Making Lustre Data Aware

Cory Spitz

What do I know about diverse workloads?

- I've worked for an HPC vendor my entire career
- I regret that I am not a Lustre architect or core developer
- I lead a Lustre development, test, and support team at Cray, an early Lustre adopter and original OpenSFS promoter
- I have been an involved member of OpenSFS since its inception

Lustre for diverse workloads

- Workshop premise: Lustre is moving beyond scaling and performance for traditional large-scale HPC workloads (my words)
- “This workshop series is intended to help explore improvements in the performance and flexibility of Lustre for supporting non-scientific application workloads.”
- So let’s explore
- What does this mean to you?

What is Lustre for diverse workloads?

- **Diversity implies a growing use case portfolio**
- **What should our goals be? That is, how to prioritize?**
  - Previous focus was all at the top 100
  - Should there be more effort on the top 1000?
  - Upstream client and http://OpenHPC.community?
- **Big Data?**
  - Need data movers & data management to feed the beast
- **Lustre isn’t just a filesystem, its an ecosystem**
  - Consider what is integrated and what stays outside Lustre proper

Every one has a take

- Do we know the requirements?
- Can we articulate all of the solutions?
- Where do we go from here? [hint, we should make a list]

First, an excursion into exascale

- **Why?**
  - I think we want exascale technologies to trickle down
  - Also, I think exascale is really about high productivity (more later)
- **While it might be painful, we can scale up if we choose**
  - Deployments like the RIKEN's K Computer show us the way
- **Instead we're thinking of new ways to scale**
- **Unfortunately, there is no one canonical definition of an exascale Lustre filesystem**

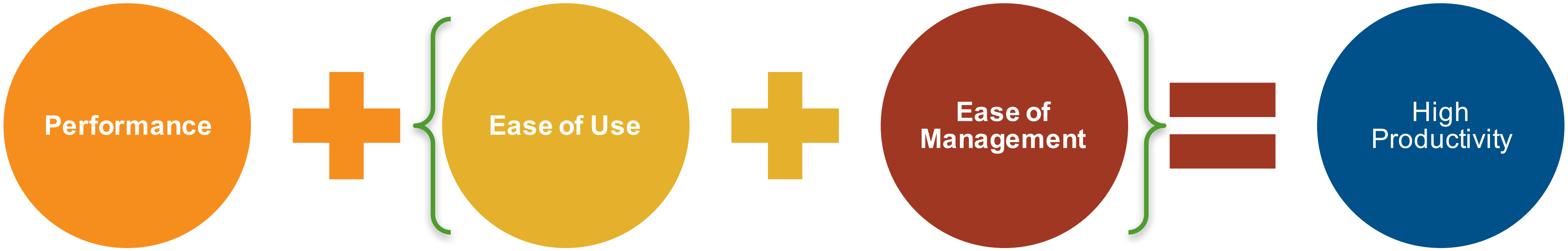
(Exascale) storage management confusion

- In fact, there are few clear paths forward
- We all seem to agree that there will be lots of devices, components, and threads
- Can't agree on a single solution for organization, workflows, access methods, or usage/semantics
- Multiple solutions should emerge, even hybrids
- For sure, things will be complex
- How will these complex systems be productive?

It’s (now) all about the data

- **Complex system with lots of parts yields broadly distributed data**
  - Yes, even at scale << exascale
- **We’re not used to HPC compute resources that have persistent storage**
  - Is this somehow different than lots of dispersed OSTs? NUMA OSTs?
  - How will we address and reference that data?
- **Lustre itself doesn’t provide the framework, tools, or technology to easily access or manage broadly distributed data**
- **Can we provide an ideal data sync that never fills?**
- **Do you know how to marshal your data? Does your admin?**
- **Solution: Lustre should be data aware; only one problem…**

Data management yields high productivity



Performance + { Ease of Use + Ease of Management } = High Productivity

Data Management

- **Performance – small file I/O, single client, streaming data**
  - Widen Lustre’s sweetspot
- **Ease of Use – automation of data movement**
- **Ease of Management – automation of data movement**

Happy coincidence

- **My view: some exascale requirements and some diverse workload requirements can be viewed as one and the same**
- **Exascale should not consist of specialized HW & SW that will only be used on the top systems**
  - There will be a trickle down effect
- **Lots of devices ➔ hard to manage**
- **Lots of users/use cases ➔ hard to manage**
- **Data awareness; data permanence; provenance of data?**
- **How to decide?**

OpenSFS technical leadership

- **Lustre community ecosystem is in flux**
- **We used to have a Technical Working Group which advised how to spend the large sums of money that the OpenSFS Promoters provided**
  - We enjoyed great leadership from the likes of John Carrier, Dave Dillow, and Jason Hill
  - The TWG had essentially two tasks: gather requirements and propose features
- **In the early days we debated, but the direction was more or less clear and defined in a requirements doc**
- **The last time we acted in this role was 2012(!)**

We’re still operating on 2012 requirements!

- **Presently OpenSFS has fewer funds to disperse**
  - TWG was subsumed into the Lustre Working Group
  - Development contracts have wrapped up and no new recommendations or contracts have been generated
- **We’re thankful for investments outside OpenSFS…**
  - (E.g., Progressive File Layouts funded by ORNL)
- **…but only two of seven Storage Management Requirements are realized in Lustre today**
- **Conclusion: we’re more or less where we were in 2012**

Recommendations to you

- **Gather requirements for diverse workloads**
- **Complete a survey?**
  - Possible questions
    - Legacy (POSIX) apps? Only?
    - Are there special performance characteristics (many threads per client?)
    - User managed containers?
    - System managed containers?
- **Publish (to OpenSFS LWG)**
- **Vote (with your voice and/or money)**

Recommendations to OpenSFS

- Reinvigorate requirement gathering
- Incorporate input and design possible solutions
- Adopt solutions that have been successfully demonstrated elsewhere
- Evolve Lustre

If Lustre is a (sledge) hammer, we need to evolve it into a Swiss Army Knife (with a hammer)

Consequences

- **No one should take Lustre for granted**
- **If we don’t make Lustre {flexible to diverse workloads, (exa)scale, <your favorite thing here>} we run the risk of users going elsewhere**
- **Let’s not put ourselves behind [5-10 years]**
  - Lustre is battle tested
  - “You don’t just write a filesystem” –Brent Gorda, yesterday
- **Further, we’ve got a working ecosystem**
  - More than ‘don’t reinvent the wheel’; more like ‘don’t reinvent the car’

Discussion and Q&A

Cory Spitz
spitzcor@cray.com

Thank You

- Thank you to our sponsors, ORNL and US DoD
- A special thank you to Neena and Sarp and the organizers

Thank You

Cory Spitz
spitzcor@cray.com

Legal Disclaimer

Information in this document is provided in connection with Cray Inc. products. No license, express or implied, to any intellectual property rights is granted by this document.

Cray Inc. may make changes to specifications and product descriptions at any time, without notice.

All products, dates and figures specified are preliminary based on current expectations, and are subject to change without notice.

Cray hardware and software products may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Cray uses codenames internally to identify products that are in development and not yet publically announced for release. Customers and other third parties are not authorized by Cray Inc. to use codenames in advertising, promotion or marketing and any use of Cray Inc. internal codenames is at the sole risk of the user.

Performance tests and ratings are measured using specific systems and/or components and reflect the approximate performance of Cray Inc. products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance.

The following are trademarks of Cray Inc. and are registered in the United States and other countries: CRAY and design, SONEXION, and URIKA. The following are trademarks of Cray Inc.: APPRENTICE2, CHAPEL, CLUSTER CONNECT, CRAYPAT, CRAYPORT, ECOPHLEX, LIBSCI, NODEKARE, THREADSTORM. The following system family marks, and associated model number marks, are trademarks of Cray Inc.: CS, CX, XC, XE, XK, XMT, and XT. The registered trademark LINUX is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis. Other trademarks used in this document are the property of their respective owners.