Sequoia Data Migration Experiences

Lustre User Group, 2013. San Diego, CA

April 18, 2013

Marc Stearman

Parallel File Systems Operations Lead

Lawrence Livermore National Laboratory

LLNL-PRES-635113

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

Overview

- Requirements
  - Move User Data before transition
  - 2 – 3 week timeline
- Sequoia Lustre SAN layout
  - bottlenecks, number of clients
- Available Tools
  - tar, rsync, cp, etc.
- Methodology
- Next Steps

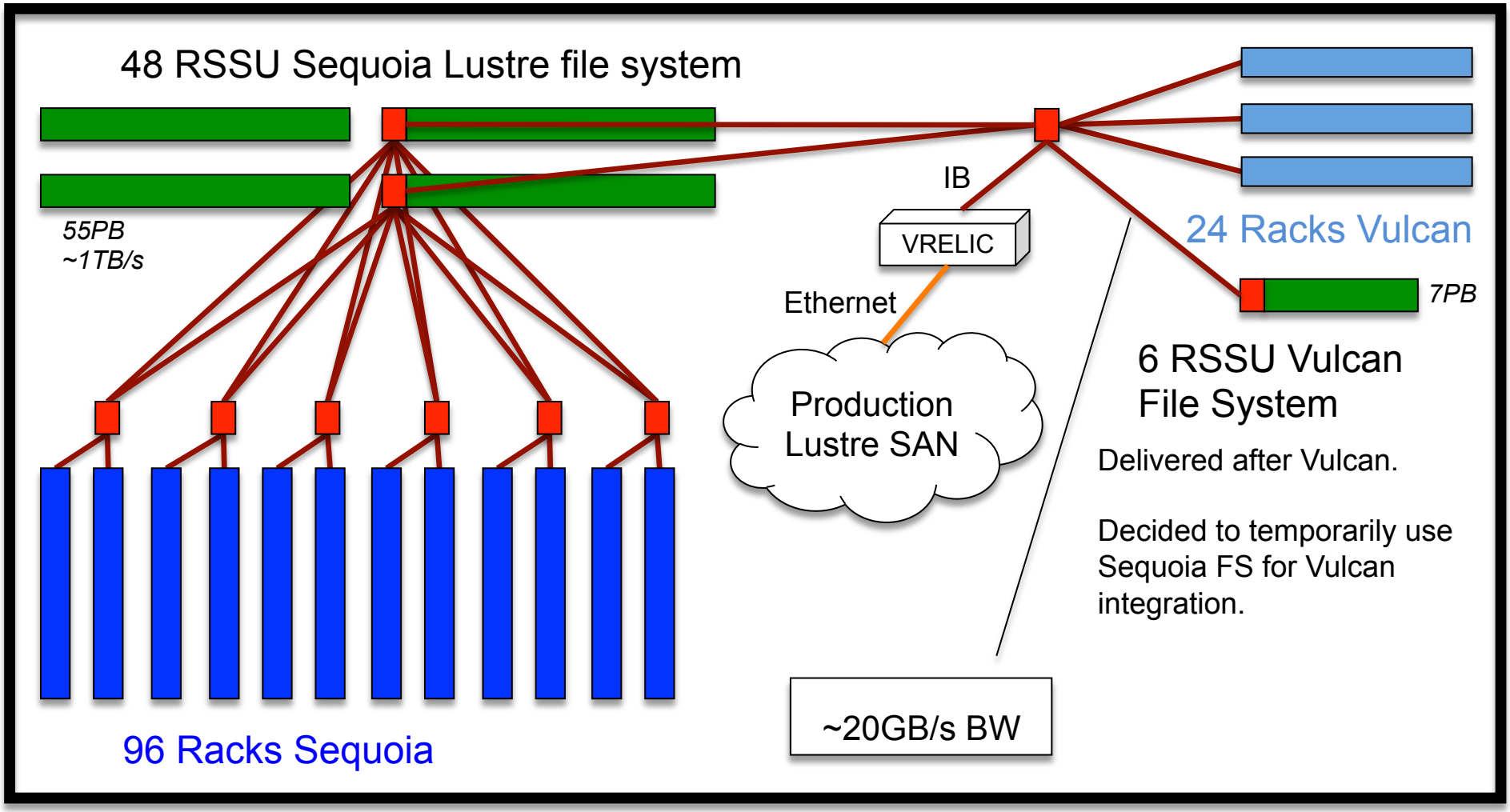
Requirements, or “You want me to do what?”

- Sequoia is destined for secure computing facility, but integrated in our collaboration zone.
- Users doing scaling studies and science runs generated 800TB of block data on the file system using 200 million files.
- Someone (*me*) decided it would be easier if we just moved the users data to avoid last minute rush and keep our deadlines.

Requirements, or “What more do you want me to do?”

- We had about 2 – 3 weeks to move the data.
- At 20GB/s it should only take 16 hours. No big deal right?
- After announcing the plan to users, they wrote another 400TB of block data and another 175 million files.
- With ZFS compression we were seeing 1.7x compression ratio. The real size of the data was 2PB.

Sequoia SAN Layout, or "What are all these cables for?"



What tools exist to move PetaBytes of data?

- At the start, I had 7 clients, each with a single QDR IB link: ~20GB/s of network bandwidth
- cp, tar, rsync, pax – all single client utilities
  - tar --xattrs flag preserves lustre striping
  - rsync --xattrs flag had issues with trusted.* attrs when run as root
  - both tools complained about xattrs on directories
- I recalled hearing about some parallel tools from prior LUG presentations

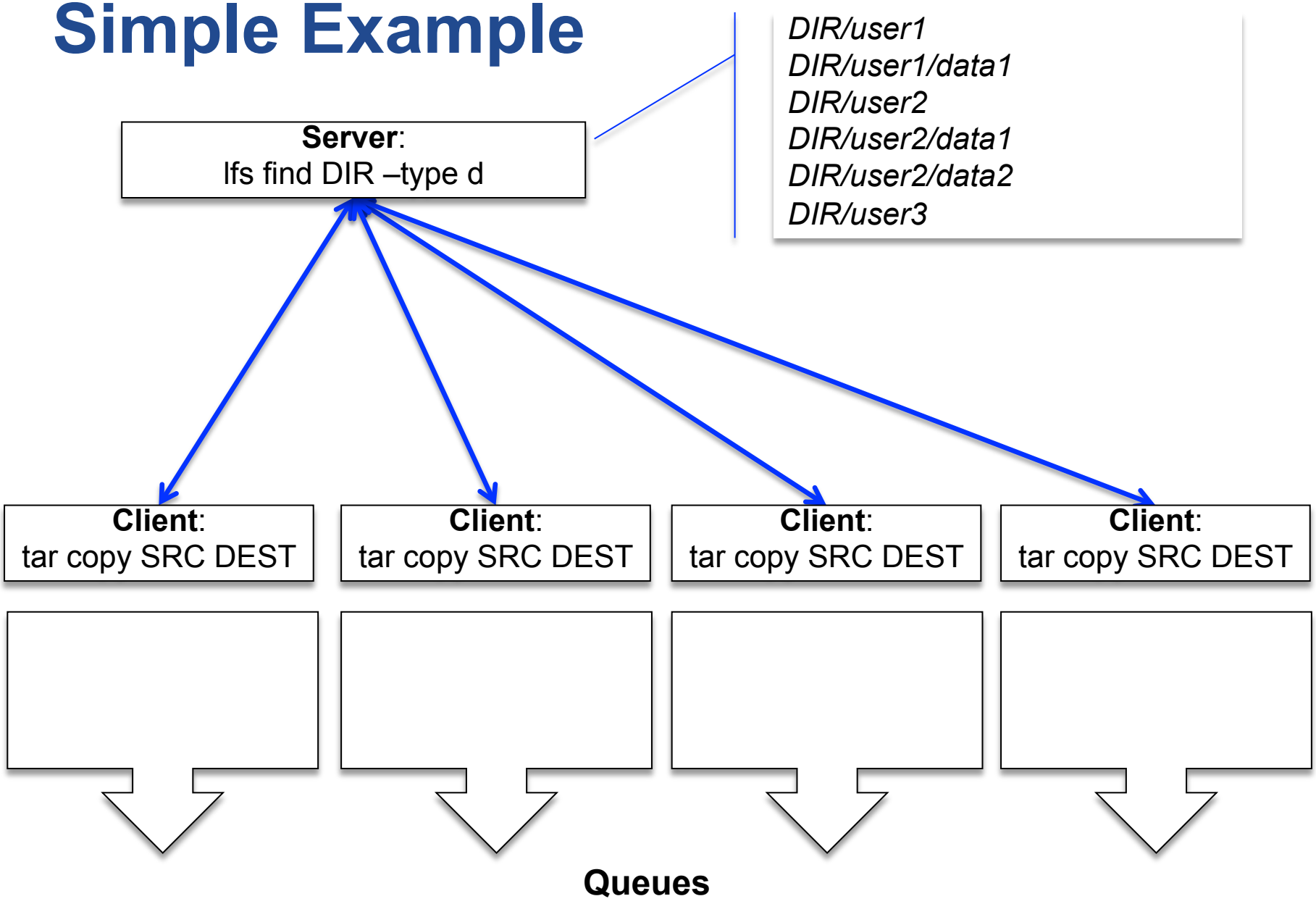
Parallel Copying Tools

- mutil 1.76.6 – http://mutil.sourceforge.net/
  - A patchset for coreutils 7.6
  - LLNL is RHEL6.2 based, with coreutils 8.4
  - I spent a day tracking down coreutils 7.6 but could not get it to build properly on our systems.
- dcp – http://filecopy.org/
  - Looked promising, and was able to get it to build
  - Lacked functionality with unimplemented features
    - change ownership, permissions, timestamp not implemented

Grow your own

- When in doubt, and lacking in time, fall back to the traditional client/server approach
- Start a server listening on a socket that does a find:
  - `lfs find $DIR -type d`
  - Server could also use a list of directories in a file
- Each client connects to the server asking for a single directory
- Each client copies all contents of that directory from one file system to another

Simple Example



Multiple passes

- First pass, use tar to copy directories
  - `tar -cpf - --no-recursion --xattrs "$DIR"/* | ( tar -C /p/lscratchv -xpvf - )`
  - This preserves striping nicely, but has some issues with large numbers of files in directories. Also deep directories suffered from MAX_ARGS command line length limit
  - Creates missing directories in the path as needed
  - Doesn't remember where it left off. Interruptions and errors force copying data a second time. Only good for the first pass

Multiple passes (*did I do this already?*)

- After some interruptions use rsync
  - `rsync -lvptgoDd --inplace $dir/ /p/lscratchv/$dir/`
  - Very metadata intensive
  - Originally did not use the `--inplace` flag but noticed rsync copies data to a temp file, then does a rename
  - Tried `--xattrs` but did not preserve striping and did not copy directories (errors on trusted.lov, which seems to be a regression in 2.3)
  - rsync doesn't handle missing directories like tar, so multiple passes are needed when only doing one directory at a time (or need to pre-create all directories)

Scaling it up

- Data was trickling in with only 7 clients
  - 300-500 MB/s
  - would take a month or more at that rate
- “Borrowed” Sequoia I/O Nodes for a bit
  - 5-10 GB/s using 768 I/O nodes as clients
- Eventually ran out of block data
  - Bound by number of files
  - At ~100-200 files/sec, it would take over a month
- Scale it back a bit
  - Copy less files; Received exclude patterns from users

Now what?

- Tried to get some sleep but kept dreaming about improvements
- Ran a final scan looking for *root* owned files left over from interruptions
- Been talking with others about collaborations on building a parallel toolset
- Need to improve Metadata rates
- Total data copied: 1PB, 128 million files

Discussion;
Let's Talk and Improve the Tools!

The spice must flow!

Marc Stearman
stearman2@llnl.gov