REDDnet: Enabling Data Intensive Science in the Wide Area
Overview

- What is REDDnet? and Why is it needed?
- Logistical Networking
- L-Store – Putting it all together
- Lessons Learned

This is a work in progress!
- NSF funded project
- Currently have 1PB
- Multiple disciplines
  - Sat imagery (AmericaView)
  - HEP
  - Bioinformatics
  - Vanderbilt TV News Archive
  - Texas Tech University Libraries
  - CMS-HI

**Goal:** Providing *working* storage for *distributed* scientific collaborations.
Focus on increasing bandwidth and raw storage.

Assume metadata growth is minimal.
“Data Explosion” of both data and metadata

- Focus on increasing bandwidth and raw storage
- Assume metadata growth is minimal
- Works great for large files
- For collections of small files the metadata becomes the bottleneck
- Need ability to scale metadata

ACCRE examples
- Proteomics: 89,000,000 files totaling 300G
- Genetics: 12,000,000 files totaling 50G in a single directory
Design Requirements

- **Availability**
  - Should survive a partial network outage
  - No downtimes, hot upgradable

- **Data and Metadata Integrity**
  - End-to-end guarantees a must!

- **Performance**
  - **Metadata**(transactions/s)
  - **Data**(MB/s)

- **Security**

- **Fault Tolerance** - Should survive *multiple* complete device failures
  - Metadata
  - Data

Focus to date has been on *data* performance, fault tolerance, and reliability!

Very different design requirements
Logistical Networking
Logistical Networking (LN) provides a “bits are bits” Infrastructure

- Standardize on what we have an adequate common model for
  - Storage/buffer management
  - Coarse-grained data transfer

- Leave everything else to higher layers
  - End-to-end services: checksums, encryption, error encoding, etc.

- Enable autonomy in wide area service creation
  - Security, resource allocation, QoS guarantees

- Gain the benefits of interoperability!
IBP Internet Backplane Protocol

- Middleware for managing and using remote storage
- Allows advanced space and **TIME** reservation
- Supports multiple connections/depot
- User configurable block size
- Designed to support large scale, distributed systems
- Provides global **`malloc()`** and **`free()`**
- End-to-end guarantees
- Capabilities
  - Each allocation has separate Read/Write/Manage keys

*http://loci.cs.utk.edu*
What makes LN different?

Based on a highly generic abstract block for storage (IBP)

Logistical Networking

Traditional Approach
What makes LN different?

Based on a highly generic abstract block for storage (IBP)
IBP Server or Depot

- Runs the ibp_server process
- Resource
  - Unique ID
  - Separate data and metadata partitions (or directories)
  - Optionally can import metadata to SSD
- Typically JBOD disk configuration
- Heterogeneous disk sizes and types
- Don’t have to use dedicated disks

- Current depots have 36 2TB to 4TB drives and can sustain 20Gb/s (disk check summing to protect against bit rot) to 28Gb/s (no disk chksum)
- 110Gb/s is the currently highest sustained transfer rate and was network limited.

| RID: CMS-1004 | Size: 120GB | Type: SSD |
| RID: 1005     | Size: 2TB   | Type: Disk |
| RID: 1006     | Size: 1TB   | Type: Disk |

Supports byte level R/W verification via checksumming!
IBP functionality

- Allocate
  - Reserve space for a limited time
  - Can create allocation Aliases to control access
  - Enable block level disk checksums to detect silent read errors

- Manage allocations
  - INC/DEC allocation reference count (when 0 allocation is removed)
  - Extend duration
  - Change size
  - Query duration, size, and reference counts

- Read/Write
  - Optionally use network checksums for end-to-end guarantees
  - Depot-Depot Transfers
  - Data can be either pushed or pulled to allow firewall traversal
  - Supports both append and random offsets
  - Pheobus support – I2 overlay routing to improve performance

- Depot Status
  - Number of resources and their resource ID’s
  - Amount of free space
  - Software version
What is a “capability”?

- Controls access to an IBP allocation
- 3 separate caps/allocation
  - Read
  - Write
  - Manage - delete or modify an allocation
- Alias or Proxy allocations supported
  - End user never has to see true capabilities
  - Can be revoked at any time
exNode
INI file containing metadata

- Analogous to a disk I-node and contains
  - Allocations
  - How to assemble file
  - Fault tolerance encoding scheme
  - Encryption keys

IBP Depots

Network

A
Normal file

B
Replicated at different sites

C
Replicated and striped
L-Store Architecture
L-Store Architecture

- **Exnode – Holds the “file” data**
  - Collection of data blocks and segments to provide different views of data
  - Different segments can be used for versioning, replication, optimized access (row vs. column), etc.

- **Segment**
  - Collection of blocks with a predefined structure.
  - Type: Linear, LUN, RAID5, Generalized Reed-Solomon, Log, caching, etc.
  - Can be stacked with other segments

- **Resource Service**
  - Data placement (stripe across depots vs. across disks) and lookup
  - Boolean query expression

- **Data Service**
  - Performs the actual data operations
  - Currently only IBP is supported

- **Object Service – File and directory services**
  - Metadata operations
  - Arbitrary attributes, symbolic, and hard linking is supported
  - Full support for streaming operations to minimize latency effects. Most operations can be implemented with a single call (ls -l, mkdir, find, etc)
Basic Segment types

**Linear**

- Single logical device
- Blocks can have unequal size
- Access is linear

**LUN – Striped data**

- Multiple devices

**LUN - Shifted Stripe**

- Multiple devices using shifted stride between rows

**RAID5 and Reed-Solomon segments**

are variations of the Shifted Stripe
Object Service

- **Performs all metadata operations**
  - Currently only implementation is based on a file system backend.

- **Functionality**
  - Fsck interface
  - Arbitrary attributes
    - Virtual attributes supported
    - Regex supported for attribute selection as well as a fixed list
    - Symbolic linking of attributes between objects is allowed
    - Can copy or move attributes between objects.
  - Credentials for AuthN/AuthZ
  - Create/Remove objects with full Regex support
  - Hard and Symbolic links are supported
  - Rename an object
  - Recursive Object iterator with full regex support.
  - Distributed File locking
L-Store – Putting it all together
The **Logistical Input/Output (LIO)** command line tools are designed to replicate the normal Linux File System Tools.

<table>
<thead>
<tr>
<th>Linux</th>
<th>LIO Specific</th>
</tr>
</thead>
<tbody>
<tr>
<td>lio_cp</td>
<td>lio_fuse</td>
</tr>
<tr>
<td>lio_du</td>
<td>Load the FUSE mount</td>
</tr>
<tr>
<td>lio_find</td>
<td>lio_get</td>
</tr>
<tr>
<td>lio_fsck</td>
<td>Equivalent to cat</td>
</tr>
<tr>
<td>lio_ln</td>
<td>lio_getattr</td>
</tr>
<tr>
<td>lio_ls</td>
<td>Get a file attribute</td>
</tr>
<tr>
<td>lio_mkdir</td>
<td>lio_inspect</td>
</tr>
<tr>
<td>lio_move</td>
<td>Checks integrity/repairs a file</td>
</tr>
<tr>
<td>lio_mkdir</td>
<td>lio_put</td>
</tr>
<tr>
<td>lio_ls</td>
<td>Pipe from stdin to a file</td>
</tr>
<tr>
<td>lio_mkdir</td>
<td>lio_setattr</td>
</tr>
<tr>
<td>lio_rm</td>
<td>Set a file attribute</td>
</tr>
<tr>
<td>lio_rmdir</td>
<td>lio_signature</td>
</tr>
<tr>
<td>lio_touch</td>
<td>Prints exnode container structure</td>
</tr>
<tr>
<td>lio_ls</td>
<td>lio_warm</td>
</tr>
<tr>
<td>lio_rm</td>
<td>Extends the file expiration date</td>
</tr>
</tbody>
</table>
Linux FUSE client

- Hard and Symbolic link supported.
- Full support for `xattr` interface.

- Tape backup and restore of both native L-Store and extended attributes.
- Can be used as a Tape backup disk cache.
- Integration with Teradactyl backup software.
L-Store Usage at VU

- CMS-HI has 3.5PB of native disk space
  - Routinely read over 1PB/week using the FUSE mount for production jobs and sustain 80Gb/s read rates (network limited).
- 800TB of shared space at VU
  - Vanderbilt TV News Archive has 150+ TB of space
- 240TB of space stored around the US
  - Working with TTU Libraries

24 hour plot of CMS-HI usage
Lessons Learned
Hardware Lessons

- Mechanical Vibrations
  - Disappearing drives
  - Broken electrical traces
  - Pre-mature drive failure

- Buggy Firmware
  - Drive hard locks requiring a power cycle

- Drive fragmentation

- Drive failures (6-8% AFR)
  - Replaced 50+ drives in 3 months

- Data integrity
  - Bit rot – Errors not detected by the drive.
  - Unrecoverable bit errors
    » typically $10^{14}$ or $10^{15}$ bits
    » $1\text{PB} = 8.8\times10^{12}$ bits
  - Journal replay
    » Metadata recovery is primary goal with data being secondary
    » Most common form of data corruption
Probability of data loss vs. Space efficiency with 6% AFR and 24hr rebuild time

Reed-Solomon $n+n/2$
Constant space efficiency
Probability of data loss vs. Rebuild Time
6% Annualized Failure Rate

Try to keep the rebuild time to a few hours!
De-clustered or Distributed RAID

Traditional RAID repair - Limited to speed of single disk

Distributed RAID has logical RAID arrays using many disks

The Logical arrays residing on the failed physical disk are repaired to remaining disks. Only data used is repaired. Many more disks take part in the repair.
Questions

http://www.lstore.org