OPENFABRICS ALLIANCE

13th ANNUAL WORKSHOP 2017

Objects over RDMA

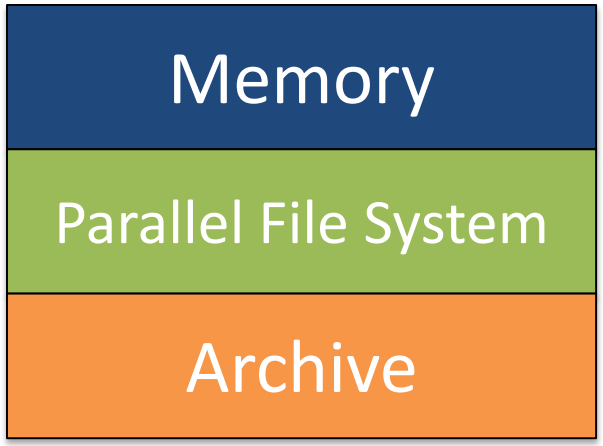
Jeff Inman, Will Vining, Garret Ransom

Los Alamos National Laboratory

March, 2017

The Problem

- **PFS: Sufficient BW implies overprovisioned capacity**
- **Tape: Expensive BW**



9.1 PB/sec
Residence – hours
Overwritten – continuous

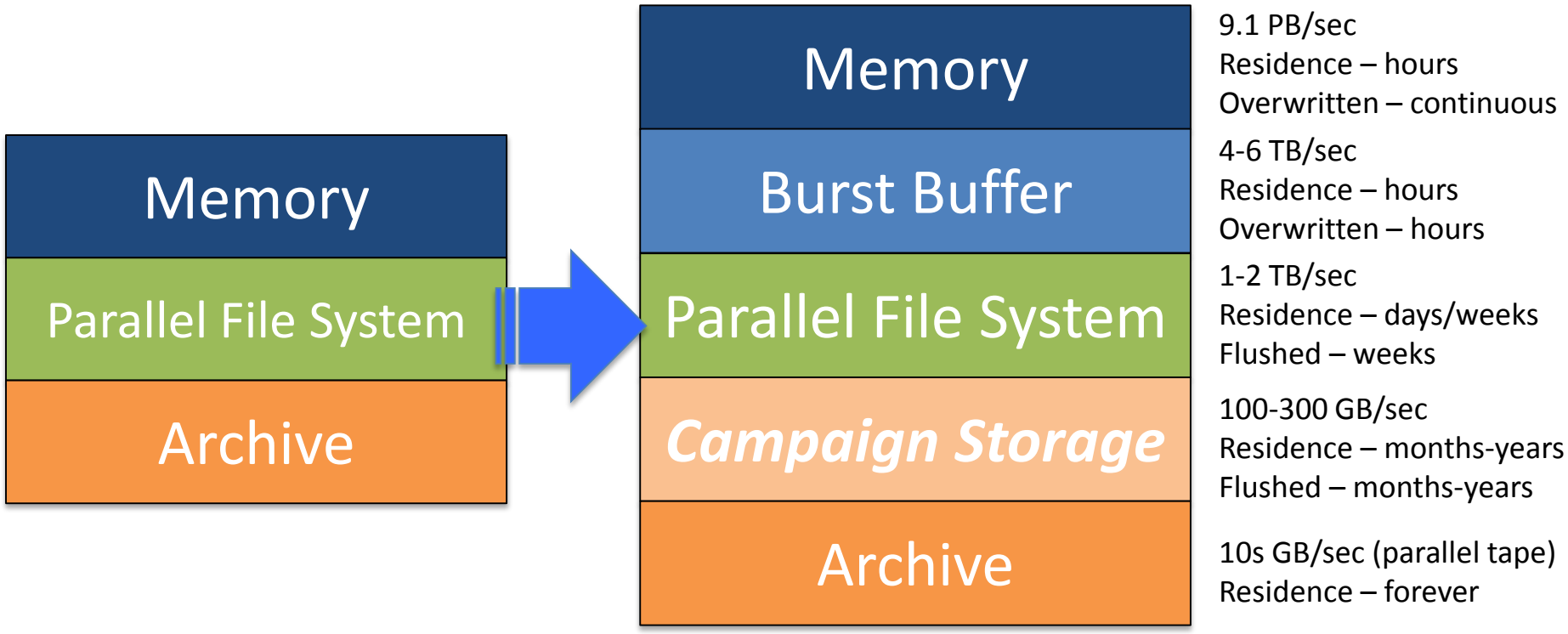
1-2 TB/sec
Residence – days/weeks
Flushed – weeks

10s GB/sec (parallel tape)
Residence – forever

Thanks: Dave Bonnie, Kyle Lamb

The Solution

- **More Layers!**



Thanks: Dave Bonnie, Kyle Lamb

Why existing solutions don't work

- **We need a high-capacity, reasonable-bandwidth storage tier**
  - Parallel tape is hard and expensive
  - Object solutions?
  - Big POSIX expensive, $$$ hardware

- **Existing solutions don't make the right compromises (for us)**
  - Petabyte scale files, and bigger
  - Billions of "tiny" files
  - Try to maintain too much of POSIX, this leads to complicated schemes, too many compromises

Thanks: Dave Bonnie, Kyle Lamb

Campaign Storage

Mar FS

- **What we need:**
  - Users see POSIX-like MetaData
    - files/directories, access-control, etc
  - Resilience
  - Economy
  - Scalable capacity and BW
  - Scalable MD
    - Single files of PB or more
    - Trillions of “tiny” files
    - Billions of files in a directory

- **Willing to trade:**
  - Universal instantaneous consistency
  - Random write access / update-in-place
  - Specific data-storage technology

Storage Tiers
(another view)

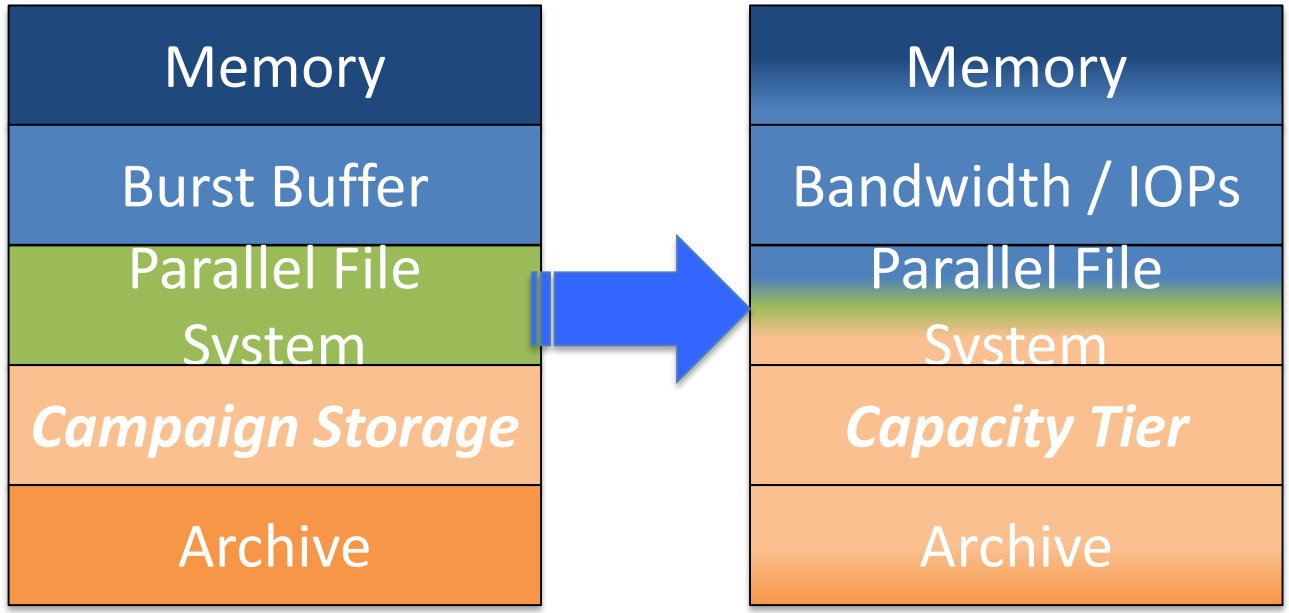
Tier	lifespan	bandwidth	change	capacity	change
RAM	hours	9.1 PB/s		2 PB	
BurstBuffer	hours	4 TB/s	x0.0005	4 PB	x2
Lustre	weeks	1.2 TB/s	x0.3	100 PB	x25
Campaign	year(s)	30 GB/s ++	x0.025	30 PB ++	x0.3
Tape	forever	10 GB/s	x0.33	60 PB ++	x2

- **"The spice must flow!"**
  - Checkpoints at BurstBuffer BW allow supercomputers to stay busy
  - Concurrent transfer of subset of elements from parent tier to child
  - Computing "campaign" can complete without having to go to tape
  - Tape only needed for true long-term archive

Is This Too Many Layers?

Perhaps we evolve toward:

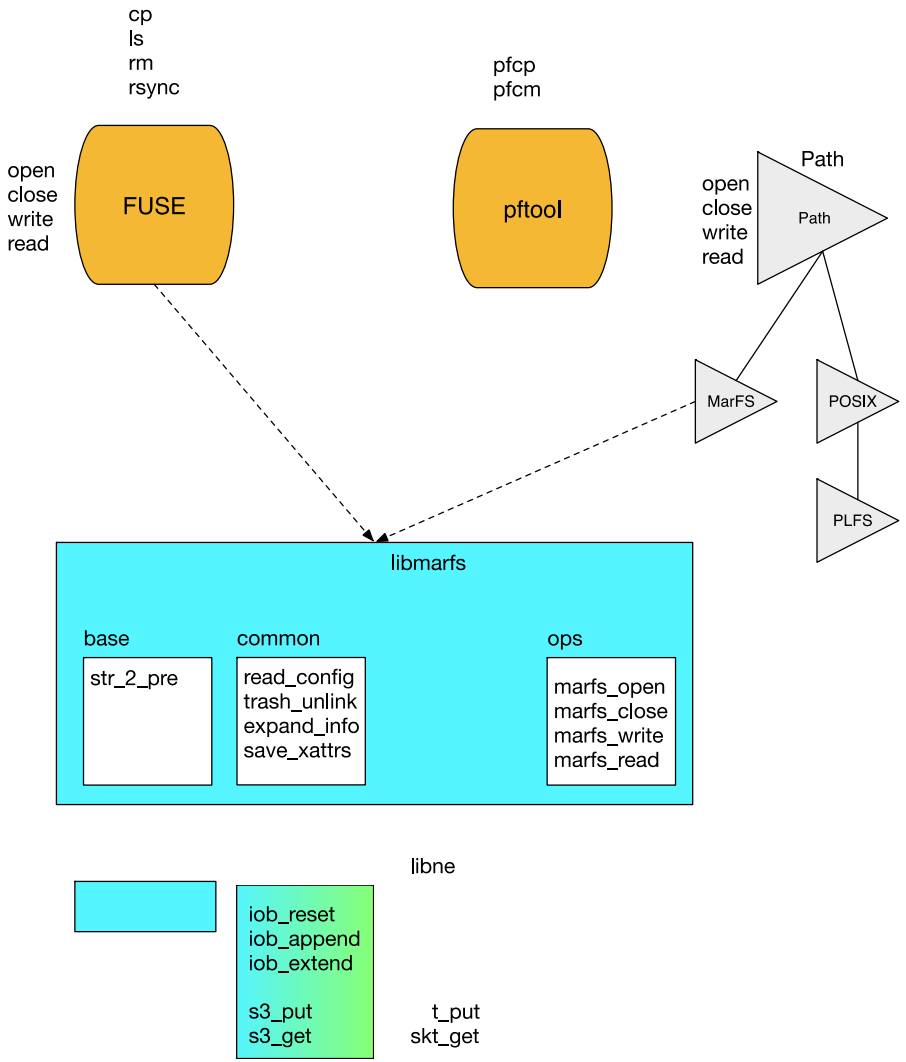
- **BW / IOPS tier**
  - In-system NVM
  - NVMe
  - SSD
  - etc
- **Capacity tier**
  - Varied degrees of EC
  - power mgt
  - off-site



Thanks: Dave Bonnie, Kyle Lamb

Applications and Libraries

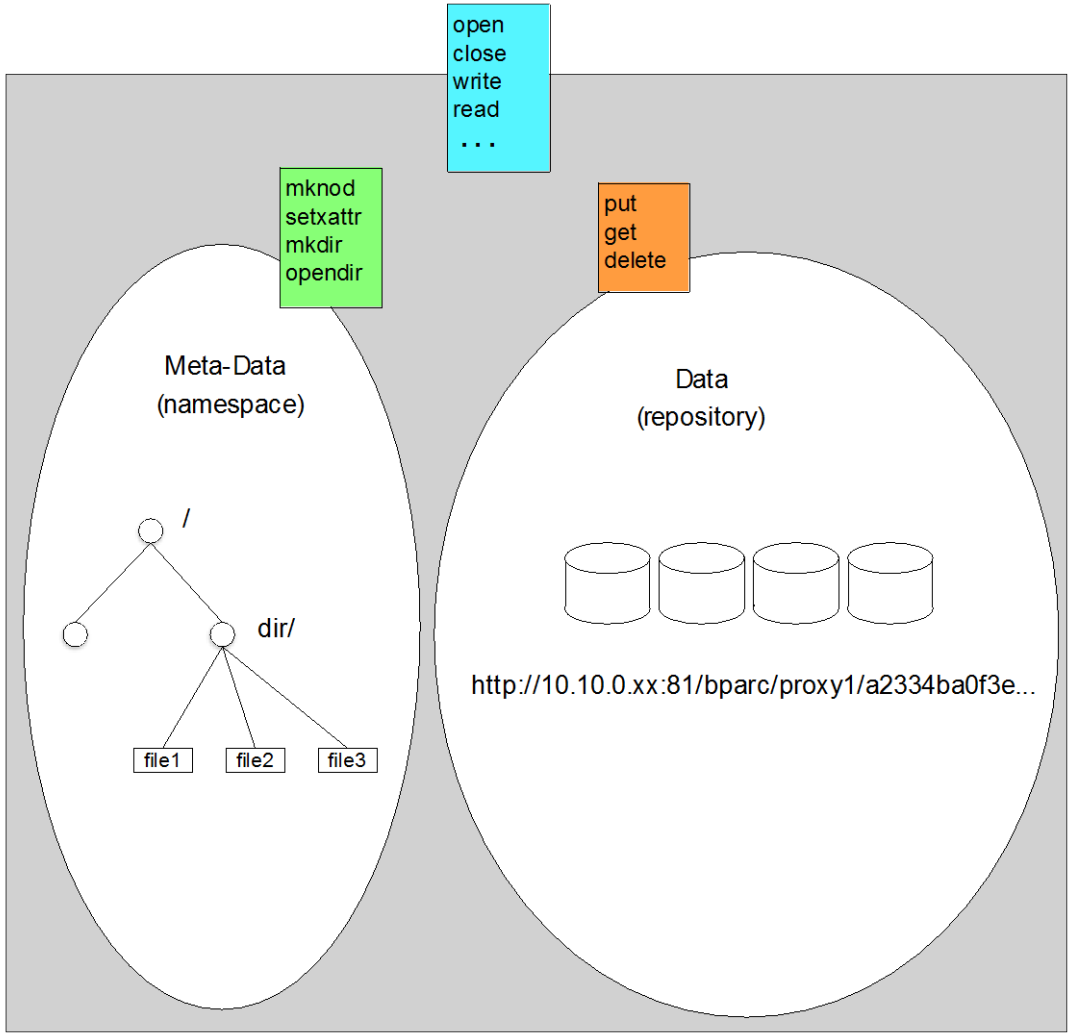
- **Users see MD through FUSE**
  - Admins can enable/disable MD/Data access
  - chown, chmod, mv, rm, etc
- **Parallel data-movement: *pftool***
  - parallel tool transfers between file-systems
- **Serial data-movement: *pipe_tool***
  - small amount of data
- **Admin tools**
  - garbage-collection
  - quota-usage update
  - etc



Separation of Concerns

MetaData / Data

- **MetaData**
  - MetaData ops (mkdir, setxattr, readdir, …)
  - POSIX files and directories
  - Truncated to size
  - System-controlled xattrs provide object-ID, etc
- **Data**
  - Data ops (put/get/delete)
  - Transparent resilience



Internal File Types

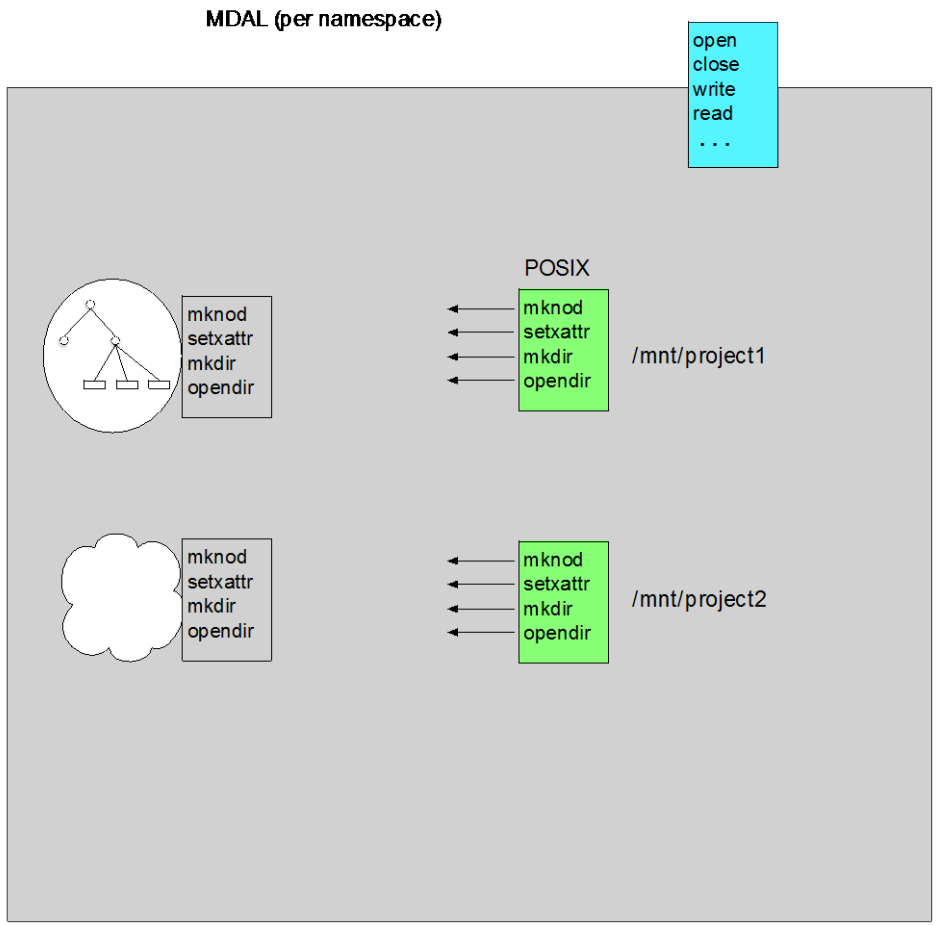
- **DIRECT (legacy files)**

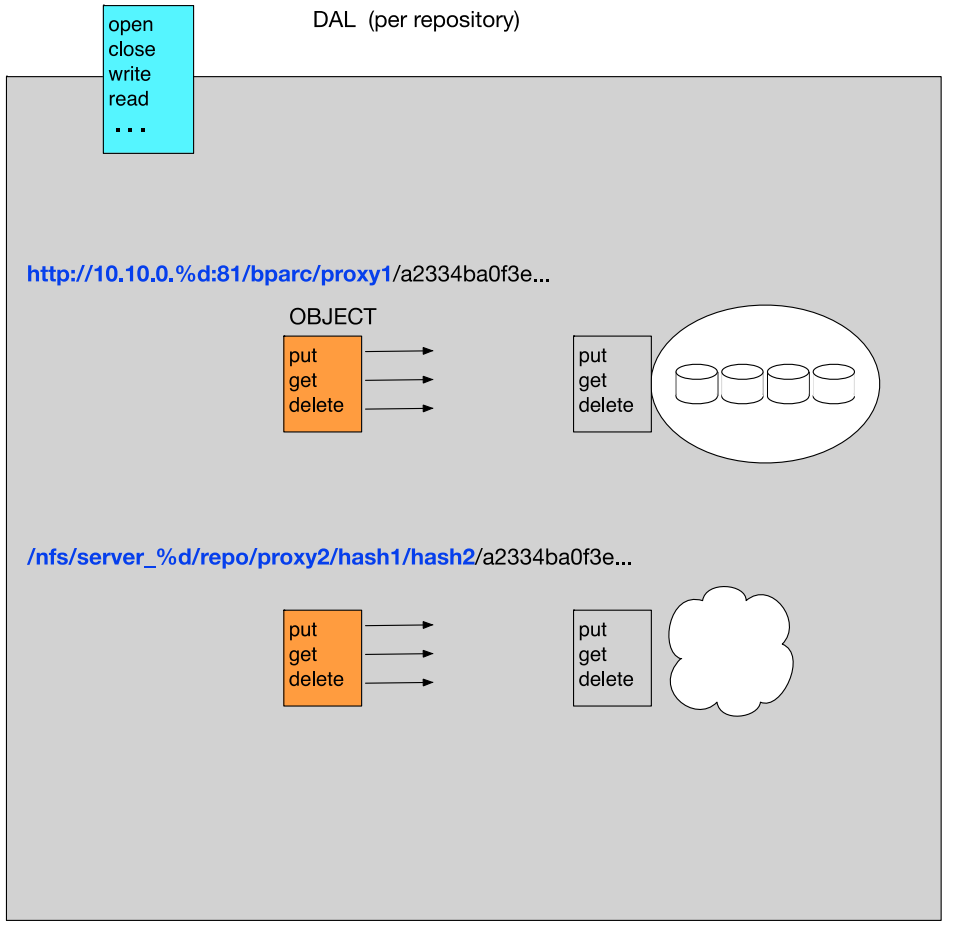
- **Uni (1 file : 1 object)**
- **Multi (1 file : N objects)**
- **Packed (M files : 1 object)**
  - offline packer
  - GC / repacker
  - pftool

- **recovery-info at tail of file storage, after file-data**
  - travels with packed files

Modular Implementations

MDAL / DAL





Configuration

- **Build Options**
  - libaws4c, libne, libmarfs, pftool
- **Configuration File**
  - XML parser-generator (PA2X)
  - **Repositories** define data-storage impl
    - Select DAL (Object, MC, RDMA)
    - Erasure-Coding parameters
  - **Namespaces** define MetaData impl
    - Select MDAL (POSIX, scaling experiment)

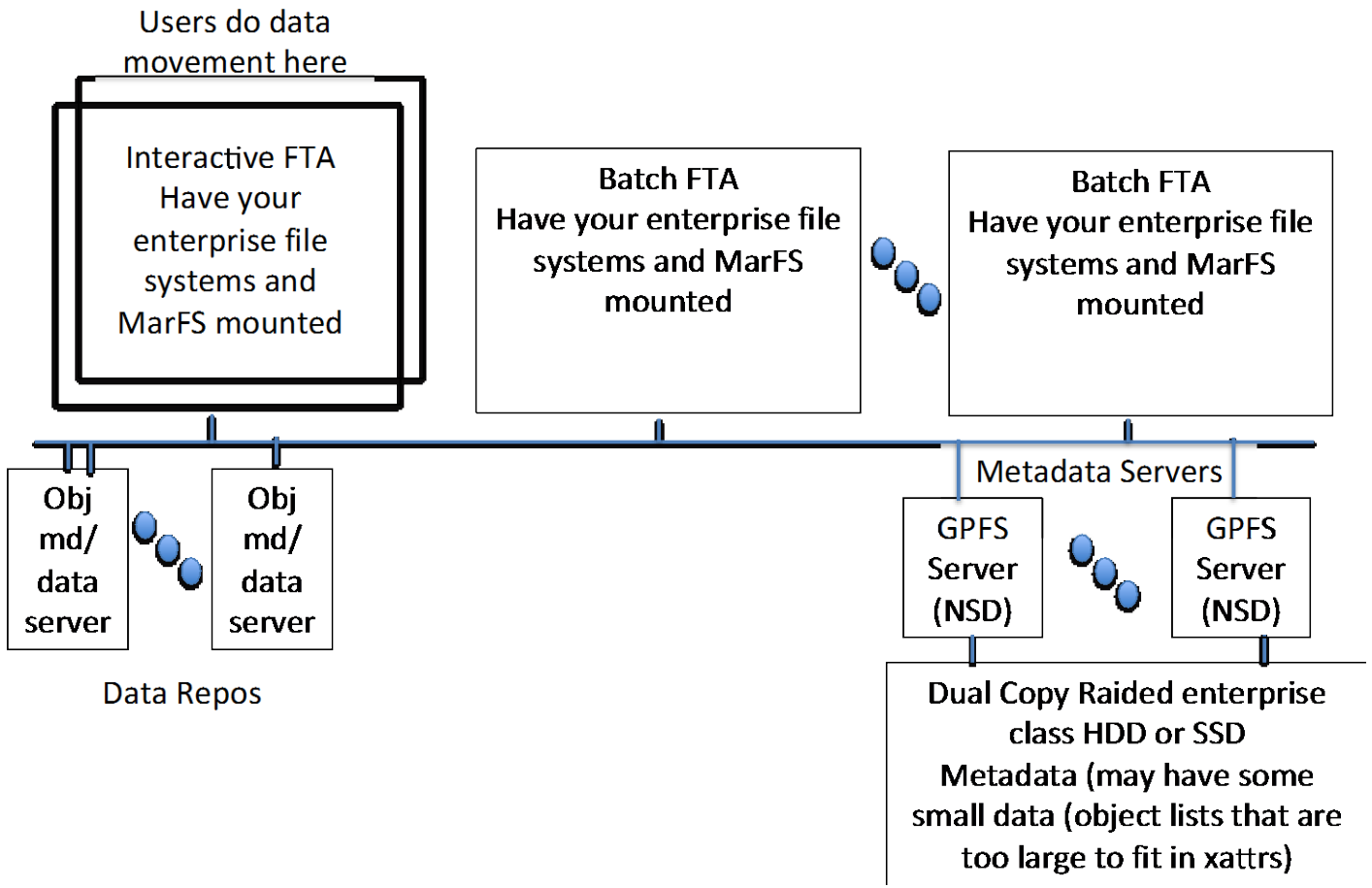
```
<config>
   <name>ODSU Testbed</name>
   <version>1.0</version>
   <mnt_top> /campaign </mnt_top>

   [repositories ...]

   [namespaces ...]

</config>
```

Configuration

```
<namespace>
   <name>admins</name>
   <alias>proxy1</alias>
   <mnt_path>/project1</mnt_path>

   <iperms> RM,WM,RD,WD,TD,UD</iperms>       # interactive (fuse)
   <iwrite_repo_name>bparc</iwrite_repo_name>  # matches <repo> spec

   <bperms>RM,WM,RD,WD,TD,UD</bperms>  # batch (pftool)
   <range>                # batch writes for files in given size-range
      <min_size>0</min_size>
      <max_size>-1</max_size>
      <repo_name>bparc</repo_name>
   </range>

   <md_path>          /gpfs/projects/project1/mdfs  </md_path>
   <trash_md_path> /gpfs/projects/trash             </trash_md_path>
   <fsinfo_path>       /gpfs/projects/project1/fsinfo       </fsinfo_path>

   <quota_space>-1</quota_space>
   <quota_names>-1</quota_names>
</namespace>
```

```
<repo>
   <name>bparc</name>

   # 10.10.0.1 - 10.10.0.12
   <host>10.10.0.%d:81</host>
   <host_offset>1</host_offset>
   <host_count>12</host_count>

   <update_in_place>no</update_in_place>
   <access_method>SPROXYD</access_method>
   <chunk_size>1073741824</chunk_size> # 1GB

   <security_method>HTTP_DIGEST</security_method>

   <enc_type>NONE</enc_type>
   <comp_type>NONE</comp_type>
   <correct_type>NONE</correct_type>
   <latency>10000</latency>
</repo>
```

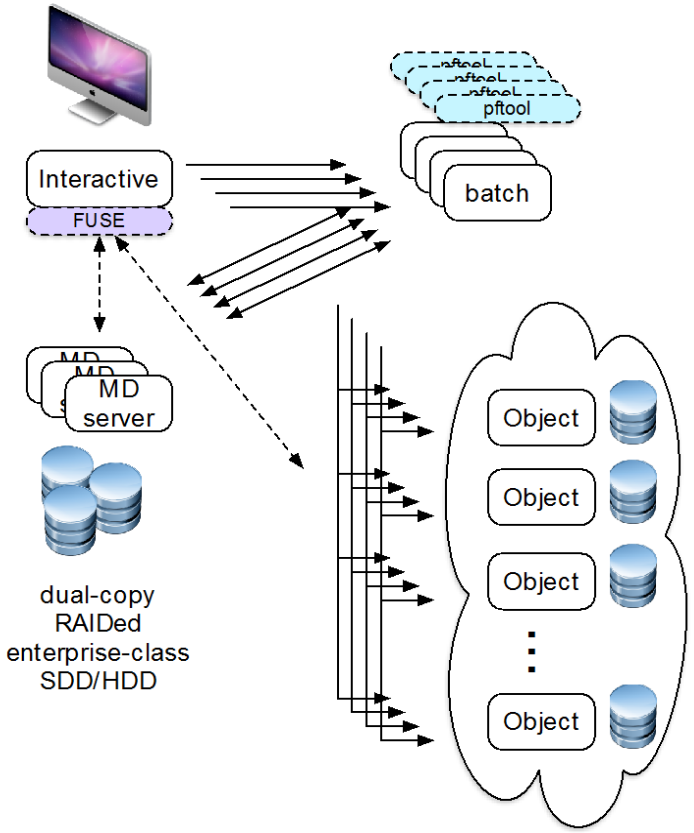
Simple MarFS Deployment



Thanks: Dave Bonnie, Kyle Lamb

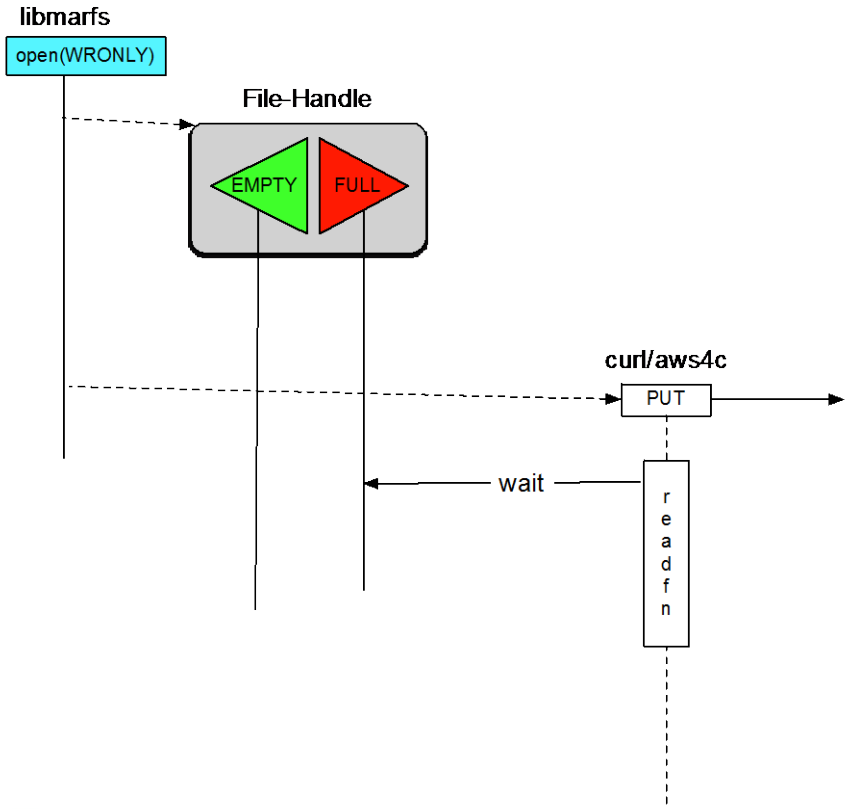
Object-Storage Repository (DAL)

- **pftool runs on batch FTAs**
  - pick object-server at random
  - object-servers mediate access to storage internally

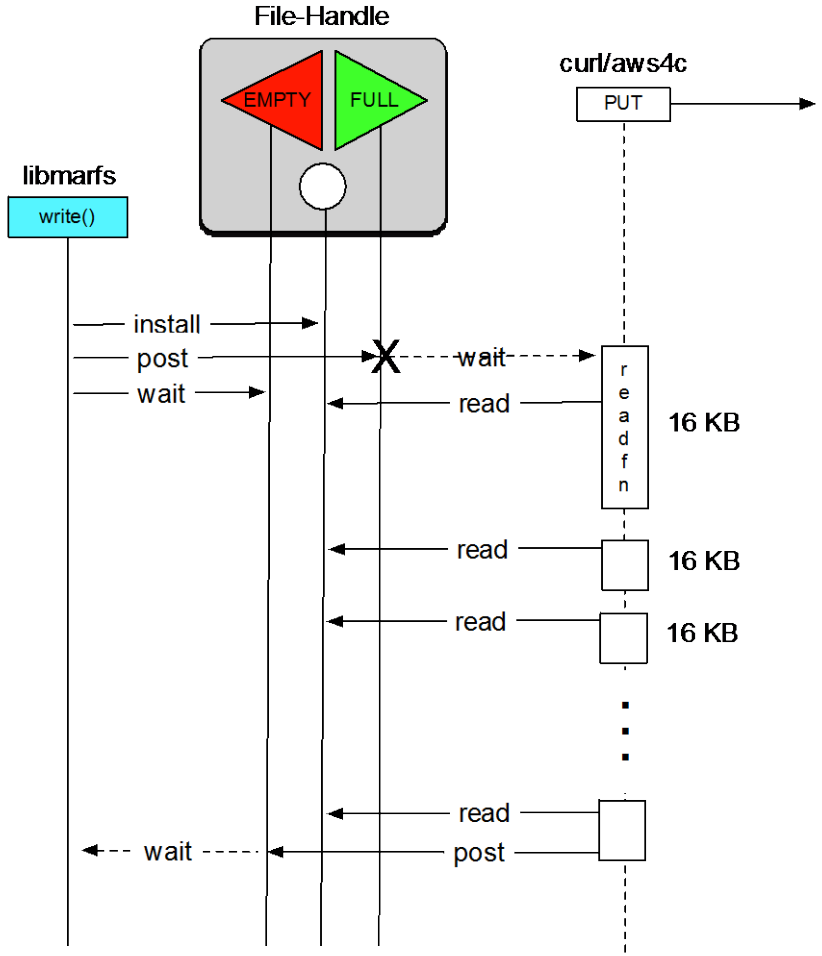


Writing to Object-Store (libcurl)

open(O_WRONLY)



write()



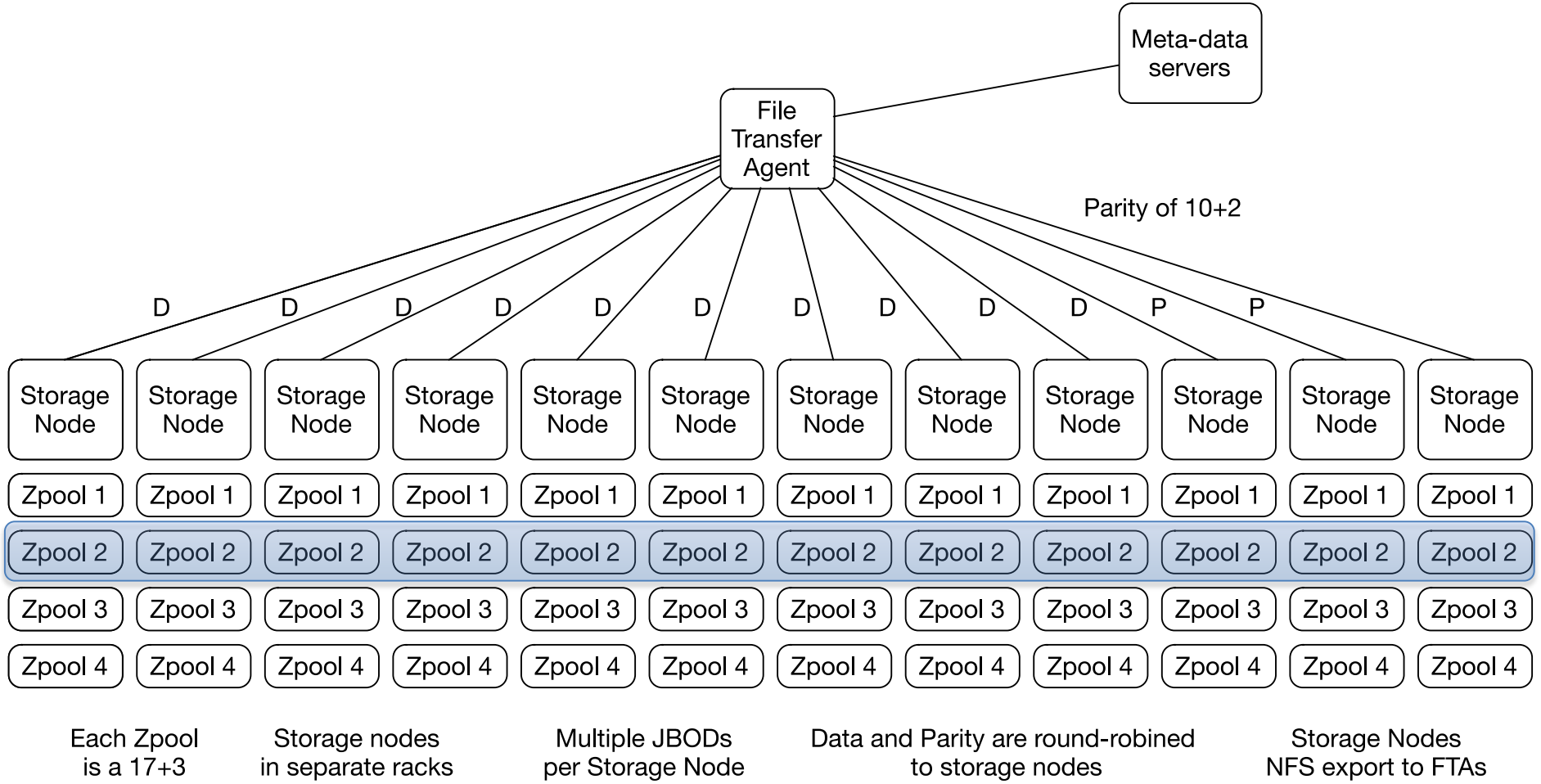
MetaData scaling demo (MDAL)

- **We built a test harness with MPI to push the limits of the MarFS design**
- **Initial MD scaling runs on Cielo (1.1 PF, ~140,000 cores, ~9000 nodes)**
  - 968 billion files, one single directory
  - 835 million file creations *per second* to metadata repos on each node
  - No cheating! Every create traversed the network from client to server
    - QD=1 for each client, 8 clients per node, 8 servers per node
- **Further testing on Mustang (230 TF, ~38,000 cores, ~1600 nodes)**
  - Large directory readdir tested from 10-50 nodes (sequential and parallel)
  - Peak of 300 million files per second across 50 nodes
  - More than 400X speedup over a single client sequential readdir

Thanks: Dave Bonnie, Kyle Lamb

Multi-Component Repository (DAL)

Parity over multiple ZFS pools



Los Alamos NATIONAL LABORATORY EST. 1943

LA-UR 17-22390

Thanks: Dave Bonnie, Kyle Lamb

Multi-Component Configuration

- **Replace object-store with files**
  - Erasure-Coding with Intel ISA-L
  - “stripes” written across independent FS
  - NFS exports = num_pods * (n + e)
  - “scatter” dirs reduce objs / dir

```
<repo>
  <name>mc10+2</name>

  <host>/nfs/mc/repo10+2/pod%d/block%s/cap%d/scatter%d</host>
  [ … ]

  <dal>
    <type>MC</type>

    <opt><key_val> n : 10                    </key_val></opt>
    <opt><key_val> e : 2                     </key_val></opt>
    <opt><key_val> num_cap : 1               </key_val></opt>
    <opt><key_val> num_pods : 1              </key_val></opt>
    <opt><key_val> scatter_width : 128 </key_val></opt>
    <opt><key_val> degraded_log_dir : /gpfs/marfs/mc/degraded  </key_val></opt>
  </dal>

</repo>
```

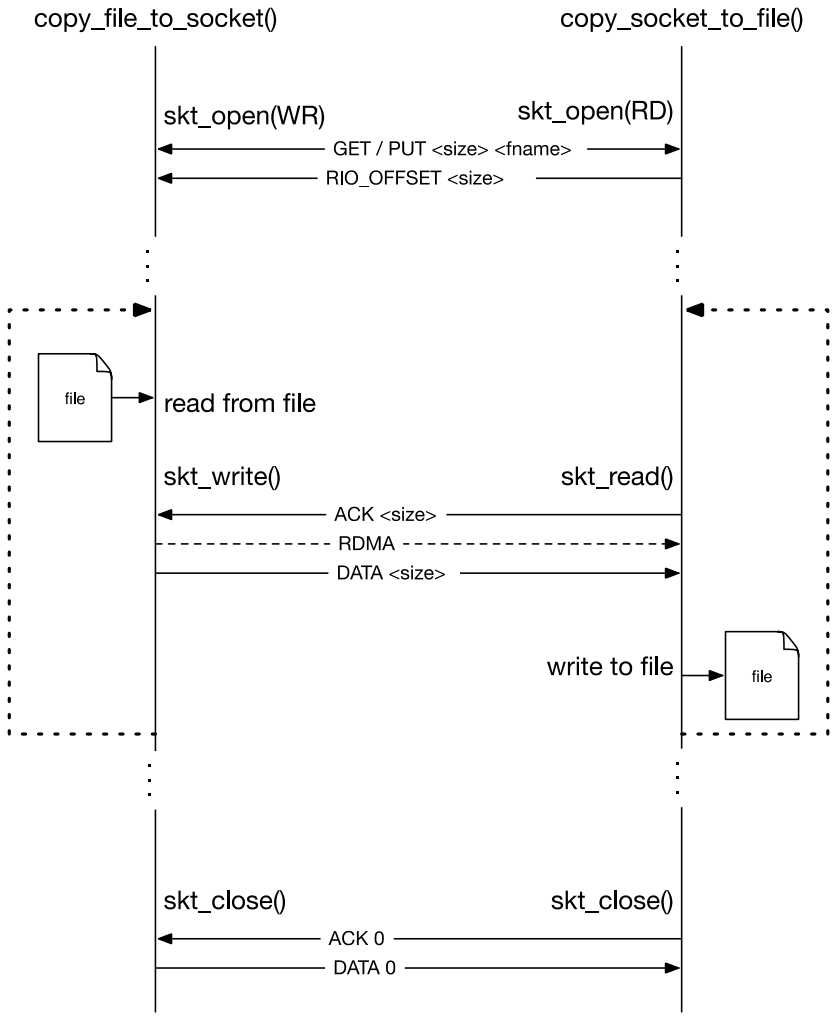
RDMA Multi-Component (DAL)

- **Replace NFS w/ remote service**
  - Erasure-Coding with Intel ISA-L
  - “stripes” written across independent FS
  - file access is a client/server interaction
  - DAL impl needs more info:
    - correlate block with server

```
<repo>
  <host>  10.10.0.%%d:xxxxx/zfs/exports/repo10+2/pod%d/block%%d/sockets%d  </host>
  [ … ]

  <dal>
    <type>MC_SOCKETS</type>

    <opt> <key_val> host_offset : 1                  </key_val> </opt>
    <opt> <key_val> host_count : 6                   </key_val> </opt>

    <opt> <key_val> blocks_per_host : 2              </key_val> </opt>
    <opt> <key_val> block_offset : 1                 </key_val> </opt>
    <opt> <key_val> global_block_numbering : 1       </key_val> </opt>
    <opt> <key_val> pod_offset : 1                   </key_val> </opt>

    [ … ]    # inherit the rest from MC DAL

  </dal>
</repo>
```

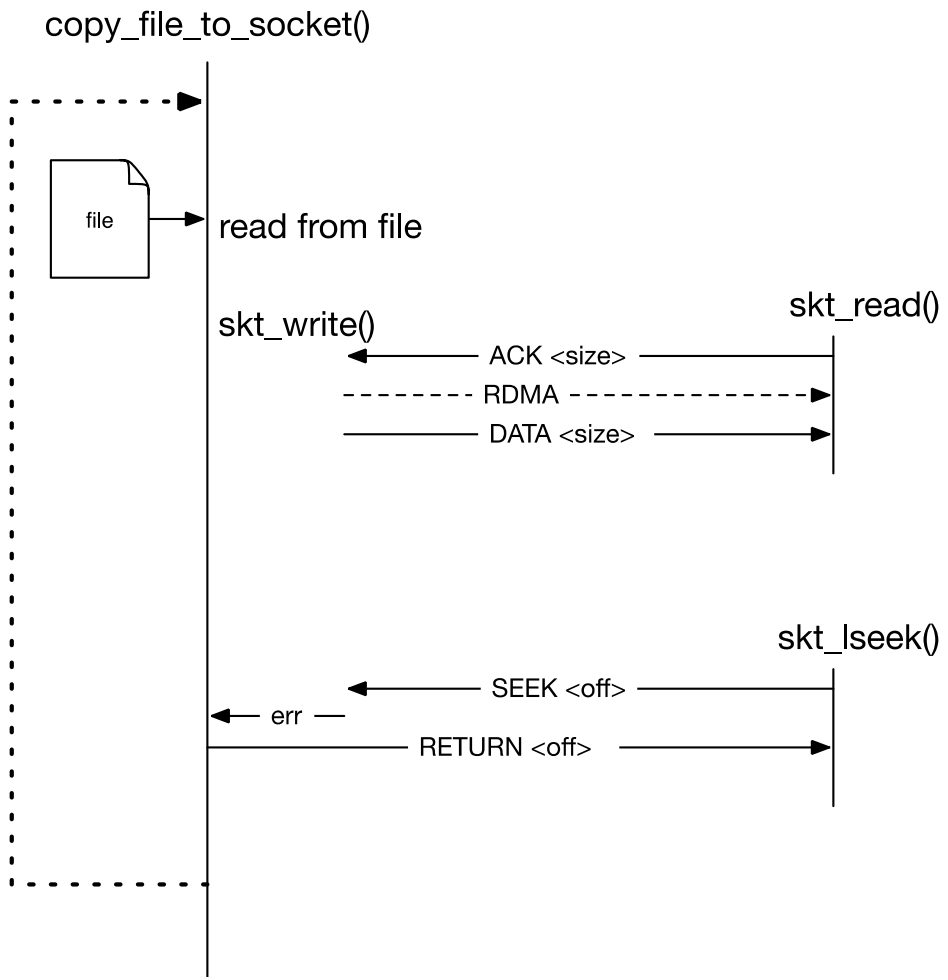
GET / PUT with RDMA

- **Using rsockets**
  - reader performs *riomap*
  - send offset to writer
- **RDMA ops transparent to reader**
  - reader informs writer when buffer is free
  - writer informs reader after RDMA
- **use get/put, or open/write/close**



Minor Complications

- **lseek() is allowed on a GET stream**
  - kick back to copy_file_to_socket()
- **fsync() allowed on a PUT stream**
  - kick back to copy_socket_to_file()



Preliminary Performance Numbers

- **Moving data all the way to disk**
  - (likely disk-limited)
  - 8TB SMR
- **Object-Store**
  - prod: 48 DSUs, 24-disk JBOD
  - test: 6 DSUs, 24-disk JBOD
- **Multi-Component (NFS / RDMA)**
  - test: 6 DSUs, 2 x 12-disk zpools
  - prod: 48 DSUs, 2 x 12-disk zpools

Test-Bed

Repository	Write (GB/s)	Read (GB/s)
Object-Store		
MC + NFS	2.3 – 1.3	
MC + RDMA	2.4 -- ?	

Production

Repository	Write (GB/s)	Read (GB/s)
Object-Store	20	20
MC + NFS	12	22
MC + RDMA		

Los Alamos NATIONAL LABORATORY EST. 1943

Team

- Dave Bonnie
- Hsing-Bung Chen
- Ron Croonenberg
- Chris DeJager
- Greg Geller
- Gary Grider
- Chris Hoffman
- Jeff Inman
- Brett Kettering
- Kyle Lamb
- Chris Mitchel
- Garrett Ransom
- George Sparrow
- Alfred Torrez
- Will Vining
- Scott White

13th ANNUAL WORKSHOP 2017

THANK YOU

Jeff Inman

Los Alamos National Laboratory