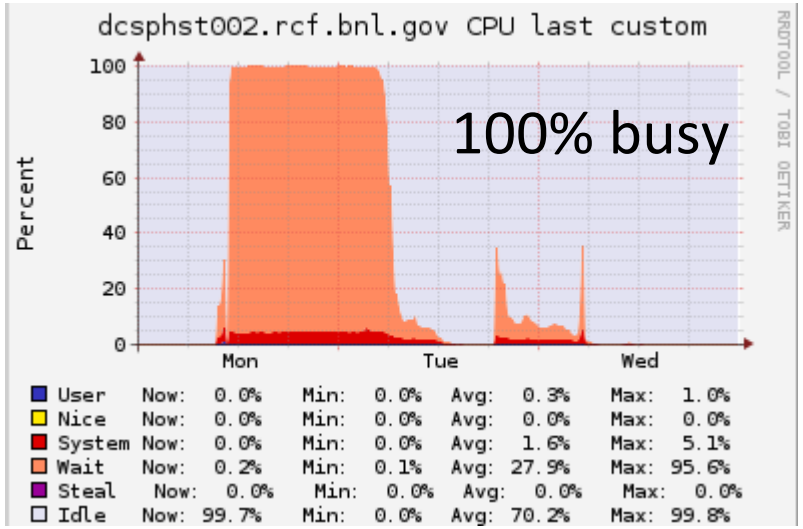
→Grand total of 120GB/s i/o needed for sPHENIX

- Initially 70PB storage (70x1PB fileservers)
- Reconstruction farm with 132k cores

Initial tests (With a lot of help by Tejas Rao+Chris Hollowell)

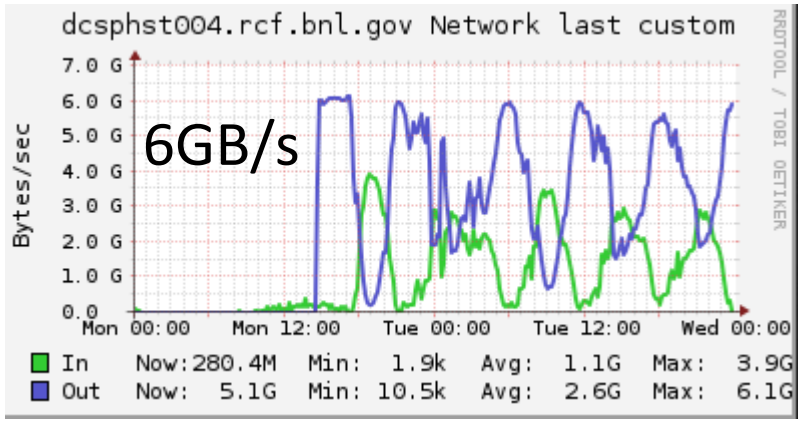
- dCache is the workhorse for PHENIX since 2004, let's go with this.
- Small 5PB 5 server dCache system (no hpss backend)
- Resulting in 2GB/s sustained per server
  - Servers are 100% busy (90% wait)
- 70 servers x 2GB/s = 140GB/s → uncomfortably close

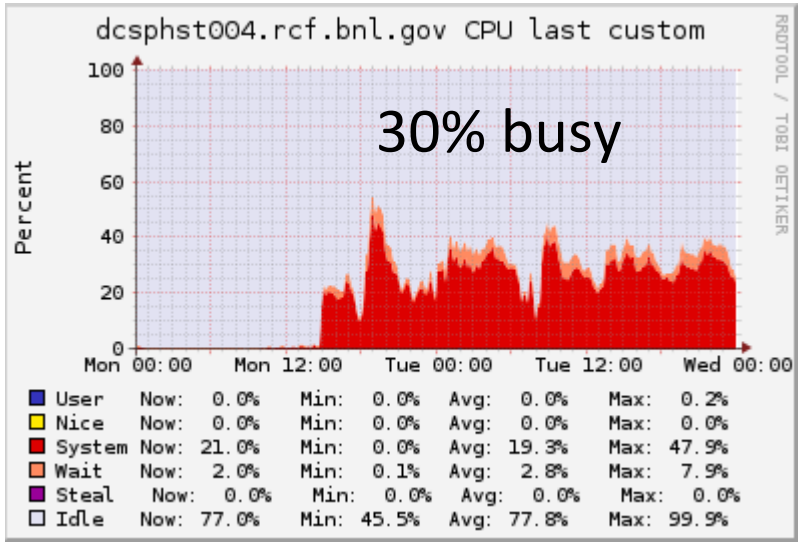




Initial tests (With a lot of help by Tejas Rao+Chris Hollowell)

- Converting the 5PB storage to lustre
- That was an easy decision





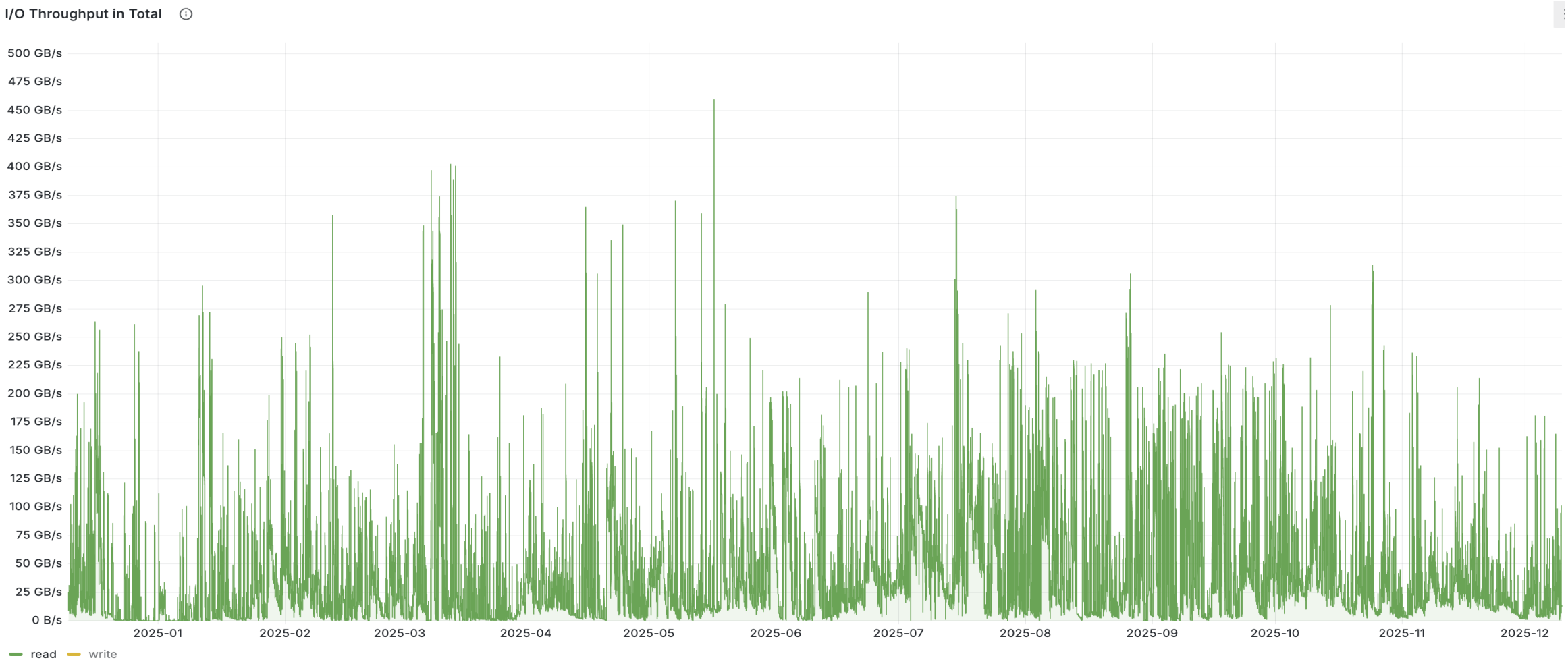
SCDF Lustre Systems

- sPHENIX Prod Lustre (Lustre 2.15.2 + RHEL8.7)
  1 MDS, 88 OSS (ZFS-based OST), 86 PB
- sPHENIX User Lustre(Lustre 2.15.2 + RHEL8.7)
  1 MDS, 5 OSS (ZFS-based OST), 4.8 PB
- STAR/EIC Lustre (Lustre 2.15.4 + RHEL8.10)
  1 MDS, 9 OSS (ZFS-based OST), 7.3 PB
- PHENIX Lustre (Lustre 2.15.4 + RHEL8.10)
  1 MDS (ZFS based MDT), 5 OSS (ZFS-based OST), 5.1 PB
- BNLBOX Lustre(Lustre 2.15.4 + RHEL8.10)
  1 MDS, 3 OSS (MDRAID-based OST), 2.3 PB
- Three NSLS2 Lustre systems
  Instance1: 1MDS, 6 OSS(MDRAID-based OST), 4 PB
  Instance2: 1MDS, 6 OSS(MDRAID-based OST), 4 PB
  Instance3: 1MDS, 12 OSS(MDRAID-based OST), 6.9 PB
- SciServer

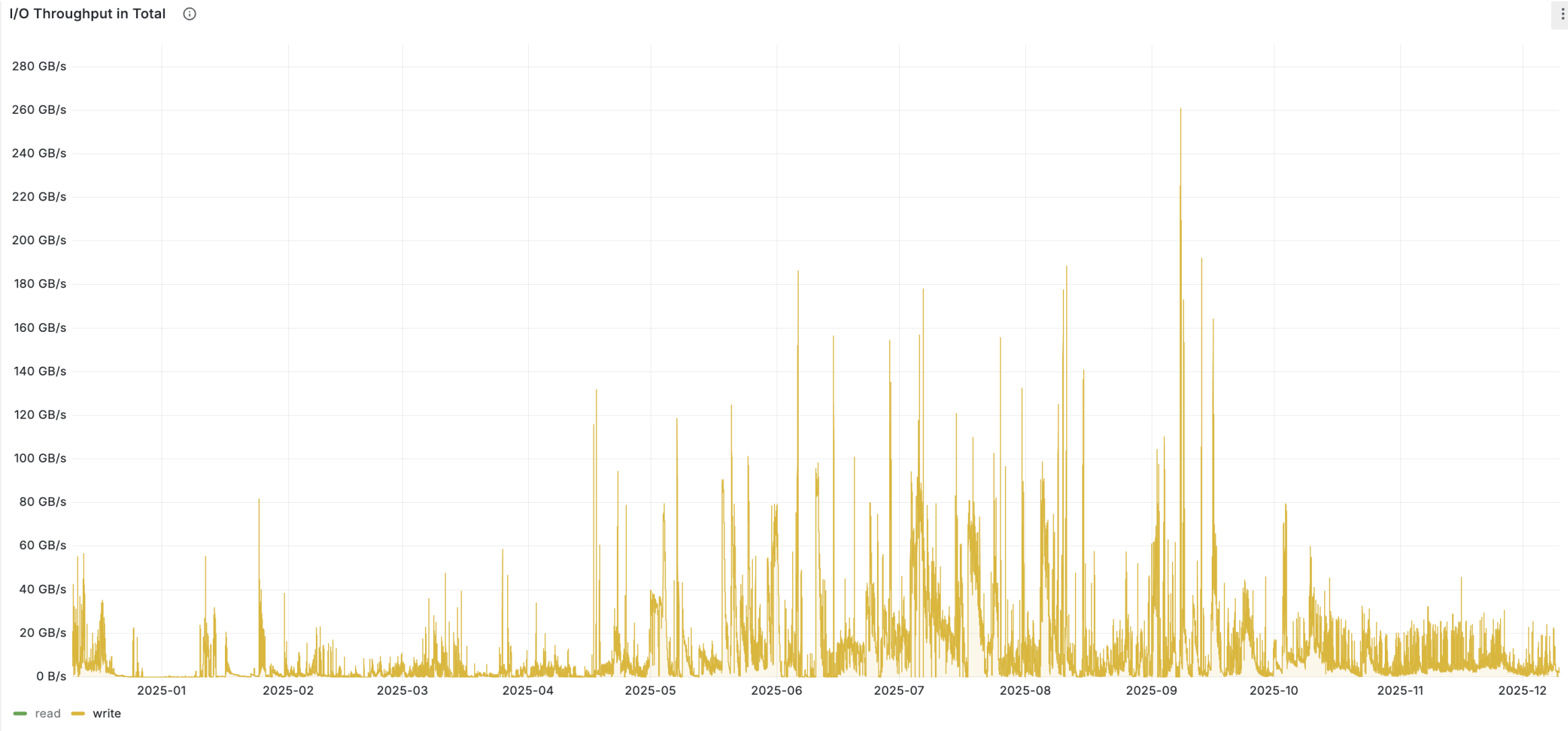
Lustre Monitoring

- Barreleye, Lustre_exporter, Node_exporter, VictoriaLog monitoring
- ZFS Monitoring
  - Node - ZFS Stats
  - ZFS Pool Metrics
  - ZFS-pdf-exporter
- Alarms on:
  - MDT, OST, ZFS health
  - OST occupancy percentage
  - lctl ping Lustre 988 port from a client
  - Minimal number of active request on OST
- Syslog monitoring
  - Client eviction

sPHENIX Lustre read I/O (past 12 months)



sPHENIX Lustre write I/O (past 12 months)



User Experiences

- Lustre scales roughly at 6GB/s per server added
  - I have observed 350GB/s reads over hourlong periods, looks like we did hit 450GB/s
- If a server goes offline – writing continues
  - Zfs resilvering does take its sweet time
  - Processes accessing that server stall until it is back
  - Removing a large number of files is just bound to get hit by this
- In a more real time environment (sPHENIX counting house) we see around 3.5GB/s writes per server
  - When pushing higher we do see lustre pushing back leading to instabilities in the daq
  - It takes real effort in optimizing and load leveling in that environment
  - Heavily loaded lustre does burn through disks (we have about 1 disk failure per week in 1008), similar problem with our older 5PB lustre instance

Not so good user experiences

- Unlike dCache – no hpss backend, we had to develop a system to manage the staging/removing of files
- Initial deployment in 2023 suffered under a bug in mdraid which rendered the space unusable
  - Heroic effort by SCDF staff took care of that (but it was a cliff hanger)
- A bug in Lustre caused hanging servers (2024 or 2025?)
  - Costin found the patch and installed it
- Since then it's been stable

And one worry

- There does not seem to be an fsck type utility, if lustre looses its brain it cannot be recovered from the storage nodes
  - Backups prevent complete loss but at 100's of GB/s its content changes quickly

One more

- We had a period during this year where data got corrupted during copy
  - It looked like short reads/writes (not all data read/written)
  - We didn’t have the time to trace this down then during data taking
- We reduced the overall load
  - lowered the number of high i/o jobs by a factor of 3
  - replaced cp by dd with bs=12MB which reduced the load on zfs
  - Use dd to local disk instead of reading/writing directly to lustre
  - Doing md5sum checks for most transfers
- Not seen again since doing this
- Just to mention this here – TTrees as used by sPHENIX pretty much disable any caching

Summary

- Lustre's capabilities made the sPHENIX program as is possible (daq and reconstruction)
  - Where the transfers from our high speed daq are a small blip on top of what our data processing does
- Overall, a very good experience, we don't spend any time worrying about i/o anymore.
- But it really takes effort to keep this running which is highly appreciated
- Based on this PHENIX decided to switch from dCache to Lustre