SAND2013-5274P

Exceptional service in the national interest

Sandia National Laboratories

Photos placed in horizontal position with even amount of white space between photos and header

Sirocco Status

Matthew Curry

U.S. DEPARTMENT OF ENERGY | NNSA

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000. SAND NO. 2011-XXXXP

Introduction

Sandia National Laboratories

- L2 Milestone
  - POSIX compliance
  - Write performance
  - Read capability
- Components
  - Lock server
  - Data location service
  - Scatter/gather API

Lock Server

Sandia National Laboratories

- POSIX requires atomicity
  - Namespace manipulation
  - File write access
- Related: TAMU work leveraging ASG semantics
- Short term: Lock service
  - Clients can obtain global leased lock
    - N-Party Locks
    - Attached metadata
    - Lock refreshing
    - Lock revocation
    - Knowledge of interspersed locking by others

Data Location

Sandia National Laboratories

- Sirocco servers may move data at will
  - Resilience, Capacity, Spite
- Related: Haiying's work, Zhiwei's work
- Short term: Unsophisticated data location
  - Data location depends on netgraph functionality
    - Netgraph does bootstrapping/overlay networking
  - A server broadcasts received requests, short-circuited by:
    - A request cache (keyed by <originating address | operation ID>)
    - TTL
  - Location information *only* is returned to client
    - <Record offset, number of records, location address, version> x *n*
  - Finite receive buffer yields need for extra information, i.e. "last known-good offset"
    - Catastrophic cancellation of location information

Scatter-Gather API

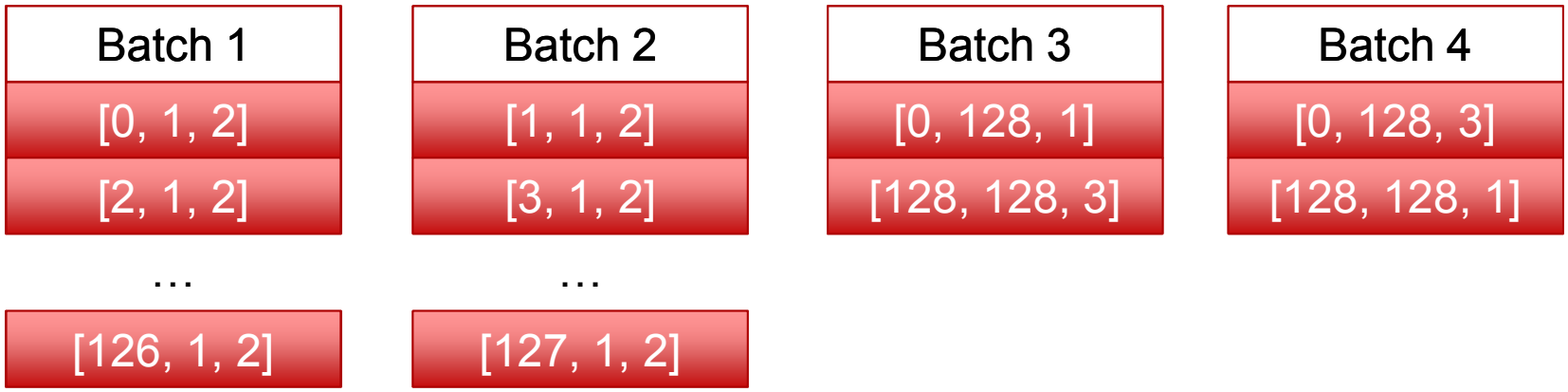
Sandia National Laboratories

- Defined wrapper API over ASG API to perform scatter/gather reads
  - Similar to PVFS noncontiguous access, or xread/xwrite in libsysio
- Internally, uses the batching/transactional functionality that we defined for Sirocco
  - ```
    ss_tx *tx = start_tx(…);
    asg_op(…, tx);
    tx_submit(tx);
    tx_wait(tx);
    ```
- The batching made the implementation of this API quick and easy, and API makes certain operations within client quick and easy.
- Is it time to consider adding batching to ASG interface?
    ```

Thanks

- Questions?

Location: Catastrophic Cancellation



- [x, y, z] is an extent with offset=x, nrecords=y, and updateID=z
- Merging from left to right, with only 128 slots available
- After merging batches 1 and 2, all slots are filled
- After merging batches <1…2> and 3, all slots are still filled
- After merging batches <1…3> and 4, two slots are filled
  - Correct data from batch 3 ([128, 128, 3]) wasn't "found" because of buffer space constraints
  - Incorrect data from batch for ([128, 128, 1]) *was* included because capacity was freed during merge
- Solution: Keep offset of known-correct data, keep minimum of all reported offsets through reduction