Spectrum Scale (GPFS)

George S. Markomanolis,

HPC Engineer

Oak Ridge National Laboratory

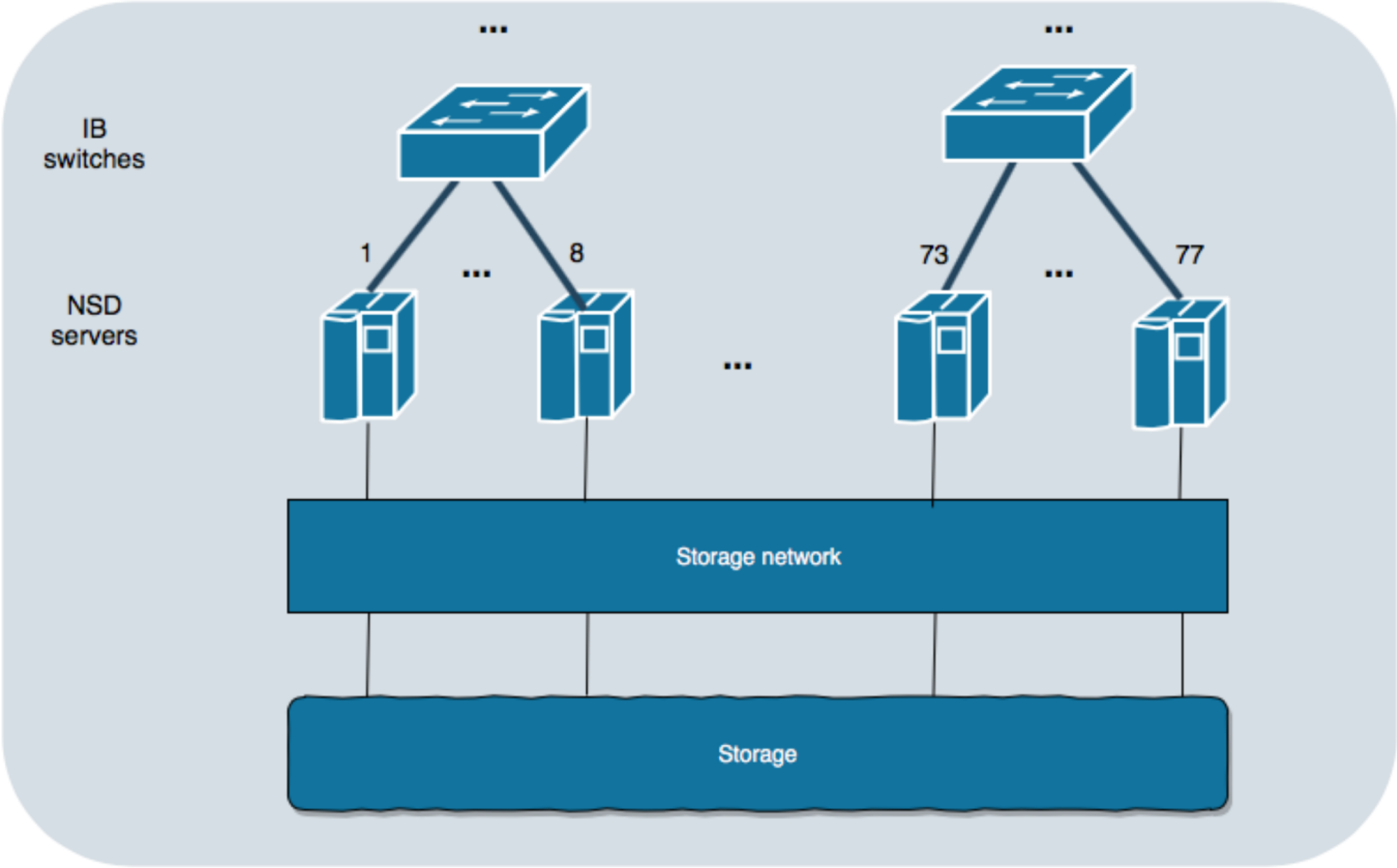
Summit Training Workshop

5 December 2018

Spider3 - Alpine

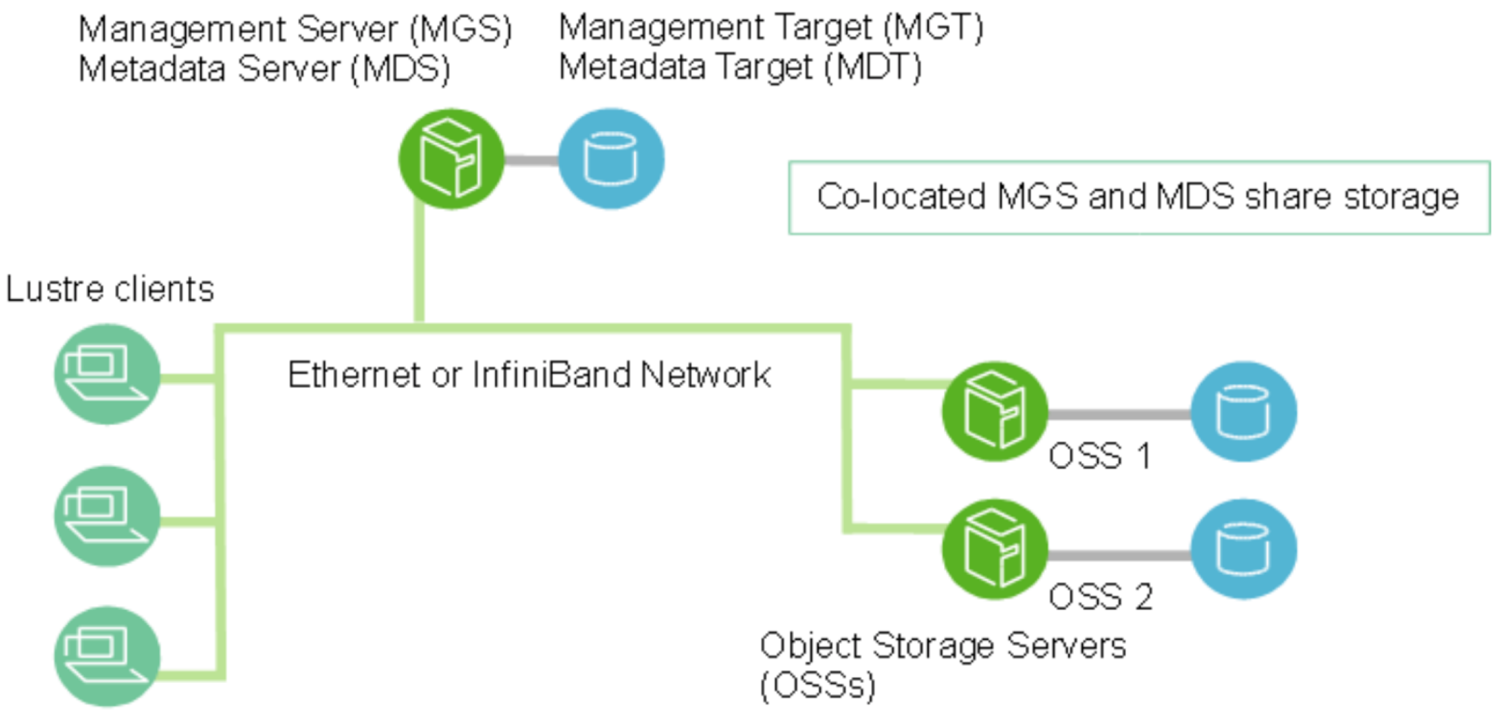
- Alpine, is a Spectrum Scale (ex-GPFS) file system of 250 PB of used space, which is mounted on Summit and Data Transfer Nodes (DTN) with maximum performance of 2.5 TB/s for sequential I/O and 2.2 TB/s for random I/O
- Largest GPFS file system installation
- Up to 2.6 million accesses per second of 32 KB small files
- It is constituted by 154 Network Shared Disk (NSD) servers
- It is a shared resource among users, supporting File Per Process (FPP), Single Shared File (SSF) and any of their combination
- EDR InfiniBand attached (100Gb/s)

Alpine – NSD servers



Atlas

- Atlas is the Lustre filesystem mounted on Titan



From Atlas to Alpine

Atlas	Alpine
User needs to stripe a folder for large files	User expects that system engineers did tune the file system
With striping, specific number of OSTs servers are used	All the NSD servers are used if the file is large enough
On Lustre there are specific number of metadata servers	On Spectrum Scale each storage server is also metadata server
On Lustre the number of the MPI I/O aggregators are equal to the number of the used OSTs	The number of the MPI I/O aggregators is dynamic, depending on the number of the used nodes

Alpine – IO-500

- IO-500 is a suite of benchmarks with 12 specific cases with purpose to extract the potential benefits of an HPC storage system based on IOR, mdtest and find tools
- During SC18, it achieved the #1 on IO-500 list, while using mainly the Spectrum Scale NLSAS and no Burst Buffer (http://io-500.org)

#	information							io500		
	institution	system	storage vendor	filesystem type	client nodes	client total procs	data	score	bw GiB/s	md kIOP/s
1	Oak Ridge National Laboratory	Summit	IBM	Spectrum Scale	504	1008	zip	366.47	88.20	1522.69
2	Korea Institute of Science and Technology Information (KISTI)	NURION	DDN	IME	2048	4096	zip	160.67	554.23	46.58
3	University of Cambridge	Data Accelerator	Dell EMC	Lustre	528	4224	zip	158.71	71.40	352.75
4	JCAHPC	Oakforest-PACS	DDN	IME	2048	16384	zip	137.78	560.10	33.89

Oak Ridge National Laboratory

What performance should we expect?

- It depends on your application and the used resources! Network could be the bottleneck if there is not enough available bandwidth available
- Results from IO-500, 504 compute nodes, 2 MPI processes per node

IOR-Write		IOR-Read	
Easy	Hard	Easy	Hard
2158 GB/s	0.57 GB/s	1788 GB/s	27.4 GB/s

- IOR Easy is I/O with friendly pattern for the storage with one file per MPI process
- IOR Hard is I/O with non-friendly pattern for the storage with a shared file
- You need always to be pro-active with the performance of your I/O

What performance should we expect? (cont.)

- It depends on the other jobs!
- There are many users on the system that they could perform heavy I/O
- The I/O performance is shared among the users

IOR Write – 10 compute nodes (not on full file system)	
Single IOR	Two concurrent IOR
144 GB/s	90 GB/s, 69 GB/s

- This is an indication that when your I/O performance does not perform as expected, you should investigate if any other large job is running with potential heavy I/O

Flags to improve I/O performance

- GPFS processes are operating only on the isolated core of each socket
- In order to give access to all the cores to handle GPFS requests, use the following option in your submission script

```
#BSUB -alloc_flags "smt4 maximizegpfs"
```

- The previous IOR write is decreased by up to 20% without the above flag
- **Important**: GPFS processes could interfere with an application, use the mentioned flag with caution and only if there is significant I/O

Spectrum Scale Internals

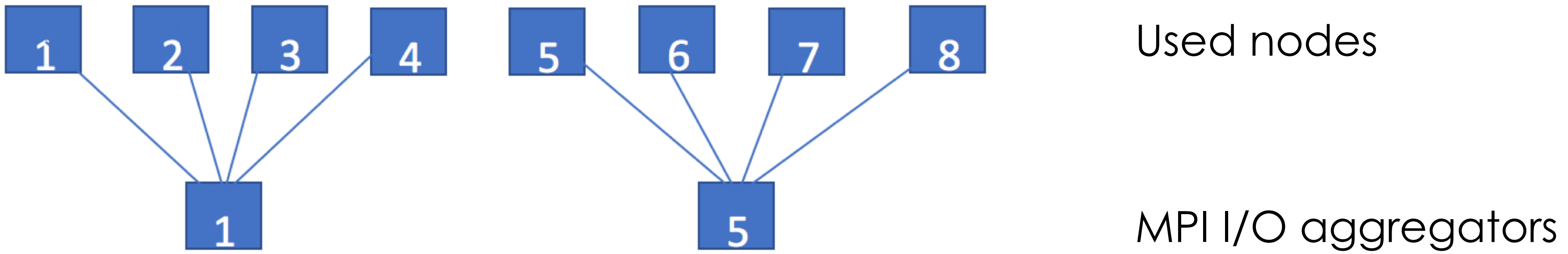
- Block-size: The largest size of I/O that Spectrum Scale can issue to the underlying device, on Summit it is 16MB

- All the previous IOR tests were executed with 16 MB block size

- A test with 2 MB of block-size provides write performance of 110GB/s, which is 23% less than using 16 MB of block-size.

Collective Buffering – MPI I/O aggregators

- During a collective write/read, the buffers on the aggregated nodes are buffered through MPI, then these nodes write the data to the I/O servers.
- Spectrum Scale calculates the number of MPI I/O aggregators based on the used resources. If we use 8 MPI nodes, then we have 2 MPI I/O aggregators



How to extract important information on **collective** MPI I/O

- Use the following declaration in your submission script

```
export ROMIO_PRINT_HINTS=1
```

- We have the following information in the output file for an example of 16 nodes with 16 MPI processes per node:

```
key = cb_buffer_size            value = 16777216
key = romio_cb_read             value = automatic
key = romio_cb_write            value = automatic
key = cb_nodes                  value = 16
key = romio_no_indep_rw         value = false
…
key = cb_config_list            value = *:1
key = romio_aggregator_list     value = 0 16 32 48 64 80 96 112 128 144 160 176 192 208 224 240
```

NAS BTIO

- NAS Benchmarks, block Tri-diagonal solver
- Test case:
  - 16 nodes with 16 MPI processes per node
  - 819 million grid points
  - Final output file size of 156 GB
  - Version with PNetCDF support
  - Blocking collective MPI I/O, single shared file among all the processes
- Write speed: **1532** MB/s
- That’s significant low performance although the I/O pattern is not friendly for most of the filesystems

NAS BTIO – Block size

- The default block size for Parallel NetCDF is 512 bytes when the striping_unit is not declared
- We create a file called romio_hints with the content:

striping_unit 16777216

- Then we define the environment variable ROMIO_HINTS pointing to the file romio_hints

export ROMIO_HINTS=/path/romio_hints

- New I/O write performance is **13602** MB/s
- Speedup of **8.9!!** times for the specific benchmark without editing or compiling the code

NAS BTIO – Hints

- Update the file romio_hints and define

```
romio_no_indep_rw true
```

- Then the processes that are not MPI I/O aggregators, they will not open the output file as they are not going to save any data on it

- New I/O write performance is **14316** MB/s
- The performance of the write, compare to the basic version, was improved almost **9.4** times
- The parameters that are required to modified are depending on the application, the resources, and the I/O pattern

NAS BTIO – Non-blocking PNetCDF

- We test the version with **non-blocking** collective PNetCDF with 16 nodes and 16 MPI processes per node.
- Default parameters provide write performance of **13985** MB/s, almost as the optimized blocking version.
- The exact same optimizations, provide **28203** MB/s, almost double performance.
- As the parallel I/O is non blocking we can increase the MPI I/O aggregators to evaluate the performance and we concluded that adding in the romio_hints the following command improves the performance

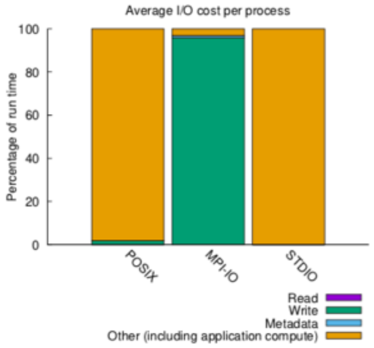
cb_config_list *:8

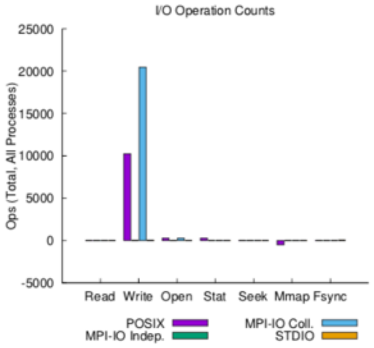
- With the above declaration, we have 8 MPI I/O aggregators per node and the performance now is **35509** MB/s, **2.54** times improved compare to the default results.

Darshan on Summit – Optimizing blocking PNetCDF

jobid: 170626	uid:	nprocs: 256	runtime: 99 seconds

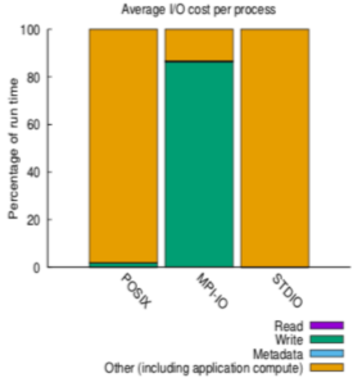
I/O performance *estimate* (at the MPI-IO layer): transferred 959 MiB at 1627.63 MiB/s
I/O performance *estimate* (at the STDIO layer): transferred 0.0 MiB at 21.65 MiB/s

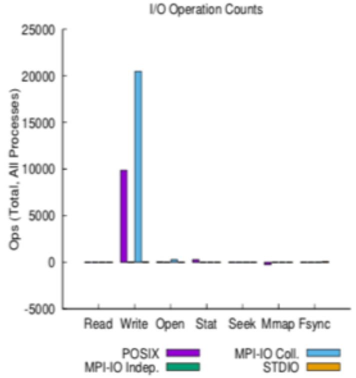




jobid: 171040	uid:	nprocs: 256	runtime: 13 seconds

I/O performance *estimate* (at the MPI-IO layer): transferred 1010 MiB at 13852.43 MiB/s
I/O performance *estimate* (at the STDIO layer): transferred 0.0 MiB at 26.65 MiB/s





Conclusion

- Use parallel I/O libraries that are optimized such as ADIOS, PNetCDF, HDF5 etc.
- Use non-blocking MPI I/O to improve the performance
- Do not re-invent the wheel!
- Remember that Alpine is a shared resource
- Use tools that provide insight I/O performance information such as Darshan

Acknowledgement

This research used resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC05-00OR22725.

Thank you!

Questions?