DataWarp Transparent Cache: Implementation, Challenges, and Early Experience

CUG 2018

Benjamin Landsteiner, Cray Inc; David Paul, NERSC/LBNL

Agenda

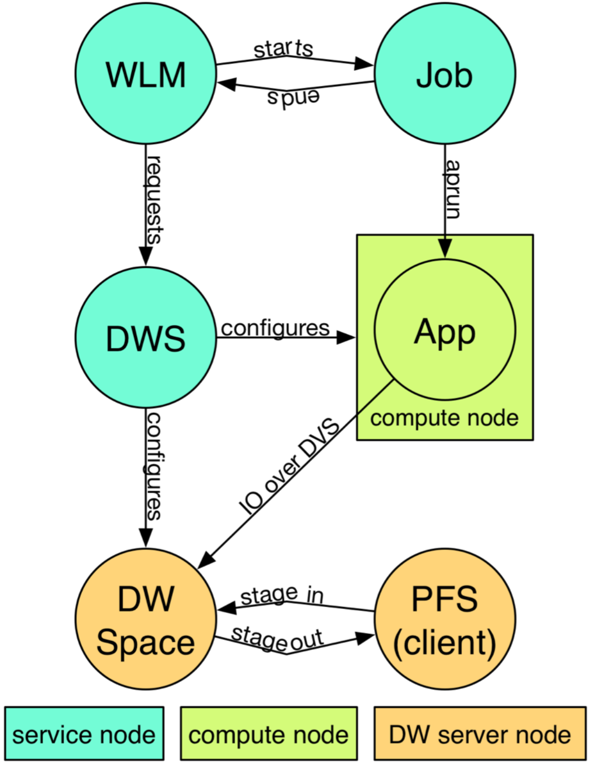
- DataWarp Background
- Transparent Cache
  - Usage
  - Data Path software
  - Orchestration software
- Early Results

DataWarp Overview

- **DataWarp is an *IO Accelerator***
  - Higher bandwidth, lower latency
- **Uses fast SSDs to absorb and serve IO faster than capacity-provisioned Parallel File Systems (PFS)**
- **Integration with workload managers enables users to tune DataWarp on a per-job basis**
  - PBS, Moab/TORQUE, Slurm

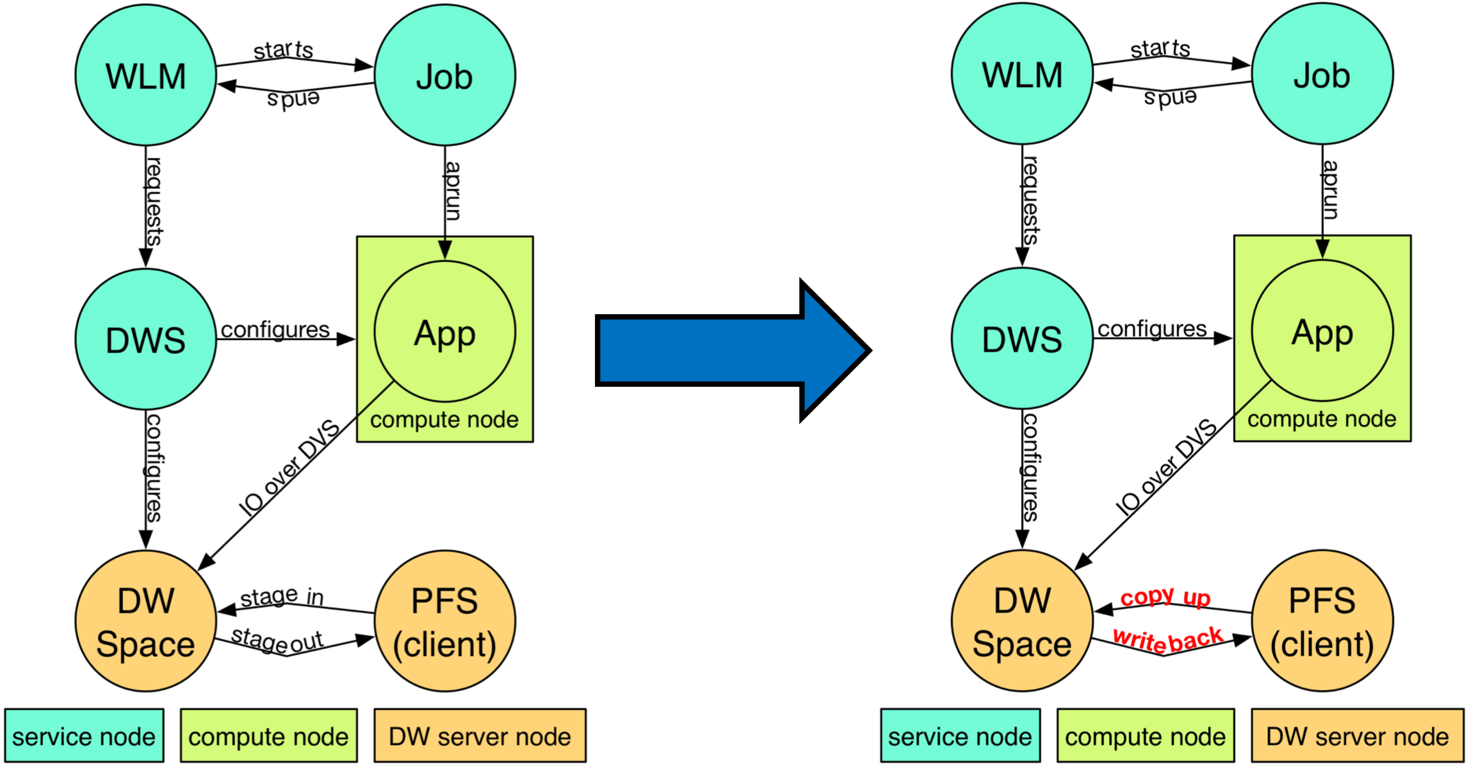
- **Existing scratch environment requires users to explicitly manage data transfer between the PFS and SSDs**
  - Start off with an empty filesystem
- **With the new transparent caching environment, DataWarp manages data transfer automatically**
  - Start off with a filesystem that looks like lustre, but faster
  - Easier to use, easier to get started

Basic DataWarp (scratch)



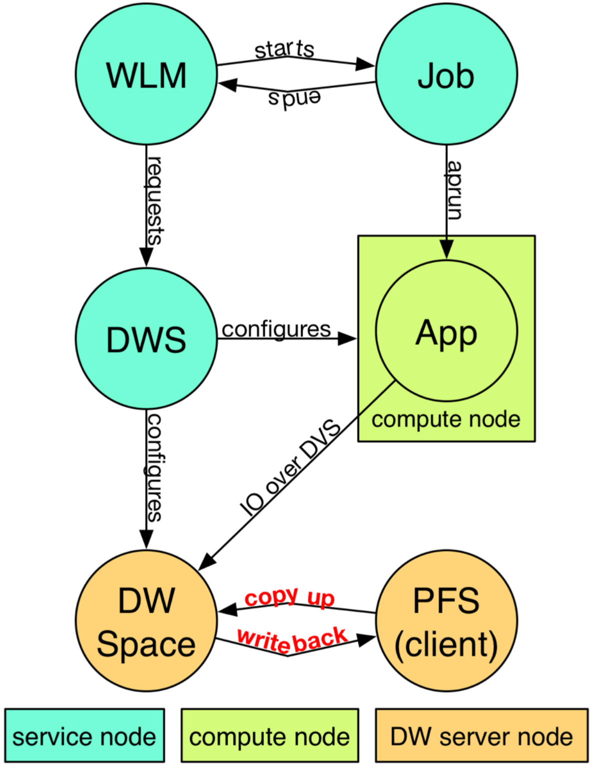
- WLM queues job, requests DWS set up job for using DW
- DataWarp Service (DWS) configures DW space, compute node access to DW
  - DW space is striped across one or more DW servers
- DataWarp File System handles data transfer interactions with PFS
  - Staging, like copy
  - User-initiated
- Compute nodes access DW via a mount point

Basic DataWarp (scratch -> cache)



Basic DataWarp (cache)

- Similar to scratch
- DataWarp File System handles data transfer interactions with PFS
  - Automatic
  - User just does IO



Using Transparent Cache

- **Primary method of user interaction**
  - Batch job script directives
  - DataWarp mount point
- **User specifiable attributes in batch job script directives**
  - PFS path to be transparently accelerated
  - Buffer capacity
- **Using the DataWarp mount point**
  - Environment variable available to batch job script contains the DataWarp path of the transparent caching filesystem
  - *ls* on DataWarp path looks like *ls* on lustre path

User Job Examples (Slurm, scratch comparison)

Without DataWarp

```
#!/bin/bash
#SBATCH --ntasks 3200

export JOBDIR=/lus/global/my_jobdir
srun -n 3200 a.out
```

With DataWarp Scratch

```
#!/bin/bash
#SBATCH --ntasks 3200
#DW jobdw type=scratch access_mode=striped capacity=1TiB
#DW stage_in type=directory source=/lus/global/my_jobdir destination=$DW_JOB_STRIPED
#DW stage_out type=directory source=$DW_JOB_STRIPED destination=/lus/global/my_jobdir

export JOBDIR=$DW_JOB_STRIPED
srun -n 3200 a.out
```

User Job Examples (Slurm, cache comparison)

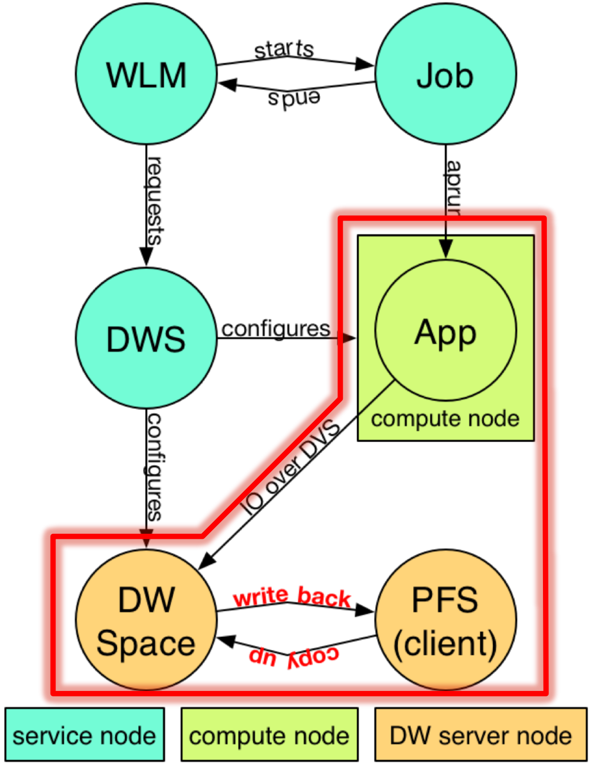
Without DataWarp

```
#!/bin/bash
#SBATCH --ntasks 3200

export JOBDIR=/lus/global/my_jobdir
srun -n 3200 a.out
```

With DataWarp Transparent Cache

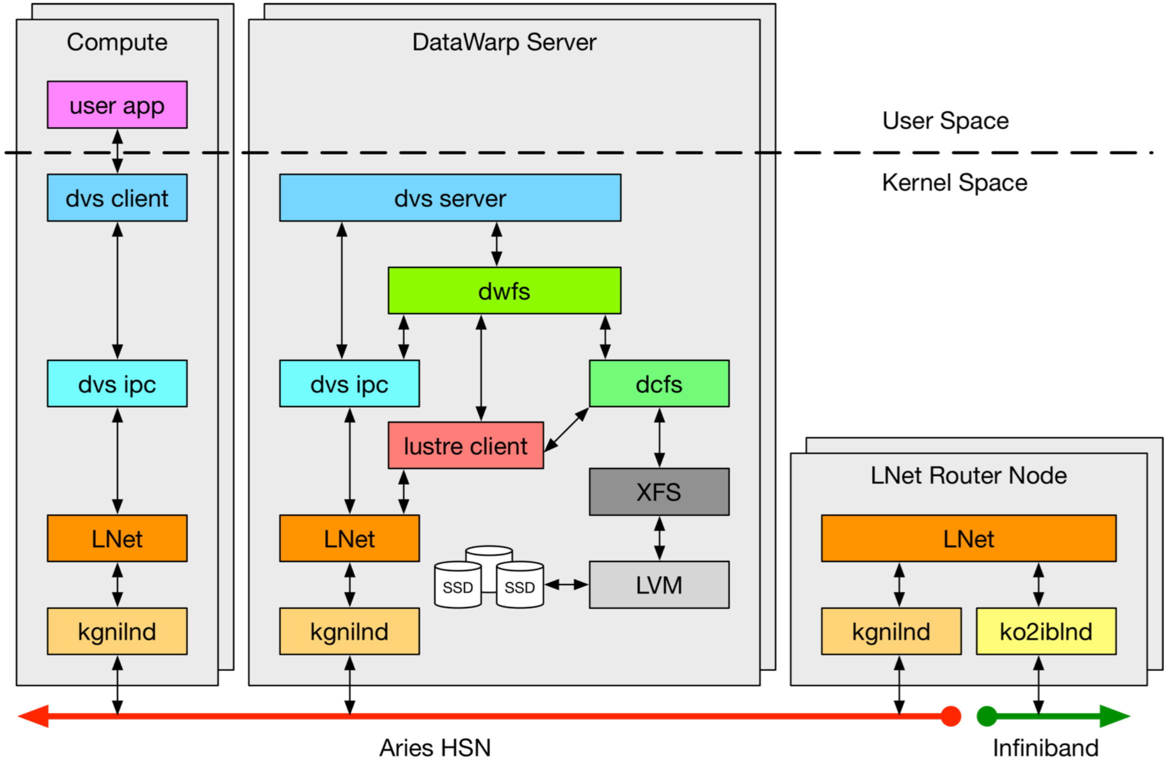
```
#!/bin/bash
#SBATCH --ntasks 3200
#DW jobdw type=cache access_mode=striped pfs=/lus/global capacity=10TiB

export JOBDIR=$DW_JOB_STRIPED_CACHE/my_jobdir
srun -n 3200 a.out
```

Transparent Cache Data Path

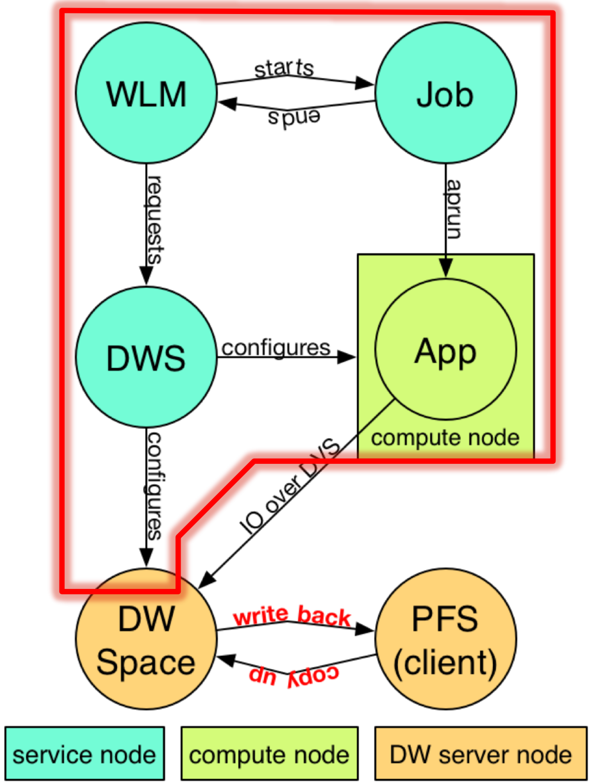


- **Compute nodes**
  - DVS client
- **DataWarp nodes**
  - DVS server
  - SSD space
  - DataWarp File System
  - Data Caching Filesystem
  - PFS client

Transparent Cache Data Path Interactions

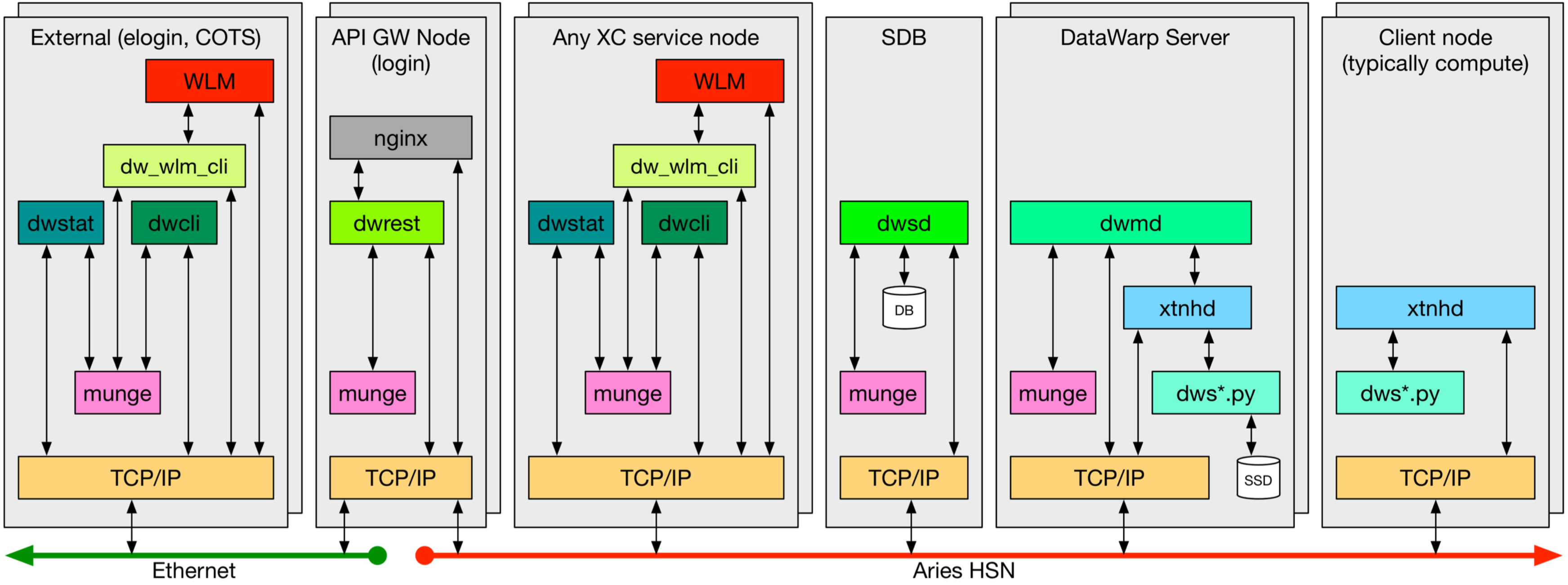


Transparent Cache Orchestration



- **Sets up and manages the data path**
- **Workload Managers**
- **DataWarp Service**
- **Node Health services**
  - Scalable fanout of commands
- **MUNGE**
  - Security

Transparent Cache Orchestration SW Overview



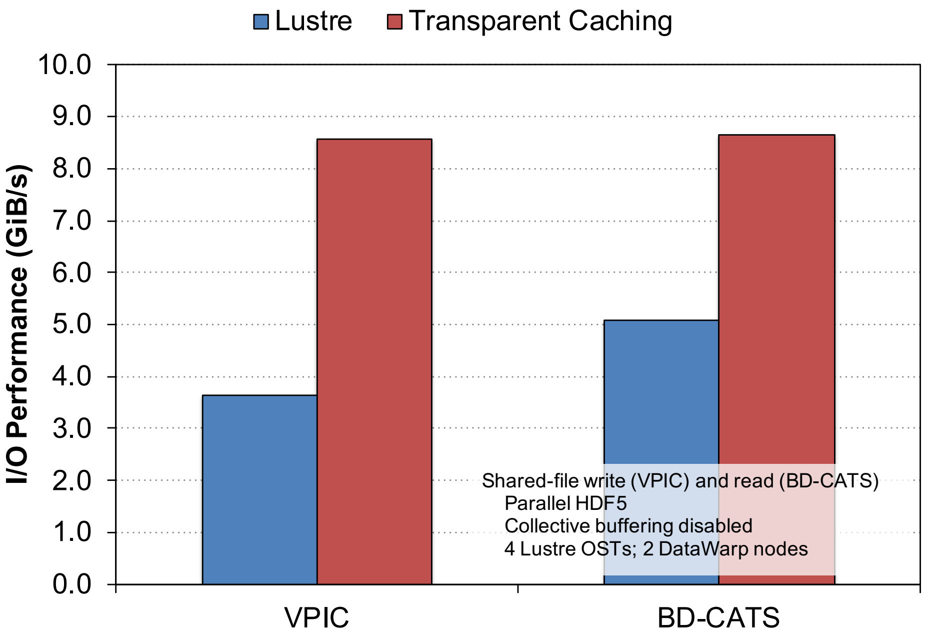
NERSC Early Results

- **Installation on Cori and extensive benchmarking has been overcome by other events**
  - Known limitations of early code were identified and understood
  - Anticipating GA release with UP06
  - Cori most likely will skip UP06 and target UP07
  - (UP05 installed week of 5/08)
- **Initially will be restricted to staff testing on Cori with alternate pool of (8) DataWarp servers (dev_pool)**
  - To isolate performance/failures from production pool

NERSC TDS Early Results

- **Initial Patch set applied to TDS (Gerty) at UP04**
  - Two DataWarp servers and smaller Lustre scratch
- **Branched image from Production cfgset and generated node images**
  - Used BTRFS ‘snapshot’ feature
- **‘Rolled Back’ for security patch installation (pre-Cori)**
- **TC Tested to be functional and stable**
- **TDS updated to UP05, Cray provided updated patch set, awaiting installation**

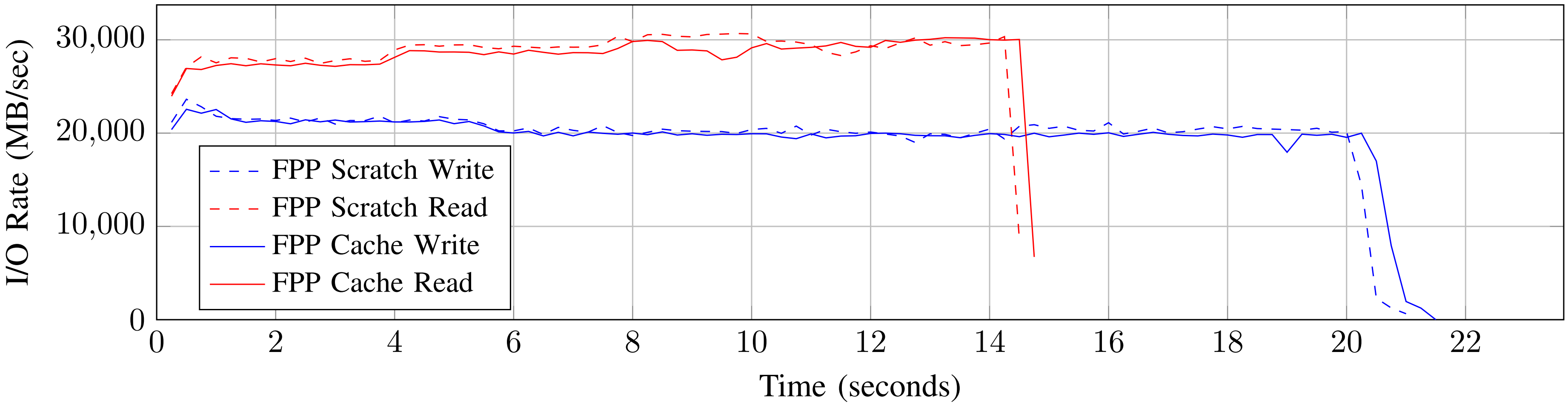
VPIC



Shared-file write (VPIC) and read (BD-CATS)
Parallel HDF5
Collective buffering disabled
4 Lustre OSTs; 2 DataWarp nodes

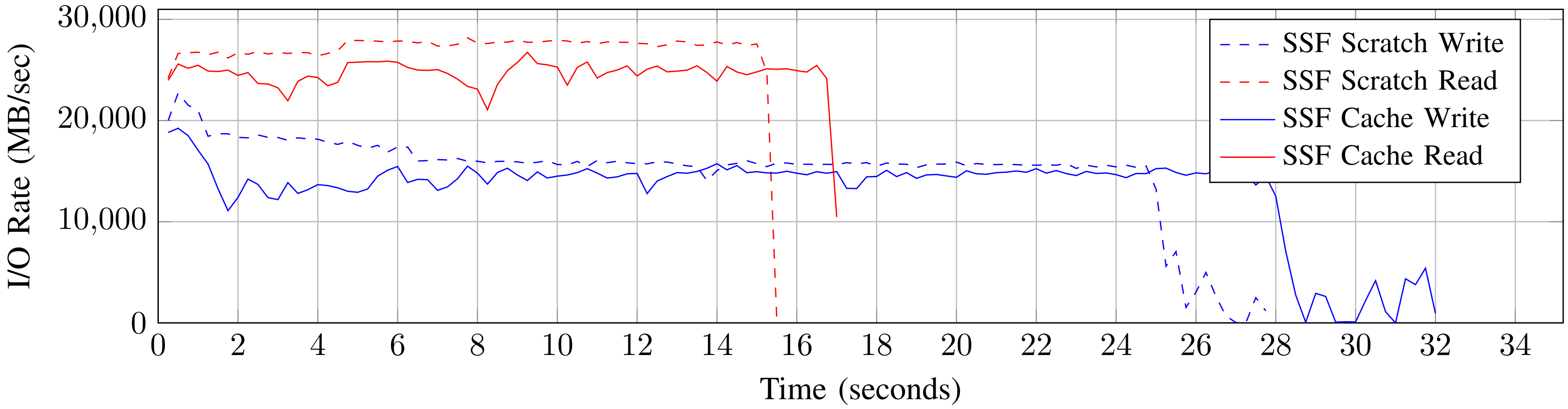
(Courtesy Glenn Lockwood – NERSC)

Early Results: Scratch vs Cache FPP



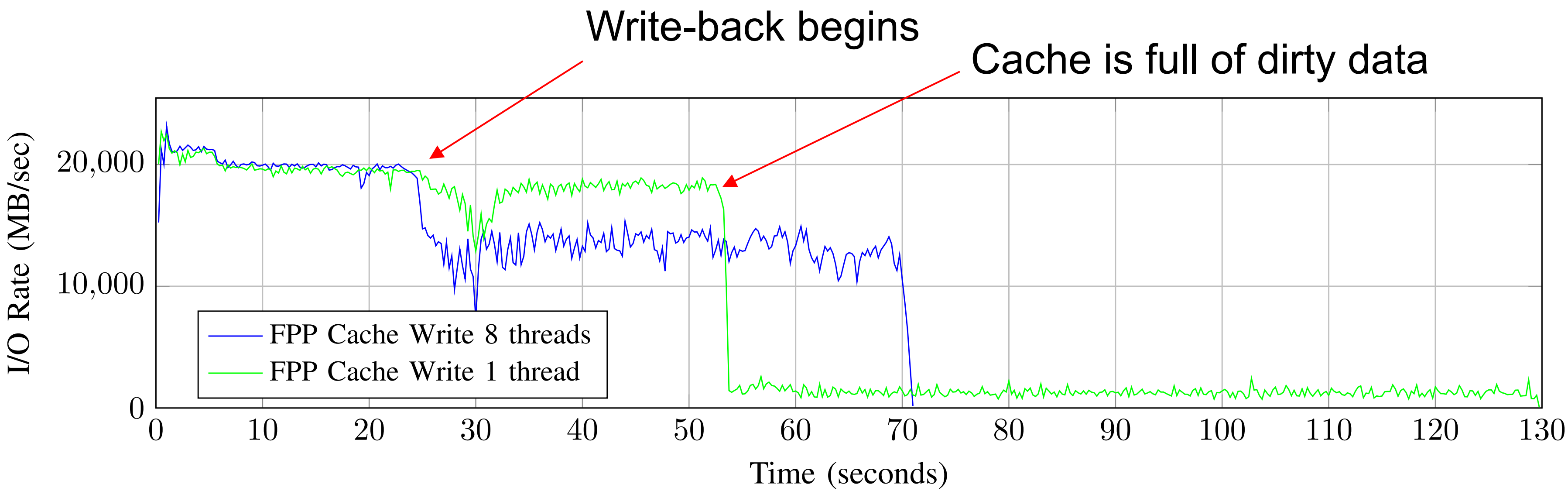
Cache is 2.2% slower for writes, 2.2% slower for reads

Early Results: Scratch vs Cache SSF



Cache is 13.4% slower for writes, 9.9% slower for reads

Early Results: 1 vs 8 write-back threads



Performance when write-back begins and when cache fills

Recap

- DataWarp Transparent Caching is in CLE 6.0.UP06
- Usability is improved over Scratch
- Transparent Cache data path re-uses all of the Scratch components
- Orchestration components are also re-used
- Early results are promising

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 publicly 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, URIKA and YARCDATA. The following are trademarks of Cray Inc.: CHAPEL, CLUSTER CONNECT, CLUSTERSTOR, CRAYDOC, CRAYPAT, CRAYPORT, DATAWARP, ECOPHLEX, LIBSCI, NODEKARE, REVEAL. 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 on this website are the property of their respective owners.

Q&A

Benjamin Landsteiner
ben@cray.com

David Paul
dpaul@lbl.gov