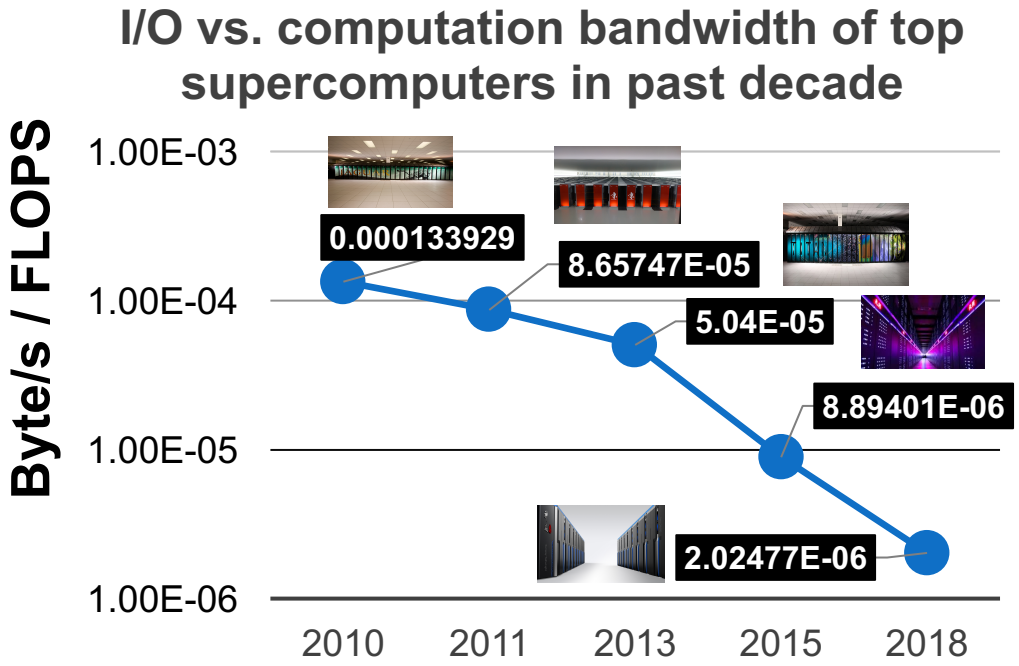
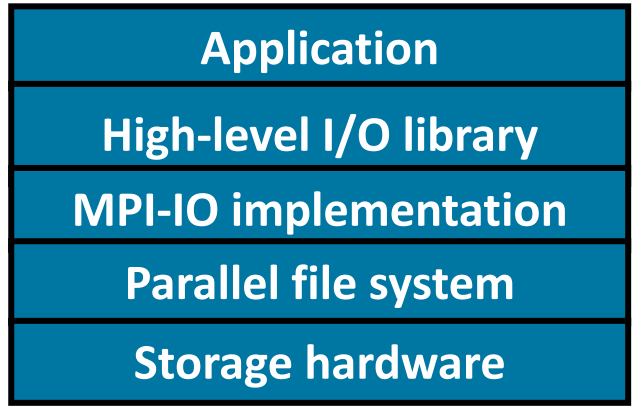
End-to-end I/O Monitoring on a Leading Supercomputer

Bin Yang, Xu Ji, **Xiaosong Ma**, Xiyang Wang,
Tianyu Zhang, Xiupeng Zhu, Nosayba El-Sayed,
Yibo Yang, Haidong Lan, Jidong Zhai, Weiguo Liu, Wei Xue

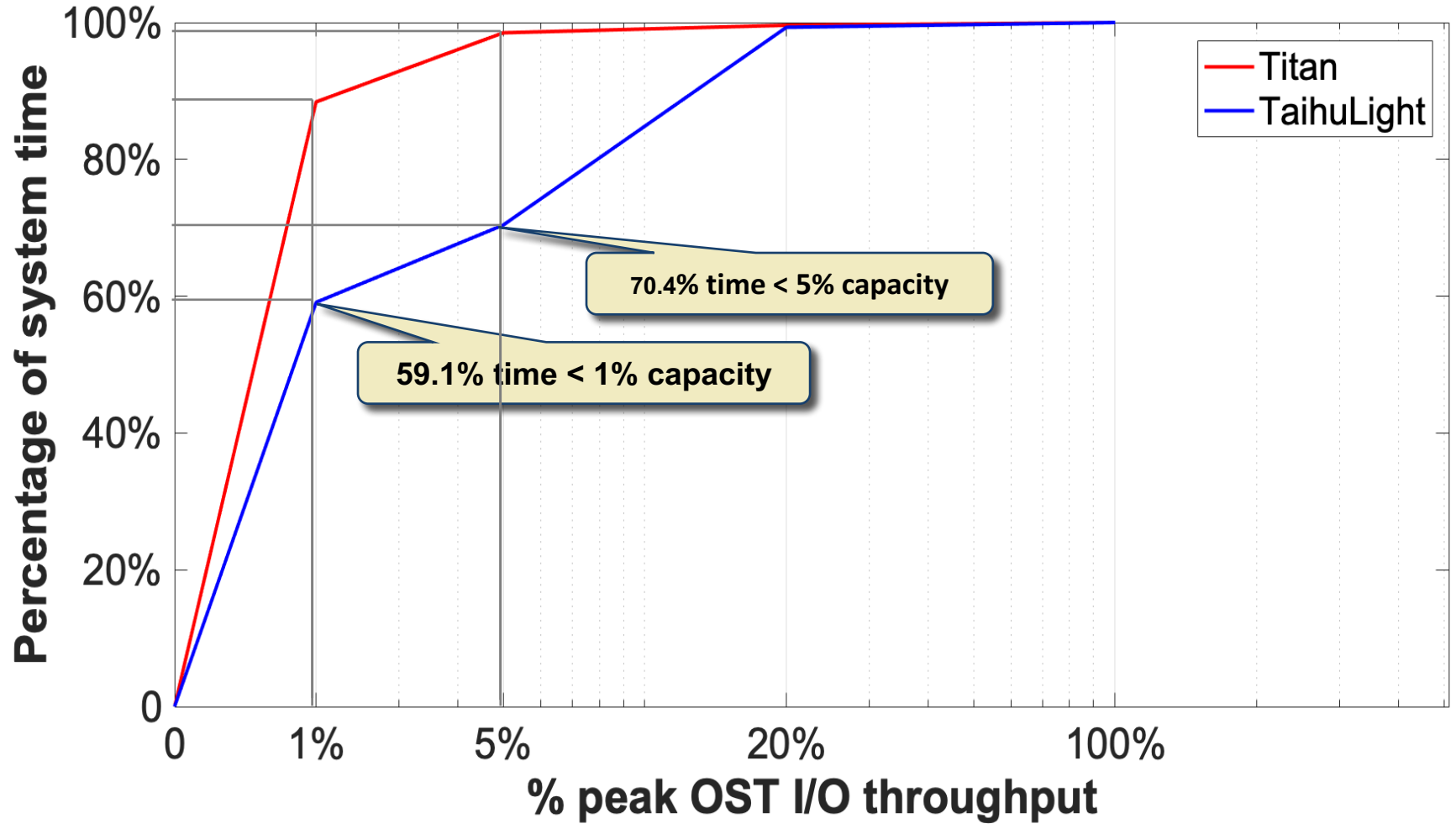
Motivation: Large Supercomputer's I/O Problems

- Supercomputers useful through data
- I/O lags behind computation
  - Secondary storage slower than processors/memory
  - Long I/O path
  - Less scalable or consistent, shared resources among many applications
- Applications tightly control time spent on I/O





Motivation: I/O System Not Well Utilized



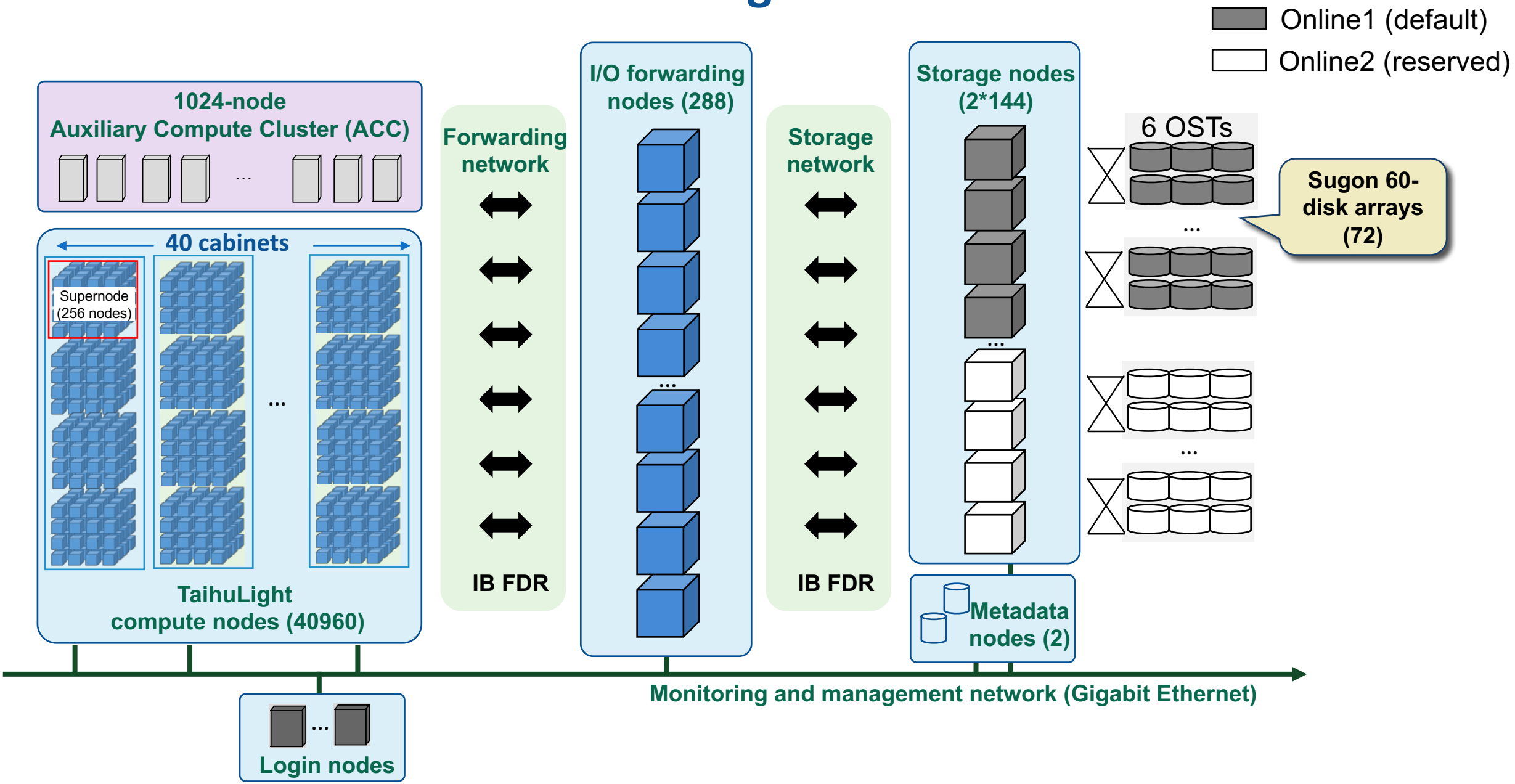
Performance bottleneck, contention point, AND system under-utilization!

Beacon: End-to-end Supercomputer I/O Monitoring

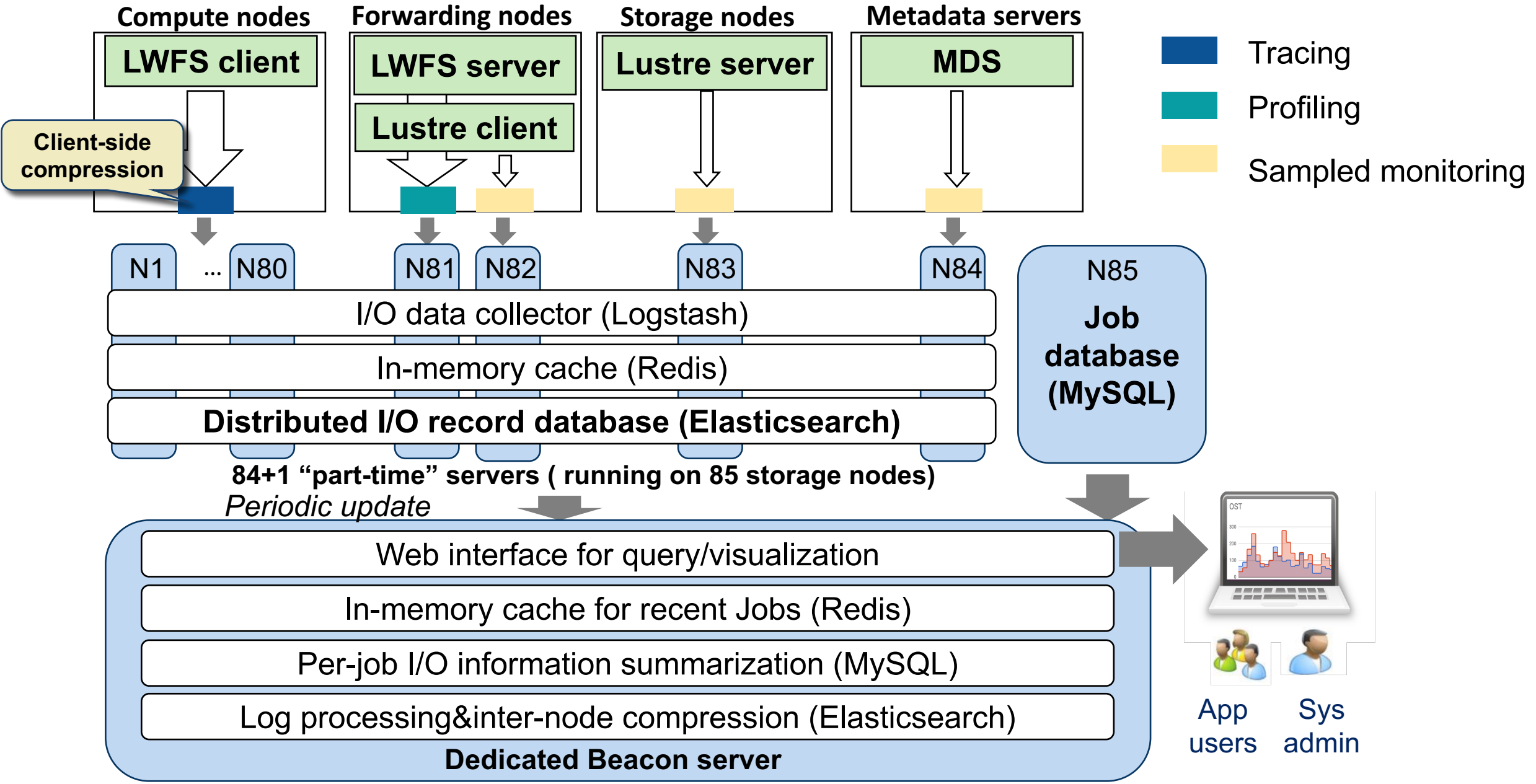
- Understand HPC I/O for designing future systems/applications
  - Lightweight end-to-end I/O resource monitoring
    - Collects per-application-level traces and statistics
    - Correlates monitoring data from multiple system levels
    - Performance diagnosis, application I/O profiling, anomaly detection

- Deployed at TaihuLight, world's No.3 supercomputer
  - Production use since April 2017
  - No user effort required
  - Identified design/configuration/policy problems
  - Code and monitoring data released

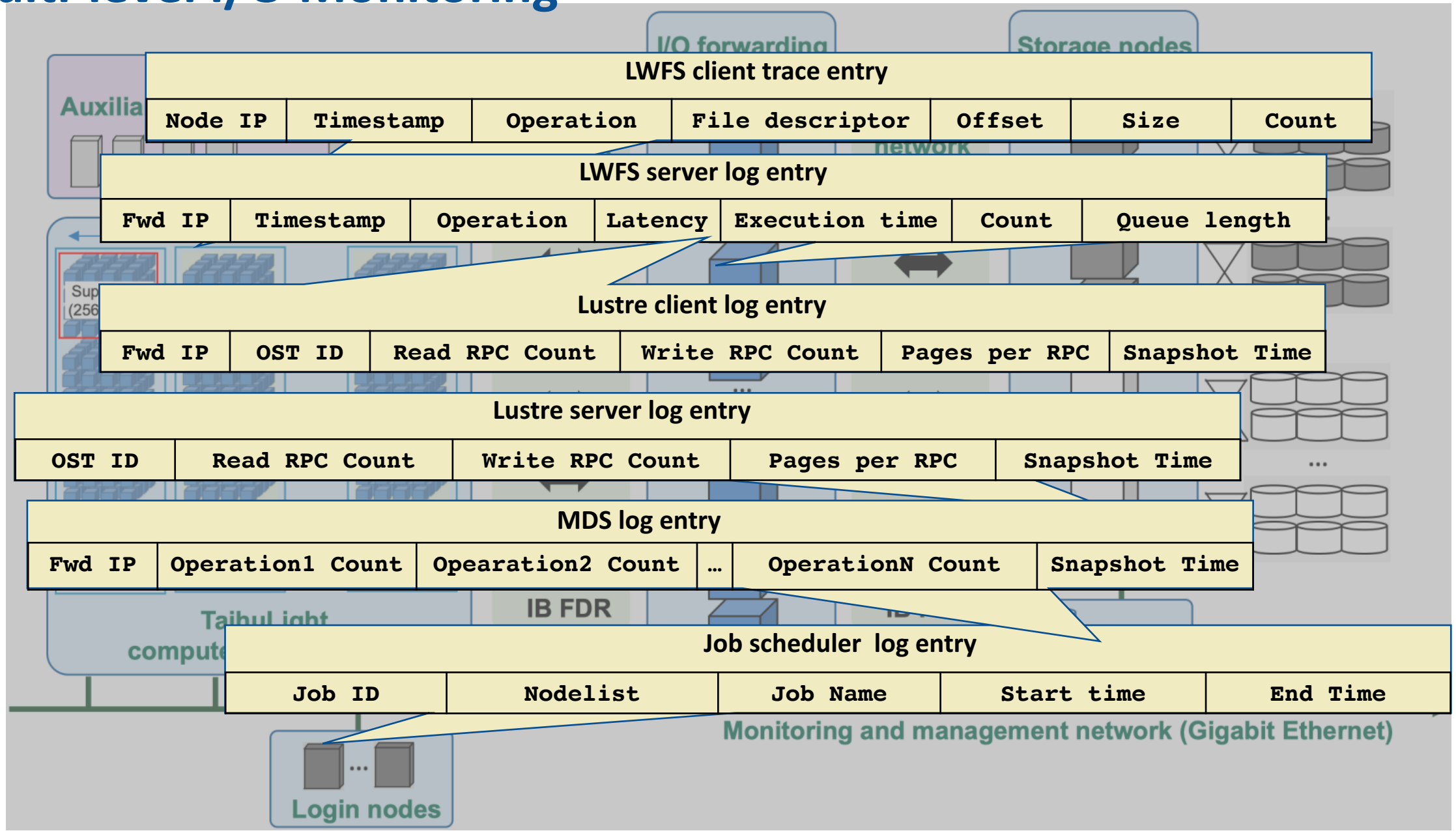
Architecture Overview of TaihuLight



Overview of Beacon



Multi-level I/O Monitoring



LWFS client trace entry

Node IP	Timestamp	Operation	File descriptor	Offset	Size	Count

LWFS server log entry

Fwd IP	Timestamp	Operation	Latency	Execution time	Count	Queue length

Lustre client log entry

Fwd IP	OST ID	Read RPC Count	Write RPC Count	Pages per RPC	Snapshot Time

Lustre server log entry

OST ID	Read RPC Count	Write RPC Count	Pages per RPC	Snapshot Time

MDS log entry

Fwd IP	Operation1 Count	Opearation2 Count	…	OperationN Count	Snapshot Time

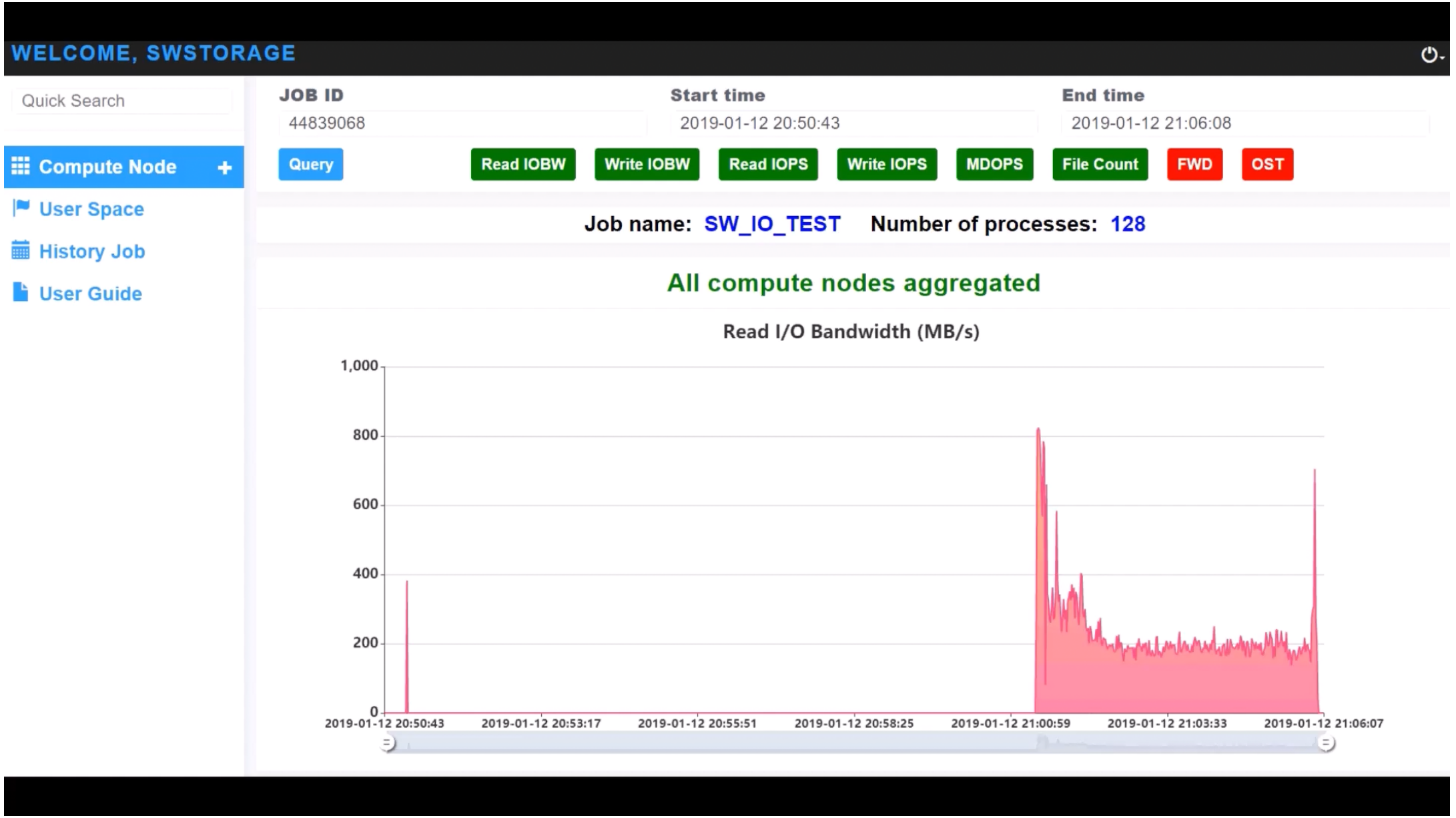
Job scheduler log entry

Job ID	Nodelist	Job Name	Start time	End Time

Cross-layer I/O Profiling Data Analysis

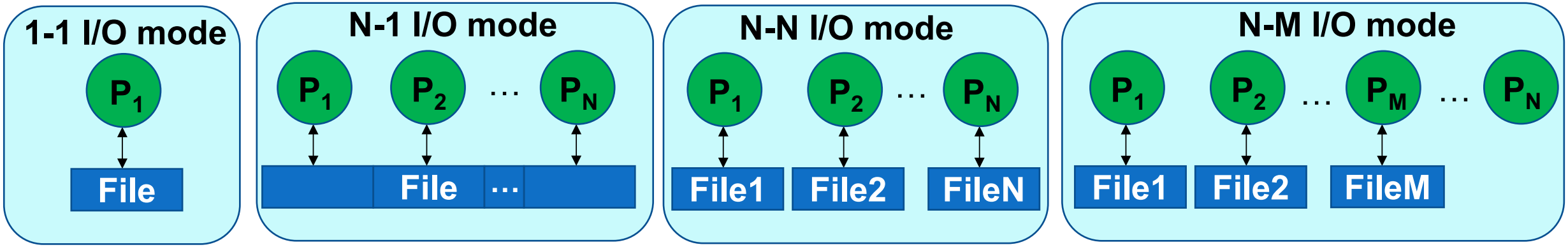
- Per-job I/O performance analysis
  - Job to compute nodes: batch scheduler history lookup
  - Compute to forwarding nodes: per-job mapping info lookup
  - Forwarding to storage nodes: file system lookup commands
- I/O subsystem monitoring for administrators
  - Visualization and query services
  - Any node, any time
- Automatic anomaly detection

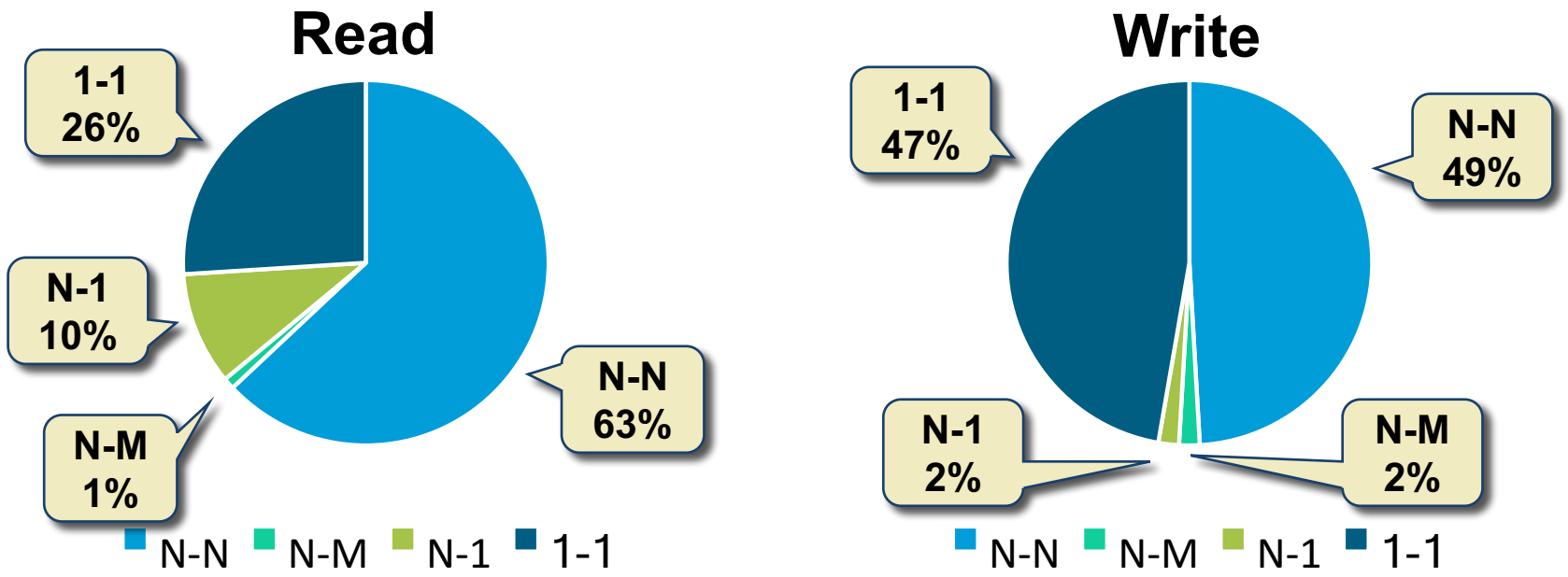
Beacon Demo


WELCOME, SWSTORAGE
Quick Search
Compute Node
User Space
History Job
User Guide
JOB ID
44839068
Start time
2019-01-12 20:50:43
End time
2019-01-12 21:06:08
Query
Read IOBW
Write IOBW
Read IOPS
Write IOPS
MDOPS
File Count
FWD
OST
Job name: SW_IO_TEST
Number of processes: 128
All compute nodes aggregated
Read I/O Bandwidth (MB/s)
1,000
800
600
400
200
0
2019-01-12 20:50:43
2019-01-12 20:53:17
2019-01-12 20:55:51
2019-01-12 20:58:25
2019-01-12 21:00:59
2019-01-12 21:03:33
2019-01-12 21:06:07

Case Study 1: Application I/O Behavior Analysis

- Based on 18-month data collection on TaihuLight
  - 116,765 jobs using at least 32 compute nodes

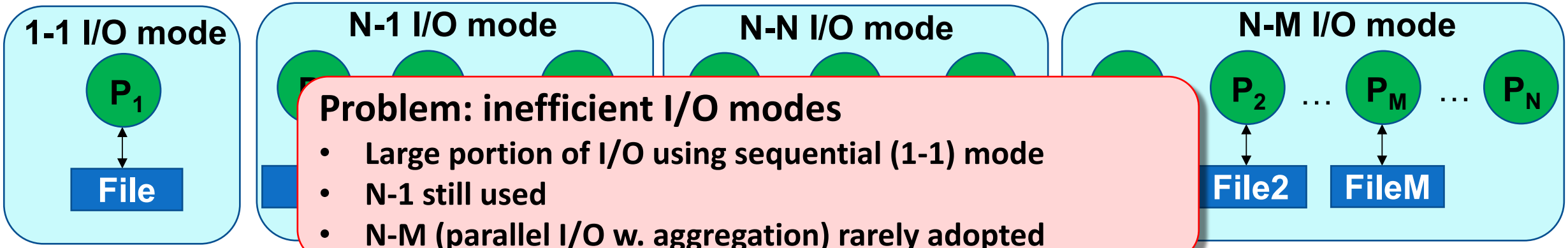




Distribution of file access modes, in access volume

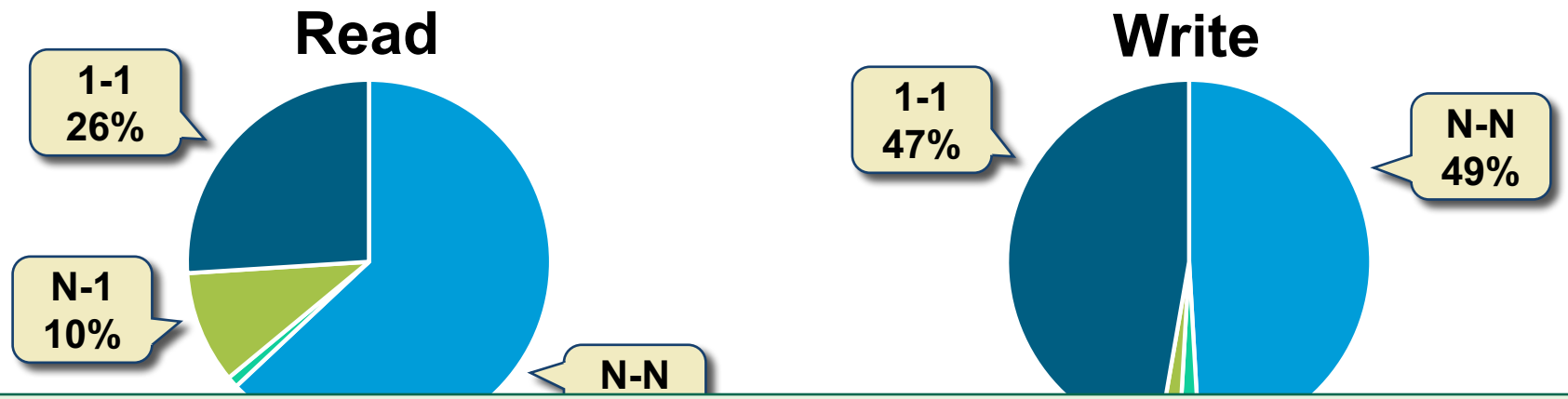
Case Study 1: Application I/O Behavior Analysis

- Based on 18-month data collection on TaihuLight
  - 116,765 jobs using at least 32 compute nodes



Problem: inefficient I/O modes

- Large portion of I/O using sequential (1-1) mode
- N-1 still used
- N-M (parallel I/O w. aggregation) rarely adopted



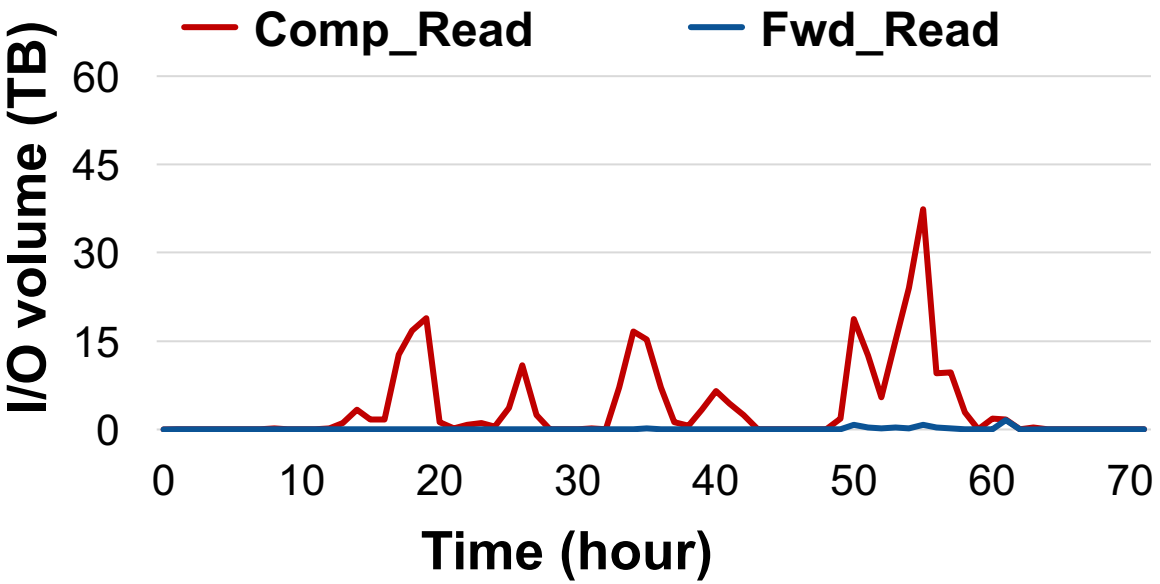
Solution: application/user behavior correction

- Targeted suggestions to large-scale, I/O-intensive applications

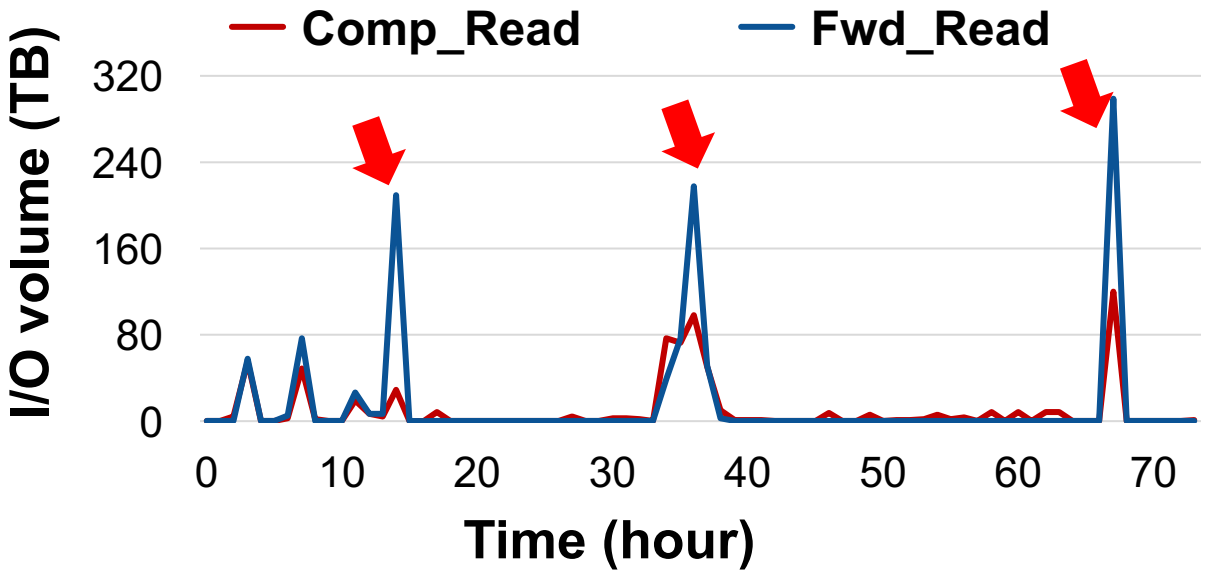
Distribution of file access modes, in access volume

Case Study 2: Cross-layer I/O Volume Comparison

Sample read volume history (normal)



Sample read volume history (thrashing)



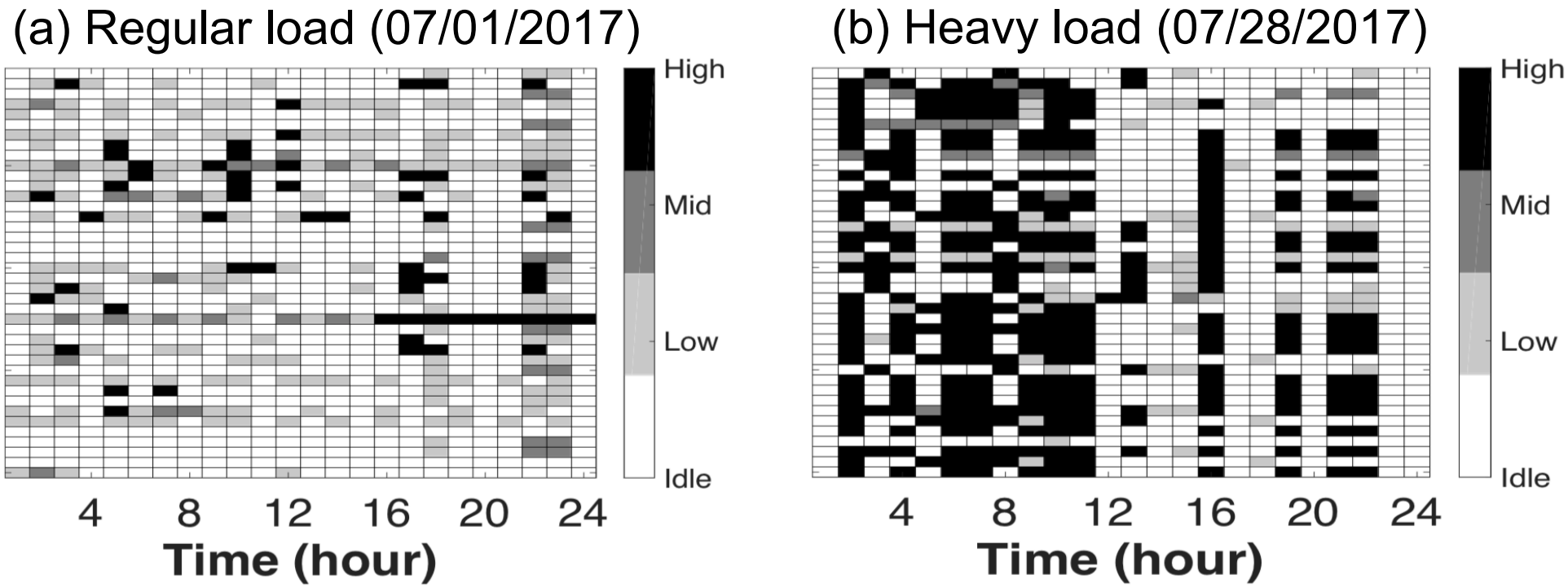
Problem: cache thrashing at forwarding nodes

- Lustre aggressive prefetching
- N-N I/O mode

Solution: reducing per-thread prefetching within same application

- Applying per-file prefetching limit
- Switching from N-N to N-M mode

Case Study 3: Unbalanced Forwarding Node Utilization

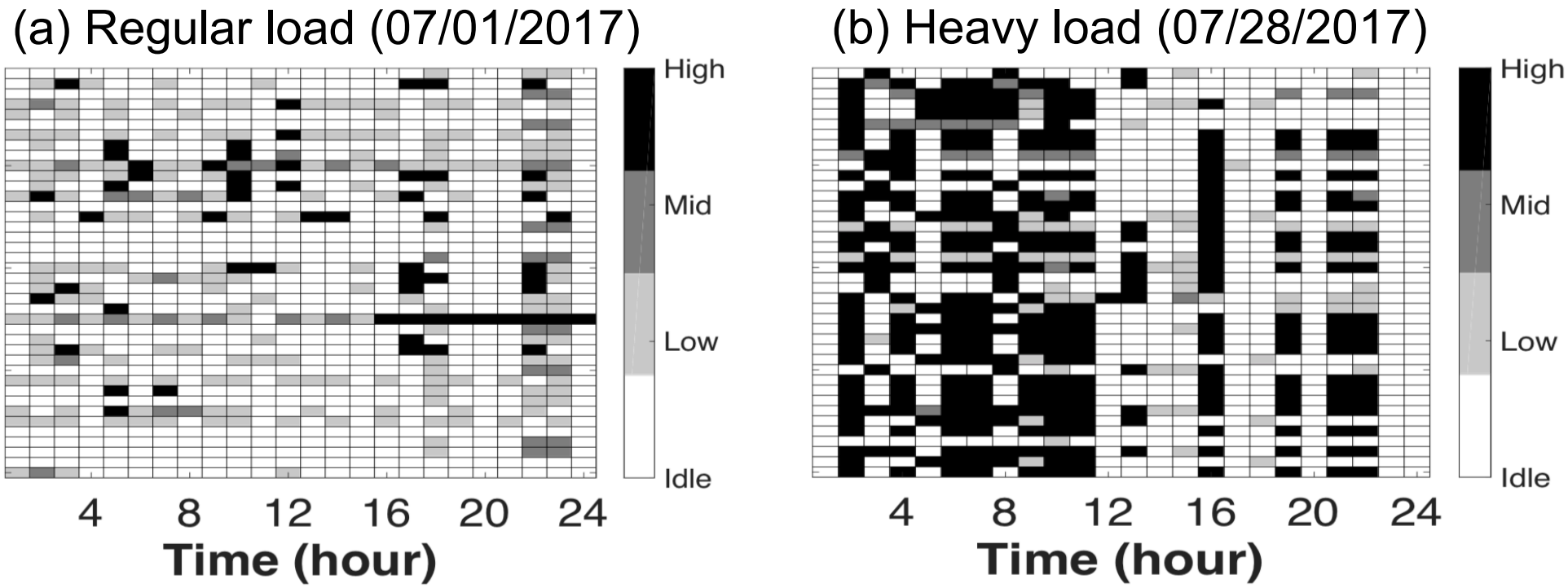


Sample TaihuLight one-day load summary, showing peak load level by hour

Problem: forwarding resource over-provisioning and load imbalance

- **Mostly under-utilized**
- **Overwhelmed under intense I/O bursts**
- **Application-oblivious forwarding node allocation**

Case Study 3: Unbalanced Forwarding Node Utilization

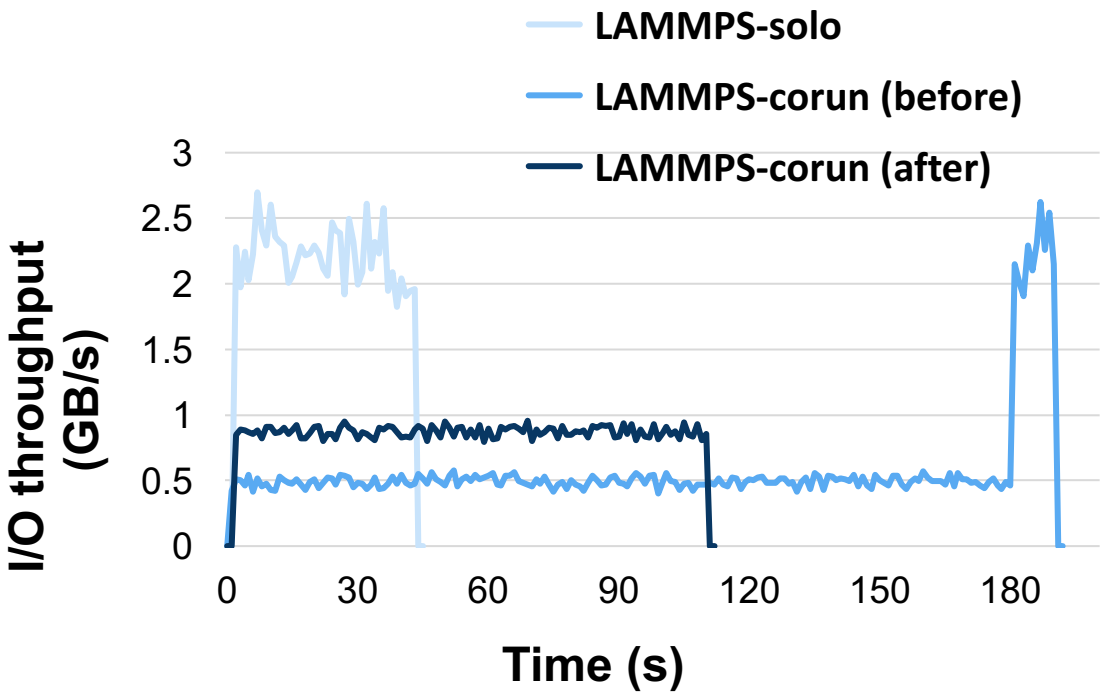


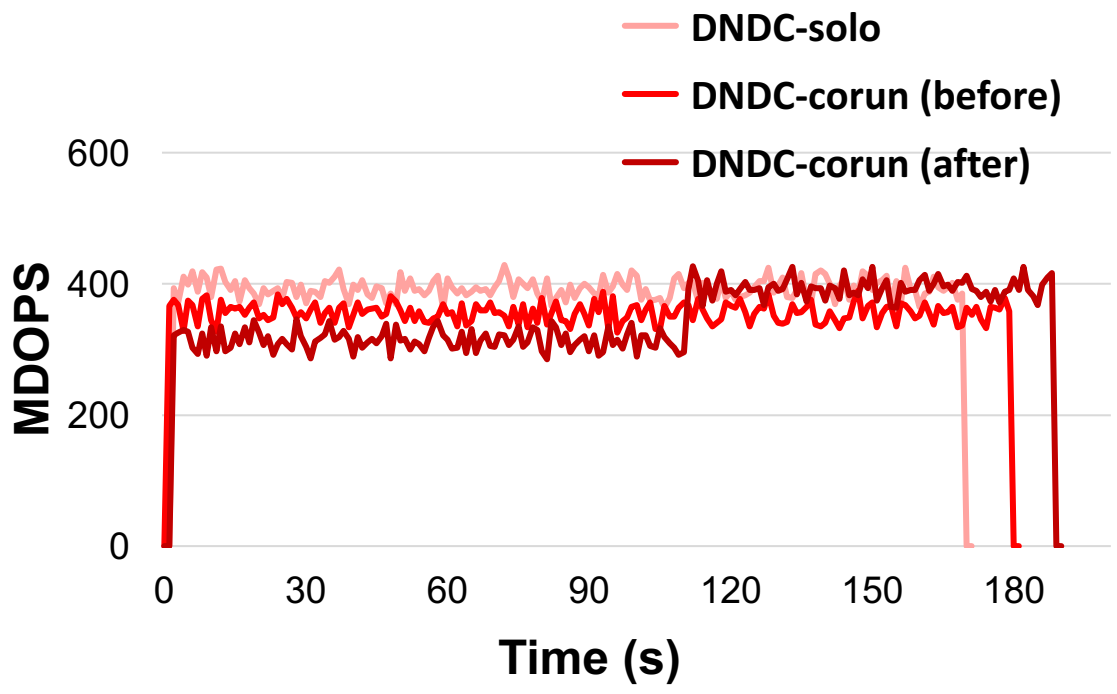
Sample TaihuLight one-day load summary, showing peak load level by hour

Solution: DFRA (Dynamic Forwarding Resource Allocation)

- **Automatic, application-aware allocation and isolation**
- **FAST 2019, presentation in an hour**

Case Study 4: MDS Request Priority Setting





Problem: metadata requests given priority over file I/O, based on

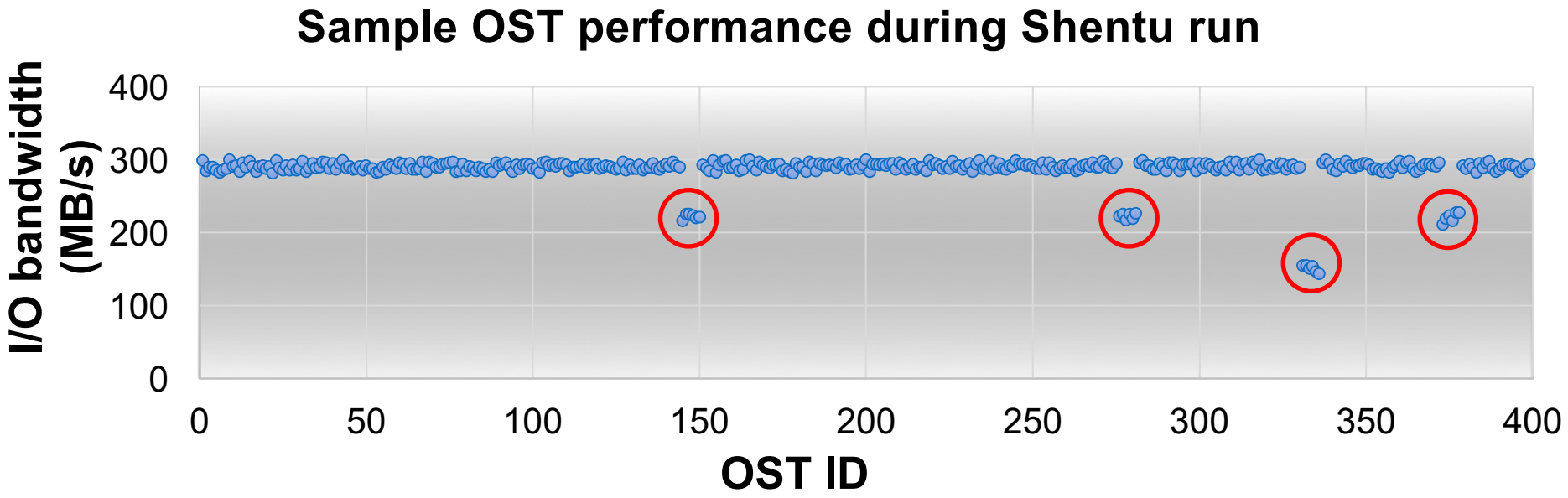
- **Single-MDS design**
- **Need for interactive user experience**

Solution: probabilistic processing between two queues

- **Metadata requests stop receiving full priority**

Case Study 5: Anomaly Detection Example

- Parallel graph engine Shentu
  - 160,000 processes
  - 400 Lustre OSTs



Solution: automatic screening of OSTs for performance anomaly

- **Redirect from slow ones temporarily**

Beacon Evaluation: Overhead on Applications

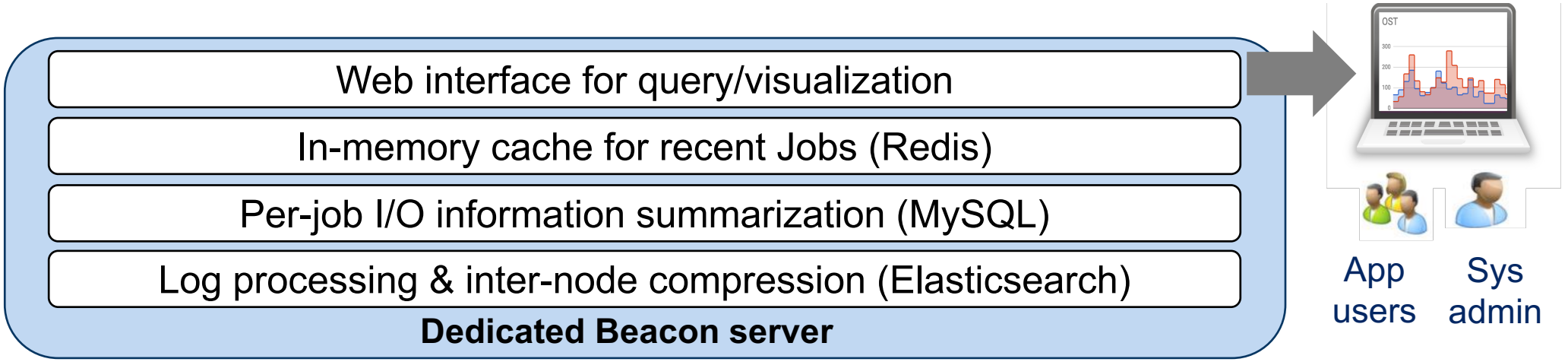
Application	#Process	$T_{w/o}$ (s)	T_w (s)	% Slowdown
MPI-IO_N	64	26.6	26.8	0.79%
MPI-IO_N	128	31.5	31.6	0.25%
MPI-IO_N	256	41.6	41.9	0.72%
MPI-IO_N	512	57.9	58.4	0.86%
MPI-IO_N	1024	123.1	123.5	0.36%
WRF_1	1024	2813.3	2819.1	0.21%
DNDC	2048	1041.2	1045.5	0.53%
XCFD	4000	2642.1	2644.6	0.09%
GKUA	16384	297.5	299.9	0.82%
GKUA	32768	182.8	184.1	0.66%
AWP	130000	3233.5	3241.5	0.25%
Shentu	160000	5468.2	5476.3	0.15%

Average application execution time: before and after Beacon turned on

Beacon Evaluation: Resource Consumption

Node type	CPU Usage	Memory Usage (MB)
Compute node	0.0%	10
Forwarding node	0.1%	6
Storage node	0.1%	5

Average resource consumption on monitoring nodes



Space consumption on dedicated Beacon server:
~10TB monitoring data (after 2 rounds of compression) in 18 months

Generality

- Beacon applies to other supercomputers
  - General building blocks
    - Operation log collection and compression
    - Scheduler-assisted, per-application data correlation and analysis
    - Online anomaly identification
  - Implemented using popular, open-source software
    - Logstash, Redis, ElasticSearch, MySQL

Conclusion

- Detailed, end-to-end I/O monitoring affordable and effective
  - Helps understand sources of inefficiency
  - Exposes hidden design or configuration problems
  - Facilitates better application I/O practice

https://github.com/Beaconsys/Beacon