Data Transfer and Filesystems

07/29/2010

Mahidhar Tatineni, SDSC

Acknowledgements:

Lonnie Crosby , NICS
Chris Jordan, TACC
Steve Simms, IU
Patricia Kovatch, NICS
Phil Andrews, NICS

Background

- Rapid growth in computing resources/performance => a corresponding rise in the amount of data created, moved, stored, and archived.
- Large scale parallel filesystems (Lustre, GPFS) use striping across several disk resources, with multiple I/O servers to achieve bandwidth and scaling performance => need to understand the I/O subsytem to compute at large scale.
- Post-processing, visualization, and archival resources can be at a different site than the compute resources; Input data for large scale computations and processing codes can come from various sources (including non-computational) => need high speed data transfer options to and from the compute site.
- Computational/processed results important to wider science community => need nearline and archival storage with easy access; Data preservation is also important.

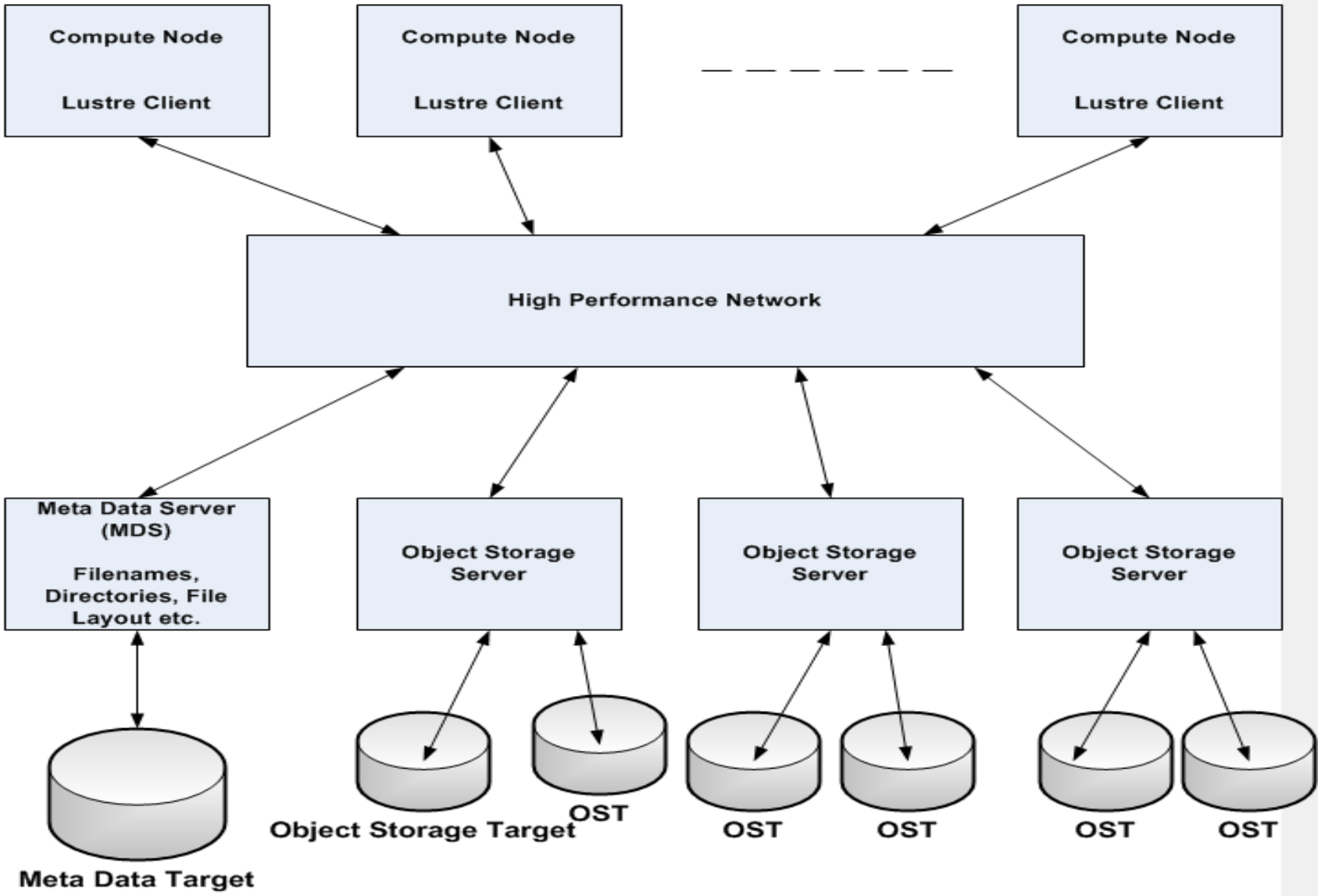
Outline of Talk

- Parallel filesystems – Lustre I/O optimization tips and examples using resources in TeraGrid.
- Wide area network (WAN) filesystems – JWAN, Lustre-WAN, GPFS-WAN.
- Data Transfer options – simple (scp, scp-hpn), threaded (bbftp, bbcp) to threaded and striped (gridftp). Specific examples using resources in TeraGrid including via the TeraGrid Portal.
- Data management – medium term storage (for processing), long term nearline/online storage (for community access), and long term archival (including archive replication). Specific examples from TeraGrid.
- Data work flow example – SCEC

Lustre Filesystem

- I/O striped across multiple storage targets. I/O subsystem processes on object storage servers (OSS).
- I/O from multiple compute nodes goes through high performance interconnect and switching to the OSSs.
- User can control stripe count (how many storage targets to use), stripe size, and stripe index (which OST to start with) for any given file or directory. Stripe size and stripe count can affect performance significantly and must be matched with the type of I/O being performed.
- Meta data operations can be a bottleneck => avoid lots of small reads and writes; aggregate the I/O (preferably to match the stripe parameters for the file).

Lustre Filesystem Schematic



Lustre Filesystem (Details)

- Metadata, such as filenames, directories, permissions, and file layout, handled by the metadata server (MDS), backended to the metadata target (MDT).
- Object storage servers (OSSes) that store file data on one or more object storage targets (OSTs).
- Lustre Client(s) can access and use the data.
- The storage attached to the servers is partitioned, optionally organized with logical volume management (LVM) and/or RAID, and formatted as file systems. The Lustre OSS and MDS servers read, write, and modify data in the format imposed by these file systems.
- Clients get file layout info from MDS, locks the file range being operated on and executes one or more parallel read or write operations directly to the OSTs via the OSSs.

Factors influencing I/O

- Large scale I/O can be a very expensive operation with data movement /interactions in memory (typically distributed over thousands of cores) and on disk (typically hundreds of disks).
- Characteristics of computational system. High performance interconnect can be a factor.
- Characteristics of filesystem – network between compute nodes and I/O servers, number of I/O servers, number of storage targets, characteristics of storage targets.
- I/O Patterns – Number of processes, files, characteristics of file access (buffer sizes etc).

I/O Scenarios

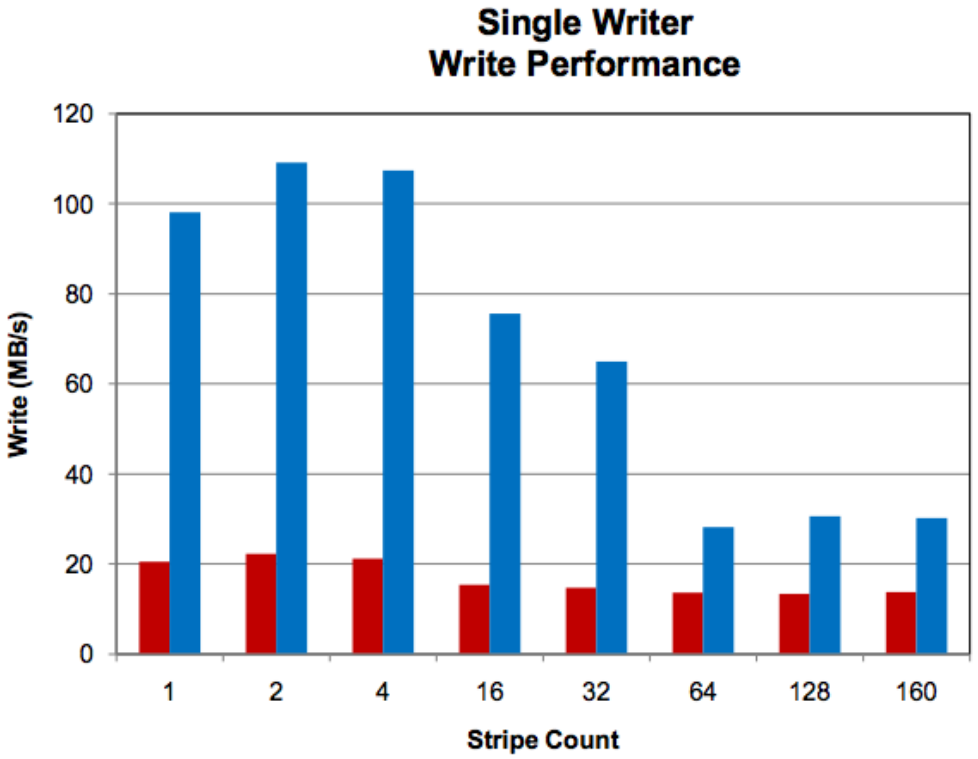
- Serial I/O: One process performs the I/O. Computational task may be serial or parallel. In the case of parallel computation this means aggregation of I/O to one task => high performance interconnect becomes a major factor. Time scales linearly with number of tasks and memory can become an issue at large core counts.
- Parallel I/O with one file per process: Each computational process writes individual files => the I/O network, metadata resources become very important factors.
- Parallel I/O with shared file: Data layout in shared file is important, at large processor counts the high performance interconnect, I/O network can be stressed.
- Combinations of I/O patterns with aggregation, subgroups of processors writing files.

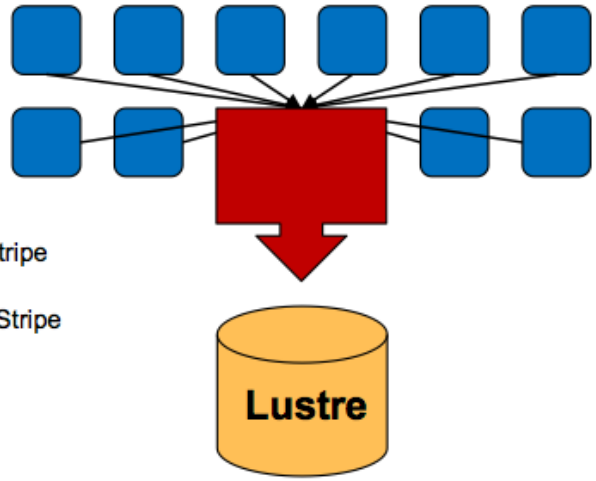
I/O Scenarios

- Detailed performance considerations for each scenario in upcoming slides.
- Low core counts (<256 cores) – serial, simple parallel I/O (one file per core) are o.k. Easy to implement.
- Medium core counts (<10k cores) – simple parallel I/O not recommended but feasible (starts to hit limits). If absolutely needed (due to memory constraints), stagger the individual file I/O to avoid metadata contention.
- Large core counts (>10k cores) – file per core scenario should **always** be done asynchronously from different cores. For MPI-IO aggregation to write from processor subsets is recommended. This can lower the metadata and filesystem resource contention.

Single writer performance and Lustre

- **32 MB per OST (32 MB – 5 GB) and 32 MB Transfer Size**
  - **Unable to take advantage of file system parallelism**
  - **Access to multiple disks adds overhead which hurts performance**

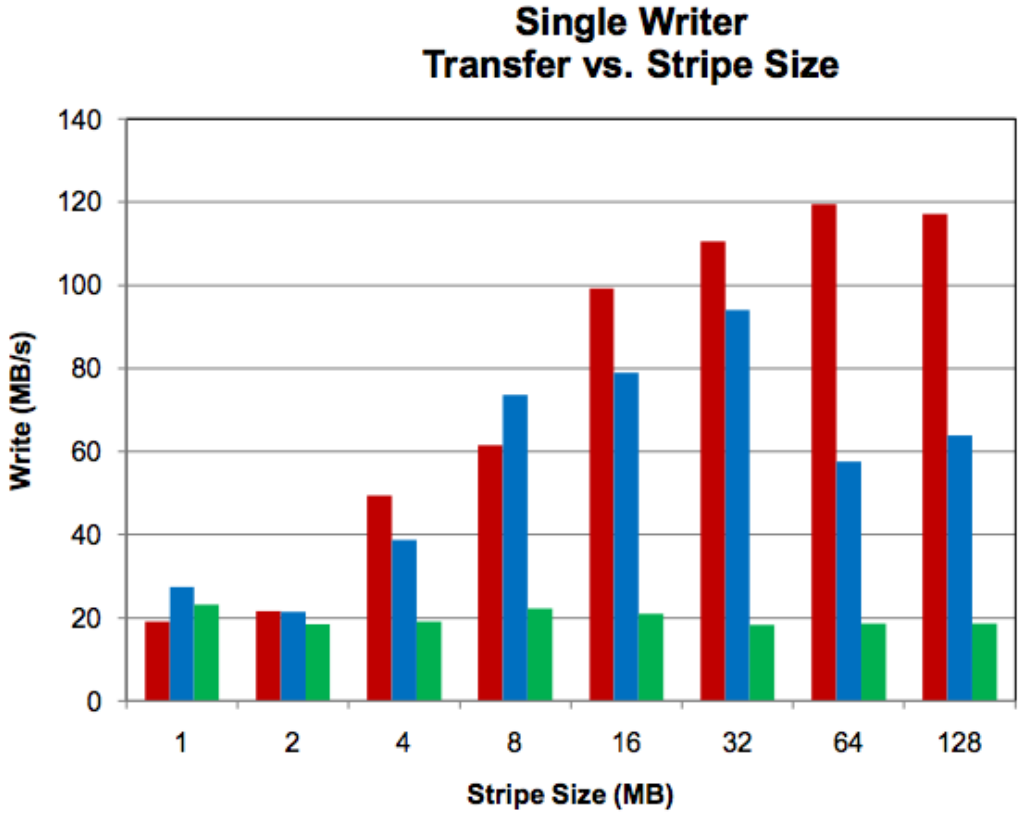


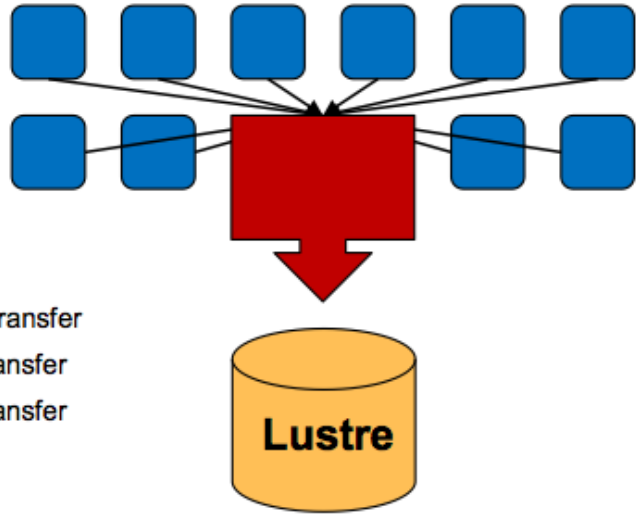


SOURCE: Lonnie Crosby, NICS

Stripe size and I/O Operation size

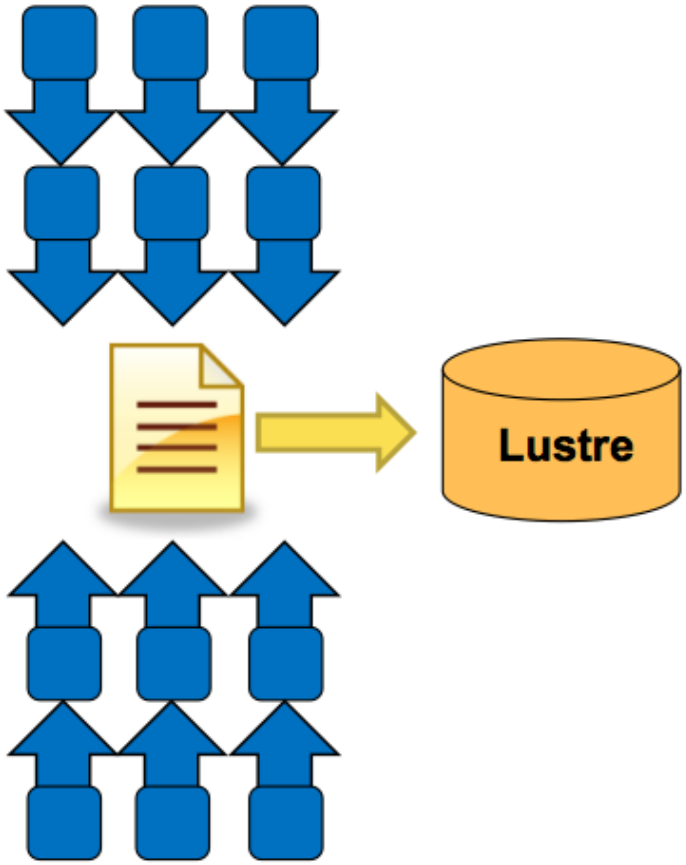
- **Single OST, 256 MB File Size**
  - **Performance can be limited by the process (transfer size) or file system (stripe size)**



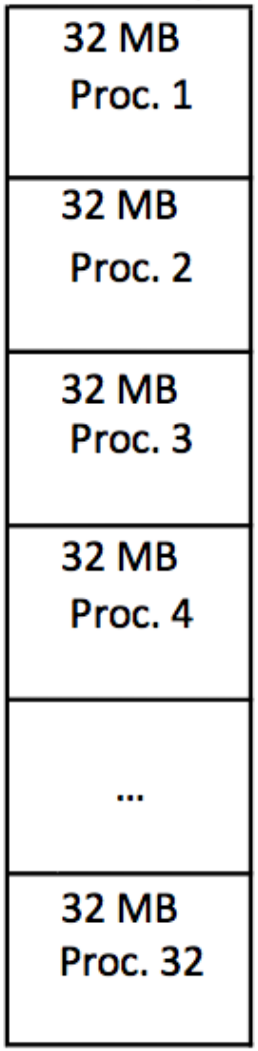


SOURCE: Lonnie Crosby, NICS

Single Shared Files and Lustre Stripes

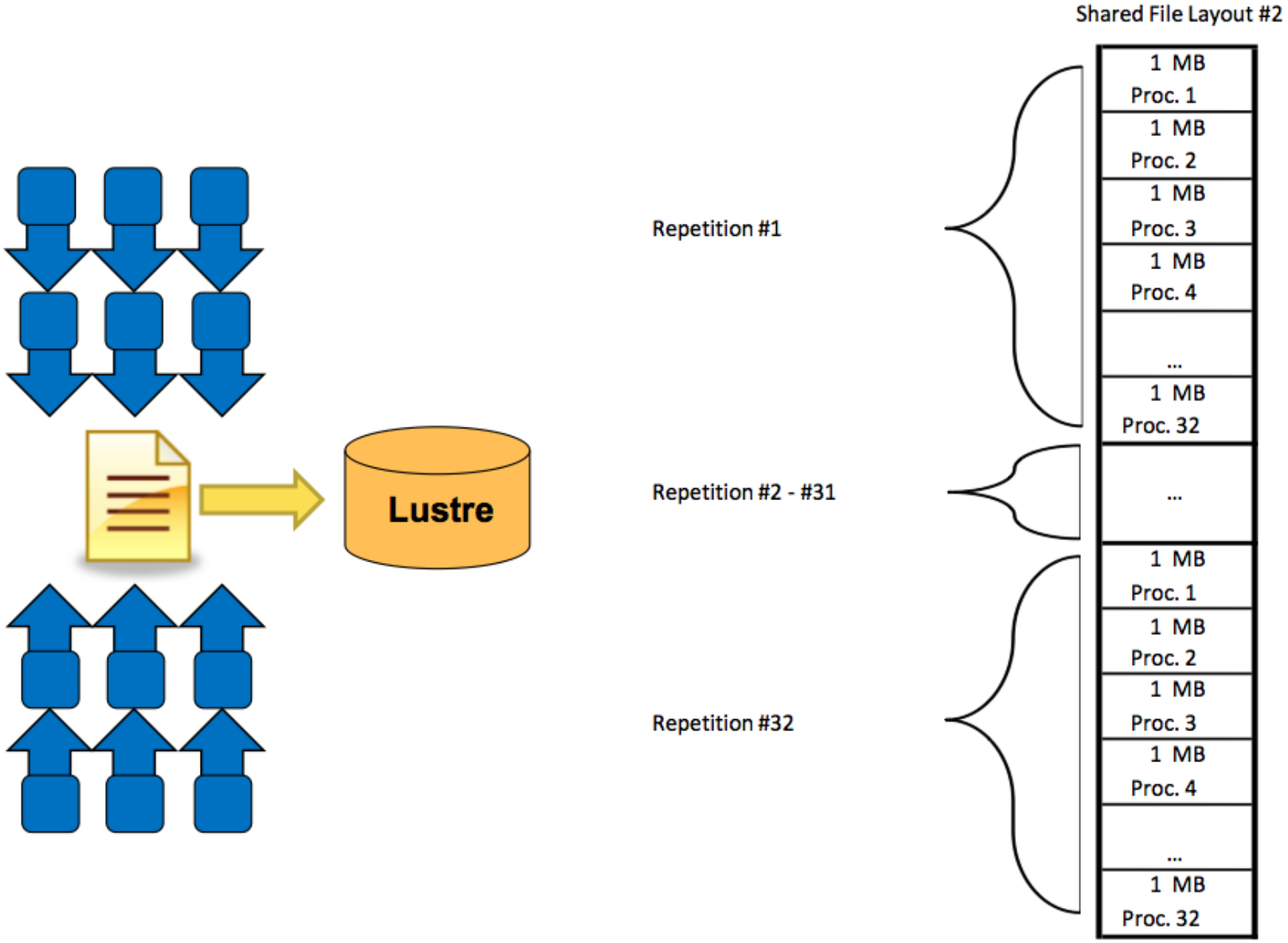


Shared File Layout #1



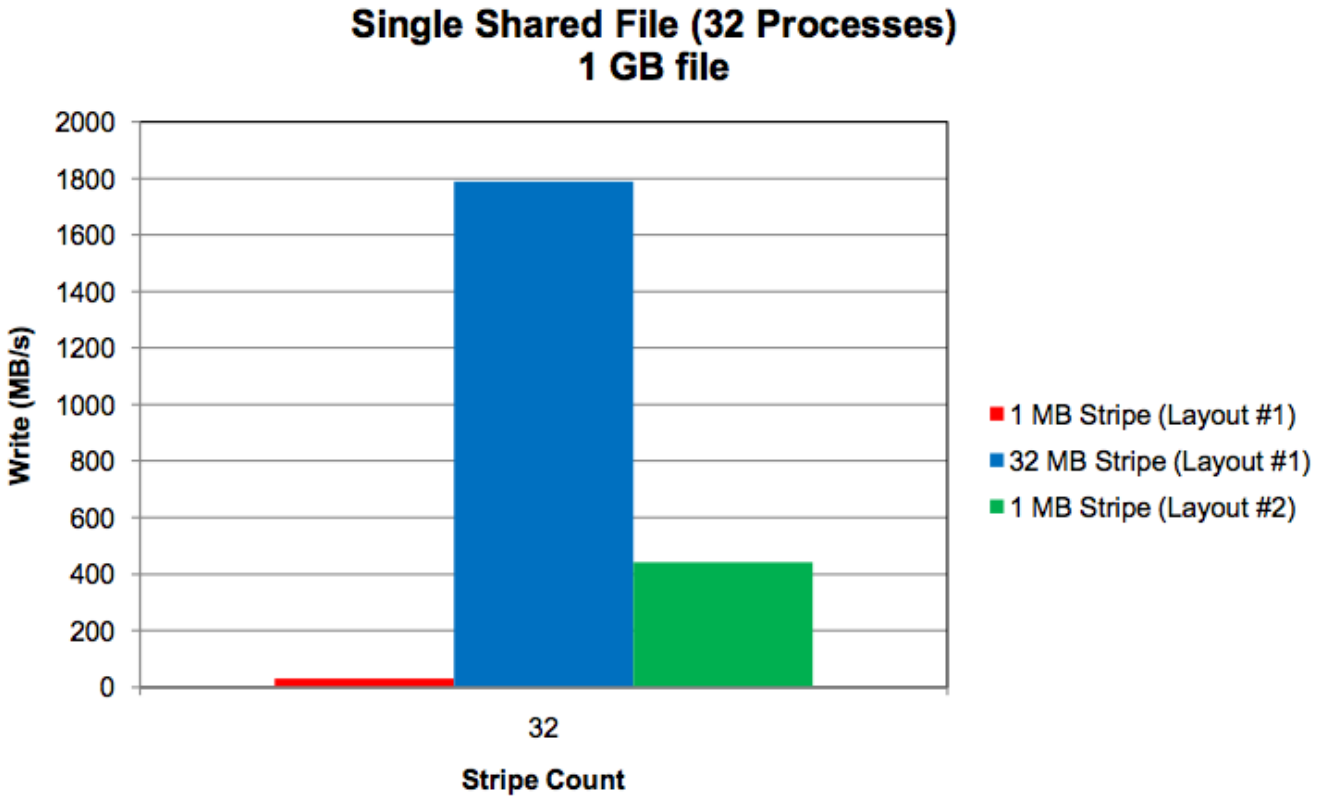
SOURCE: Lonnie Crosby, NICS

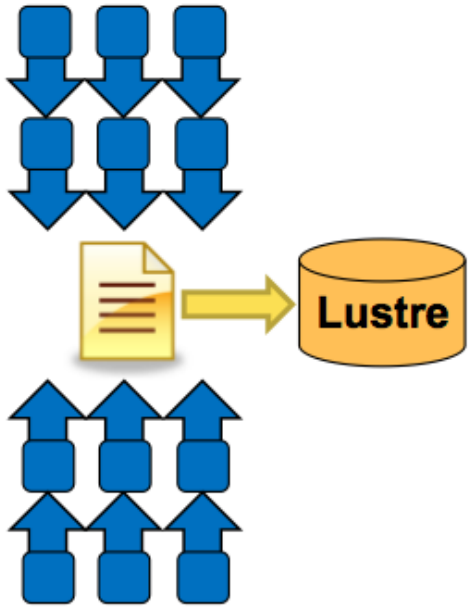
Single Shared Files and Lustre Stripes



SOURCE: Lonnie Crosby, NICS

File Layout and Lustre Stripe Pattern

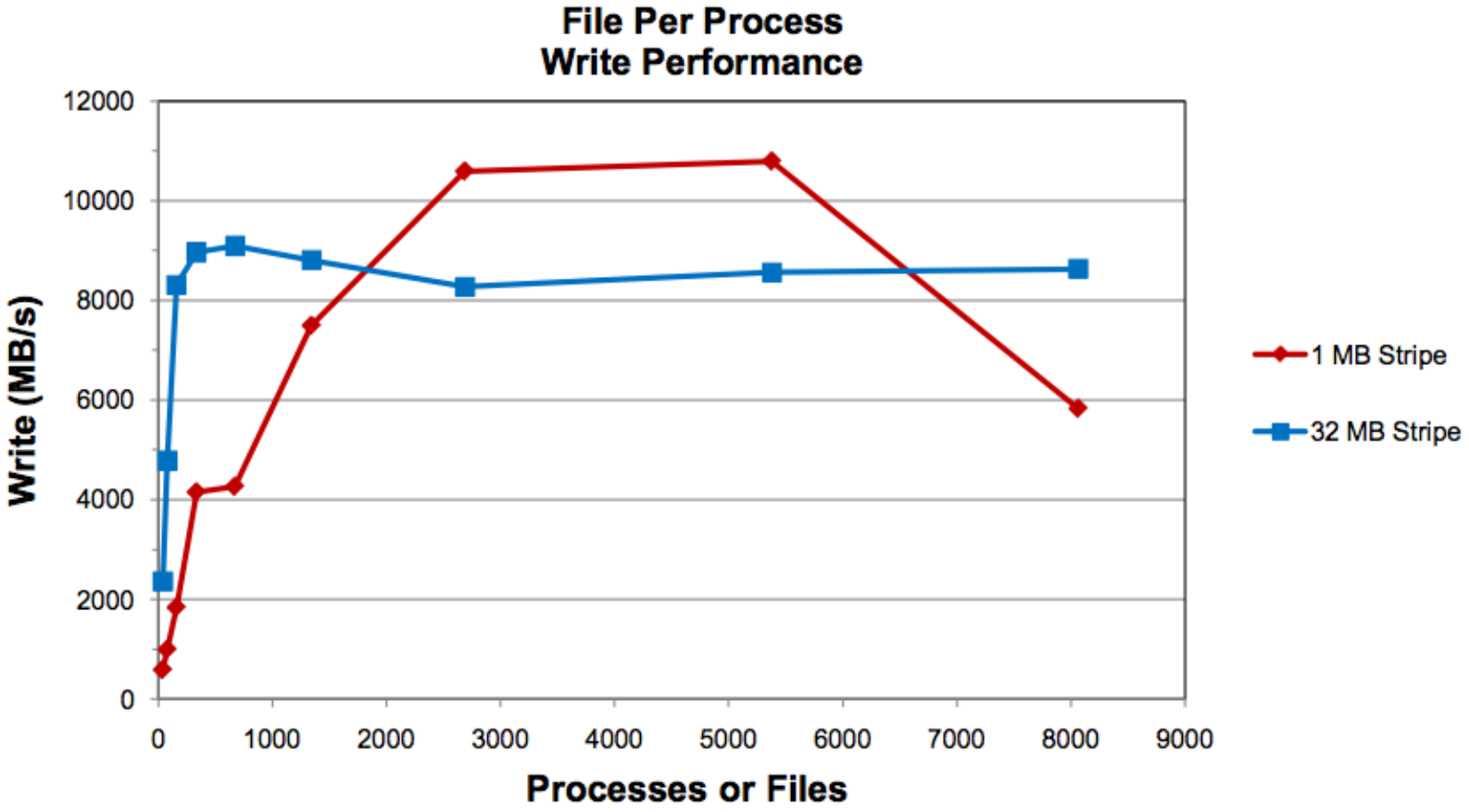




SOURCE: Lonnie Crosby, NICS

Scalability: File Per Process

- **128 MB per file and a 32 MB Transfer size**



SOURCE: Lonnie Crosby, NICS

Lustre- Performance Considerations

- Minimize metadata contention. This becomes an issue if there is I/O to too many files – typically the case when you have a file per core situation at very large scales (>10K cores).
- Minimize filesystem contention. Problem can arise in cases:
  - Shared file with large number of cores
  - File per core combined with large stripe count on each file. This might happen because the default stripe count is used without checking.
- Stripe size, stripe count.
- If possible, a process should not access more than one or two OSTs.

Lustre Commands

- Getting striping info of file/directory:

```
mahidhar@kraken-pwd2(XT5):/lustre/scratch/mahidhar> lfs getstripe test
OBDS:
0: scratch-OST0000_UUID ACTIVE
1: scratch-OST0001_UUID ACTIVE
2: scratch-OST0002_UUID ACTIVE
…
….
334: scratch-OST014e_UUID ACTIVE
335: scratch-OST014f_UUID ACTIVE
test
        obdidx                  objid                   objid                   group
            92              12018931          0xb764f3              0
            38              11744421          0xb334a5              0
           138              11679805          0xb2383d              0
            26              11896612          0xb58724              0
```

- Setting stripe parameters:

  lfs setstripe –s 1M –c 8 –i -1

  -s sets the stripe size (1 MB in this case)

  -c sets the stripe count (8 in this case)

  -i sets the stripe index start

Subsetting I/O

- **At large core counts, I/O performance can be hindered**
  - **by the collection of metadata operations (File-per-process) or**
  - **by file system contention (Single-shared-file).**
- **One solution is to use a subset of application processes to perform I/O. This limits**
  - **the number of files (File-per-process) or**
  - **the number of processes accessing file system resources (Single-shared-file).**
- **If you can not implement a subsetting approach, try to limit the number of synchronous file opens to reduce the number of requests simultaneously hitting the metadata server.**

SOURCE: Lonnie Crosby, NICS

References (for Lustre part)

- Lonnie Crosby (NICS) has detailed paper & presentation on lustre performance optimizations:

http://www.cug.org/5-publications/proceedings_attendee_lists/CUG09CD/S09_Proceedings/pages/authors/11-15Wednesday/13A-Crosby/LCROSBY-PAPER.pdf

http://www.teragridforum.org/mediawiki/images/e/e6/Lonnie.pdf

Wide Area Network (WAN) Filesystems

- A single “file system” entity that spans multiple systems distributed over a wide area network.
- Often but not always spans administrative domains.
- Makes data available for computation, analysis, visualization across widely distributed systems.
- Key usability aspect is that there is nothing special about a WAN-FS from the user perspective – no special clients, no special namespace, etc.

A Long History in TeraGrid

- First demonstration by SDSC at SC 2002.
- Numerous demonstrations at Supercomputing.
- Several production file systems past and present – currently GPFS-WAN at SDSC, DC-WAN at IU, and Lustre-WAN at PSC.
- Many TeraGrid research projects have used the production WAN file systems.
- Many TeraGrid research projects have used experimental WAN file systems.
- Continuing research, development, and production projects from 2002-2010.

WAN File System Challenges

- Security
  - Identity mapping across administrative domains
  - Control of mount access and root identity
- Performance
  - Long network latencies imply a delay on every operation
  - Appropriate node/disk/network/OS configuration on both client and server
- Reliability
  - Network problems can occur anywhere
  - Numerous distributed clients can inject problems

GPFS-WAN 1.0

- First Production WAN File System in TeraGrid
- Evolution of SC04 demo system
- 68 IA64 “DTF Phase one” server nodes
- .5 PB IBM DS4100 SATA Disks, Mirrored RAID
- ~250 TB Usable storage, ~8GB/sec peak I/O
- Still the fastest WAN-FS ever deployed in TeraGrid (30Gb/s) – network got slower afterward
- Utilized GSI “grid-mapfile” for Identity Mapping
- Utilized RSA keys w/ OOB exchange for system/cluster authentication

Use of GPFS-WAN 1.0

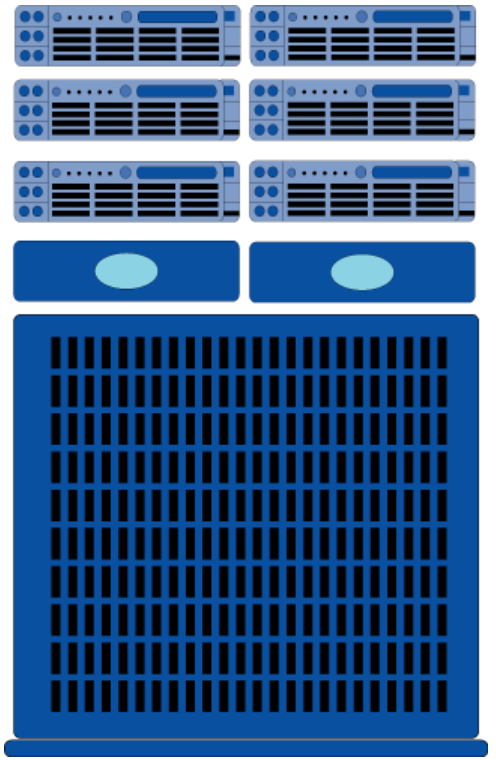
- Production in October 2005
- Accessible on almost all TeraGrid resources (SDSC, NCSA, ANL, NCAR)
- Required major testing and debugging effort (~1 year from SC 2004 demo)
- BIRN, SCEC, NVO were major early users
- Lots of multi-site use in a homogeneous computing environment (IA64/IA32)
- BIRN Workflow – compute on multiple resources, visualize at Johns Hopkins

GPFS-WAN 2.0

- In production late 2007
- Replaced all Intel hardware with IBM p575s
- Replaced all IBM Disks with DDN arrays
- Essentially everything redundant
- Capacity expanded to ~1PB raw
- Added use of storage pools and ILM features
- Remains in production 3 years later

IU's Data Capacitor WAN

- Purchased by Indiana University
- Announced production at LUG 2008
  - Allocated on Project by Project basis
- 1 pair Dell PowerEdge 2950 for MDS
- 2 pair Dell PowerEdge 2950 for OSS
  - 2 x 3.0 GHz Dual Core Xeon
  - Myrinet 10G Ethernet
  - Dual port Qlogic 2432 HBA (4 x FC)
  - 2.6 Kernel (RHEL 5)
- DDN S2A9550 Controller
  - Over 2.4 GB/sec measured throughput
  - 360 Terabytes of spinning SATA disk
- Currently running Lustre 1.8.1.1



DC-WAN Applications

- Wide range of applications and domains
- Several projects spanning both TeraGrid and non-TeraGrid resources
- Utilized as a simple “bridge” to bring data into TG
- Has also been used for transatlantic mount to Germany
- Diverse range of systems with DC-WAN lends itself to use in workflows

Lustre-WAN 2.0 at PSC

- J-WAN – Josephine Palencio
  - Support use of Kerberos for identity mapping and user authentication
  - Potentially very convenient for management of user identities and authorization
  - Kerberos is well-accepted, widely used
  - Many other valuable features of Lustre 2.0
- Successful tests with storage at PSC and SDSC, client mounts at several TeraGrid sites

Lustre-WAN 2.0 History

- Systems have been operational for over 2 years
- Successful tests have been done with distributed storage at PSC and SDSC
- Work is ongoing to improve, harden Kerberos and other features of Lustre 2.0
- Still pre-release, but expected to appear late this year

TG-Wide Lustre-WAN

- Lustre 1.8 now supports distributed storage
- Storage nodes can be co-located with compute, vis resources for local access to data
- 6 Sites installing storage, 1PB total usable
- Will use Indiana's UID-mapping mechanism
- Almost all TeraGrid resources are now compatible
- Single namespace and access mechanism will make data on Lustre-WAN near ubiquitous in TeraGrid
- Planned for production October 1 2010

Data Transfer: It often starts as a trickle!

- scp/sftp sufficient for simple data transfers – codes, small input datasets etc.
- Windows clients available – putty, ws-ftp etc.
- Globus enabled client (gsiscp) can make it easier on the authentication front (for TG).
- Data transfer is encrypted. Not multithreaded. SSH is network performance limited by statically defined internal flow control buffers.
- High performance SSH/SCP (hpn-ssh from PSC) overcomes these issues.

High Performance SSH (hpn-ssh)

- User can manually set the TCP buffer size.
- Enable buffer sizing to work with buffer auto-tuning (if supported by the end host)
- Can turn off data encryption.
- Available on most TeraGrid hosts with gsi authentication. You can confirm by typing *gsissh -v* and looking for *-hpn* in the version string. For example on Ranger:

  login3% gsissh -v

  OpenSSH_5.0p1-hpn13v1 NCSA_GSSAPI_GPT_4.3 GSI, OpenSSL 0.9.7d 17 Mar 2004

- More details at:
  - http://www.psc.edu/networking/projects/hpn-ssh/
  - http://www.psc.edu/networking/projects/hpn-ssh/faq.php

High Performance SSH Example

- Example of transfer between Ranger and Kraken.
- First without HPN:

  mahidhar@kraken-pwd3(XT5):/lustre/scratch/mahidhar> /usr/bin/scp ranger.tacc.utexas.edu:/work/00342/mahidhar/had.tar ./had.tar

  had.tar 100% 264MB 1.8MB/s 02:27

- HPN with auto TCP buffer size:

  mahidhar@kraken-pwd3(XT5):/lustre/scratch/mahidhar> gsiscp ranger.tacc.utexas.edu:/work/00342/mahidhar/had.tar ./had.tar

  had.tar 100% 264MB 14.7MB/s 00:18

- HPN with auto TCP buffer size, data encryption turned off:
  - mahidhar@kraken-pwd3(XT5):/lustre/scratch/mahidhar> gsiscp -oNoneEnabled=yes -oNoneSwitch=yes ranger.tacc.utexas.edu:/work/00342/mahidhar/had.tar ./had.tar
  - WARNING: ENABLED NONE CIPHER
  - had.tar 100% 264MB 33.0MB/s 00:08

Multi-Stream Transfers (BBCP, BBFTP)

- BBCP - peer-to-peer network file copy application.
  - No server process is required
  - All standard methods of authentication can be used: passwords and certificates
  - Multiple streams, can use all cores, performance approaches line speed (if enough streams are used and if enough memory is available).
- BBFTP
  - multi-stream file transfer application
  - Data transfer unencrypted
  - All standard methods of authentication can be used: passwords and certificates

BBCP (Options)

- Important Options:

-w	sets the size of the disk I/O buffers; The TCP/IP socket buffer is set to wsz plus 32 bytes to account for network overhead. This effectively sets the TCP/IP window size for the associated connection. The default is 64k.
-s	sets the number of parallel network streams. Default is 4.

- Detailed documentation:

http://www.slac.stanford.edu/~abh/bbcp/

BBFTP (Options)

- Important Options:

  setrecvwinsize `` WindowSize"

  Set size in Kbytes of the receive TCP window of each stream of the bbftpd daemon. This also set the send window size of the client to the same value.

  setsendwinsize `` WindowSize"

  Set size in Kbytes of the send TCP window of each stream of the bbftpd daemon. This also set the receive window size of the client to the same value.

  -p NumberOfParallelStreams

  sets the number of parallel network streams. Default is 1.

- Detailed documentation:

  http://doc.in2p3.fr/bbftp/3.2.0.bbftp.html

Multi-Stream Transfer Example

First with HPN-SSH, encryption turned off:

```
mahidhar@kraken-pwd3(XT5):/lustre/scratch/mahidhar> gsiscp -
    oNoneEnabled=yes -oNoneSwitch=yes
    ranger.tacc.utexas.edu:/work/00342/mahidhar/hd1.yuv ./hd1.yuv
WARNING: ENABLED NONE CIPHER
hd1.yuv                                   100% 3983MB  33.5MB/s   01:59
```

BBCP:

```
mahidhar@kraken-pwd3(XT5):/lustre/scratch/mahidhar> bbcp -P 2 -w
    1M -s 8 ranger.tacc.utexas.edu:/work/00342/mahidhar/hd1.yuv
    ./hd1.yuv
bbcp: Source I/O buffers (24576K) > 25% of available free memory
    (58860K); copy may be slow
bbcp: Creating ./hd1.yuv
bbcp: At 100727 20:06:38 copy 99% complete; 41606.0 KB/s
```

GridFTP – Multithread, striped over multiple hosts

- BBCP, BBFTP can maximize the transfer rate from a single host. Typically, this is limited by the network bandwidth out of one host (typically 1 Gigbit).
- GridFTP (globus-url-copy, tgcp, uberftp) can stripe over several hosts and also use multiple threads on each host.
- Transfer between high performance parallel filesystems (lustre, GPFS-WAN) to maximize throughput.
- Measured rates as high as 750MB/s between GPFS-WAN and lustre on Kraken.
- gsi certificate based authentication. Third party transfers are feasible.

GridFTP- Example

- Using GridFTP, no striping, multithreaded:

```
mahidhar@kraken-pwd3(XT5):~> globus-url-copy -vb -fast -tcp-bs 8M -p 8
    gsiftp://gridftp.ranger.tacc.teragrid.org///work/00342/mahidhar/hd1.yuv
    file:///lustre/scratch/mahidhar/hd1.yuv
Source: gsiftp://gridftp.ranger.tacc.teragrid.org///work/00342/mahidhar/
Dest:   file:///lustre/scratch/mahidhar/
  hd1.yuv
   4139778048 bytes       109.66 MB/sec avg       97.85 MB/sec inst
```

- Using GridFTP, striping, multithreaded:

```
mahidhar@kraken-pwd3(XT5):~> globus-url-copy -vb -fast -stripe -tcp-bs 8M -sbs 0 -
    p 8 gsiftp://gridftp.ranger.tacc.teragrid.org///work/00342/mahidhar/hd1.yuv
    gsiftp://gridftp.nics.teragrid.org//lustre/scratch/mahidhar/hd1.yuv
Source: gsiftp://gridftp.ranger.tacc.teragrid.org///work/00342/mahidhar/
Dest:   gsiftp://gridftp.nics.teragrid.org//lustre/scratch/mahidhar/
  hd1.yuv
   4176230400 bytes       393.97 MB/sec avg       200.92 MB/sec inst
```

GridFTP – Important Links

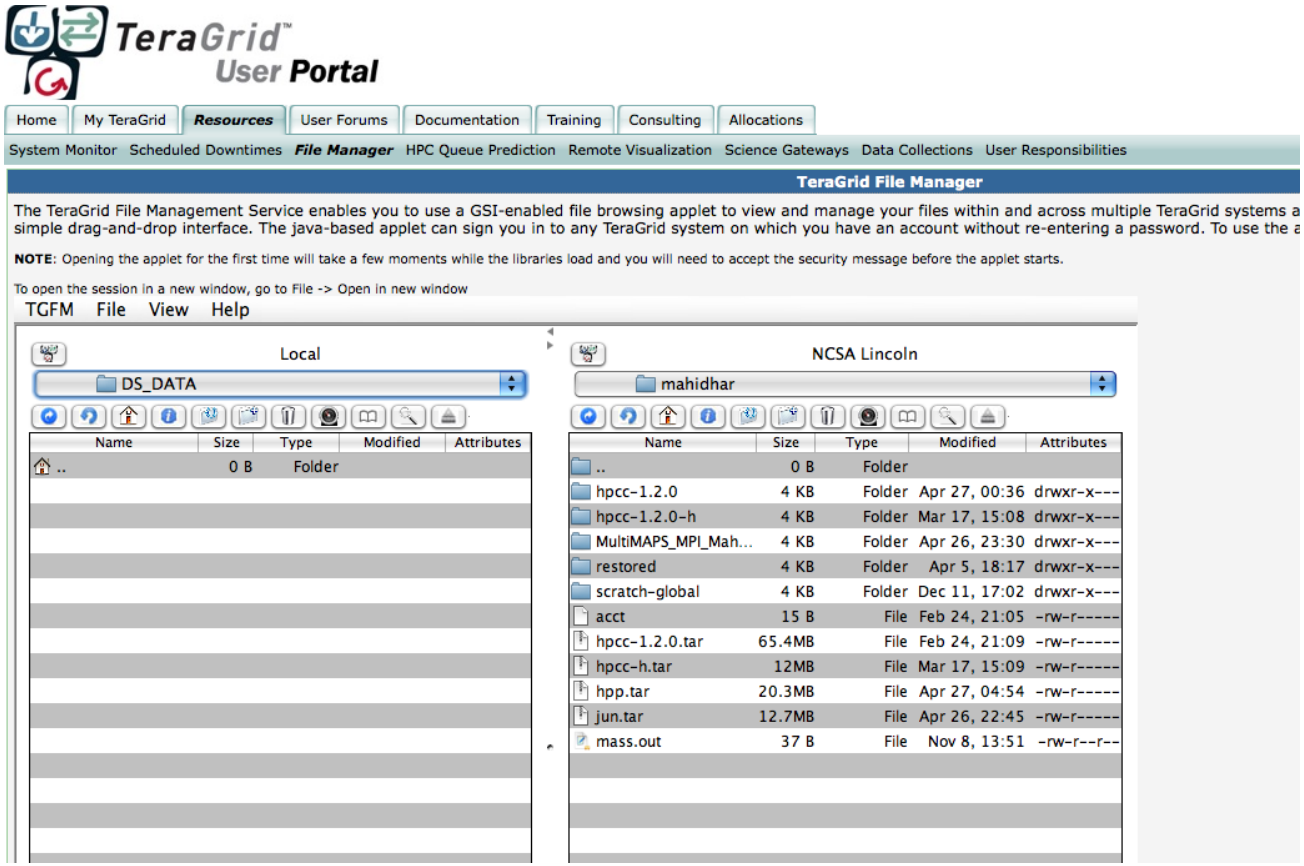
- TeraGrid GridFTP page:
  https://www.teragrid.org/web/user-support/gridftp

- TeraGrid GridFTP server info:
  https://www.teragrid.org/web/user-support/transfer_location#deployment

- TeraGrid client toolkit for Mac, Linux:
  https://www.teragrid.org/web/user-support/sso_tg_client_toolkit

File Transfer using TG User Portal

- TeraGrid File Manager uses gsi enabled file browsing applet to view and manage files across multiple TeraGrid sites.
- Simple drag and drop interface to move files/directories.



Data Management – Post Processing

- Large scale runs can produce tens of terabytes of data. Typically the only place to keep this data for immediate post processing is the parallel filesystems (e.g. /lustre/scratch on Kraken, lustre based scratch, work on Ranger).
- Typically cannot keep the data online for long term (due to space restrictions, purging) => first step should be to archive any important data (e.g. to HPSS at NICS/Kraken, Ranch at TACC/Ranger).
- **Important to verify for any critical archived data** – the only way to be sure is via checksums (compare between original, archived, retrieved data). This can be very expensive … can do it in parallel if a lot of files are involved.
- At extremely large scales it is advisable to integrate the post processing with the computational run (if possible).

Data Management – Projects Space

- Projects space useful for sharing information within research group/community. This can be on NFS filesystems or parallel filesystems depending on need. Typically available on most TeraGrid machines.
- Data Capacitor (IU) and GPFS-WAN (SDSC) are another option to support research datasets.
- Size of the filesystems => typically not backed up! **Users should archive any important data!**
- TeraGrid Share is a new service providing users with 2GB of secure, personal space for collaborating with other TeraGrid users and the world. Both the TGUP File Manager and the Mobile TGUP provide access to this space.

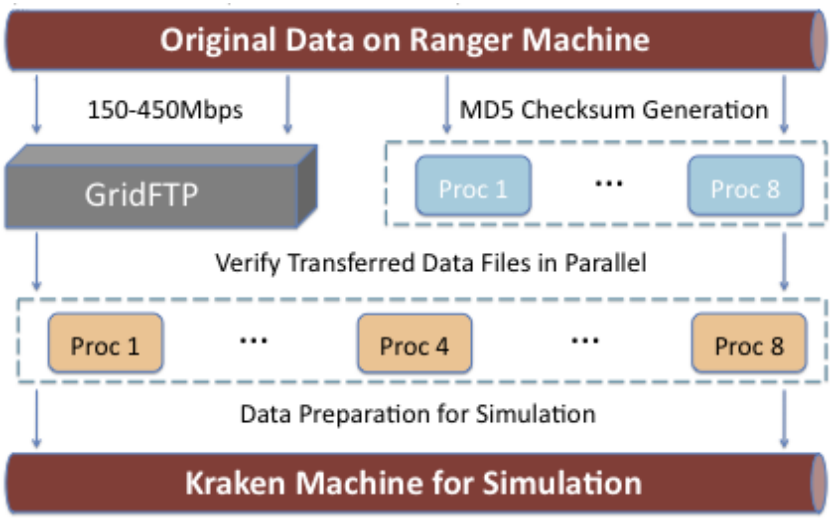
Data Management - Archive

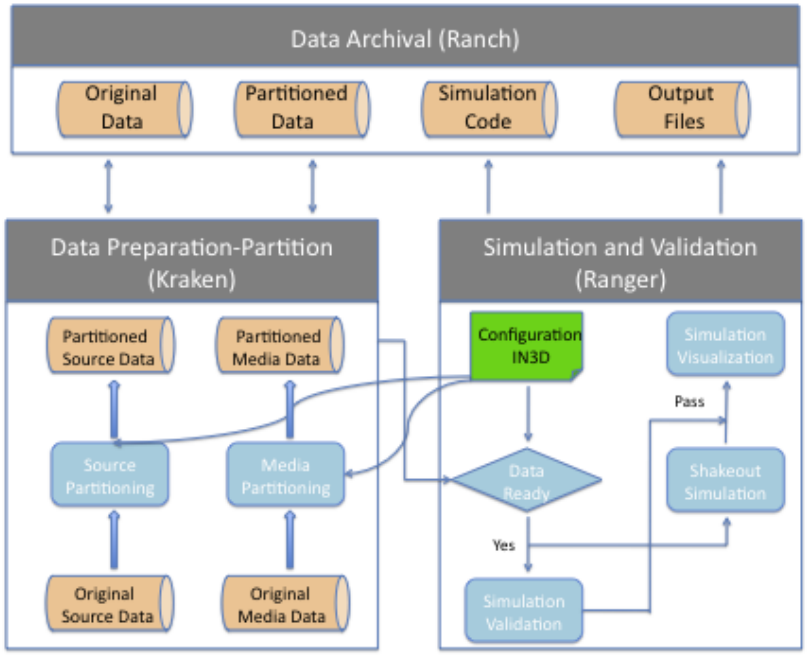
- Sites typically have local tape archives. Any important files should be archived asap.
- Tapes can go bad! If something is very important, replicated backup at a different site is recommended.
- TeraGrid offers a distributed replication and data management service, utilizing archive and disk systems at multiple sites (IU, NCSA, PSC, Purdue, SDSC, TACC). Access via iRODs. Details at:

https://www.teragrid.org/web/user-support/data_resources

Data Workflow Example: SCEC Data Archive and Management

- ❑ Execute different jobs on different TeraGrid machines and control all execution sequences remotely
- ❑ Validate correctness of input and output data and detect errors occurred during the simulation process and recover them automatically
- ❑ High performance data transfer using GridFTP
- ❑ 90k–120k files per run, 150TBs organized as a separate sub-collection in iRODs, direct transfer using iRODs from Ranger to SDSC SAM-QFS up to 177 MB/s using our data ingestion tool PiPUT





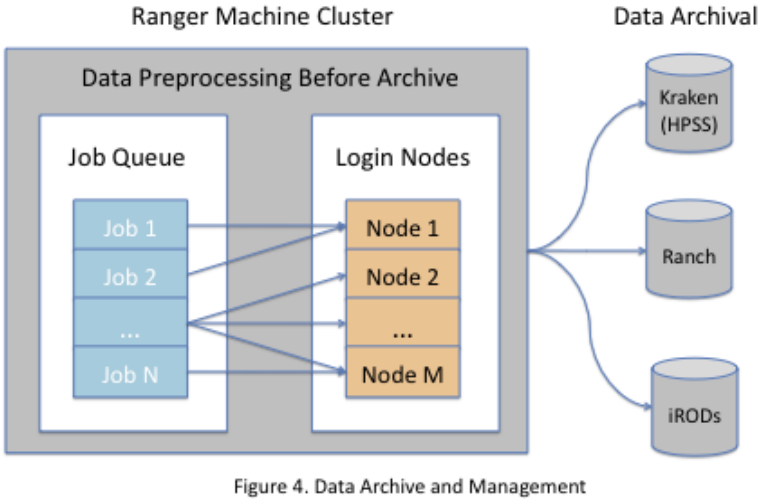


Figure 4. Data Archive and Management

(Source, Y. Cui, SDSC,2010)

Thank You

Questions?

Email mahidhar@sdsc.edu