PLFS/HDFS: HPC Applications on Cloud Storage

Chuck Cranor, Milo Polte, Garth Gibson, Carnegie Mellon University

Problem

High Performance Computing applications N-1 checkpoint to a single shared file in a parallel filesystem:

- Filesystem must allow multiple concurrent writers to files

Emergence of Hadoop: new types of clusters available. Hadoop storage on Hadoop Distributed Filesystem, key attributes:

- Single sequential writer (not POSIX, no prite)
- Not mounted, accessed through Java API
- Local storage on compute notes
- Data replicated 3 times (local+remote1+remote2)

HDFS storage resources cannot be used by HPC applications for N-1 checkpointing

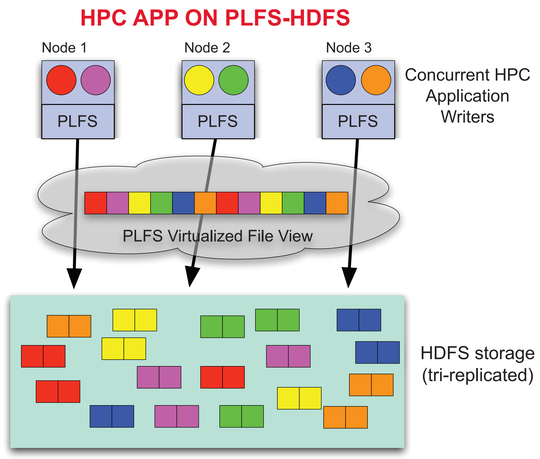
- Efforts to make HDFS a mountable filesystem (e.g. hdfs-fuse, fuse-dfs) do not relax HDFS' single writer restriction

Parallel Log Structured Filesystem (PLFS)

FUSE or MPI-based filesystem that converts N-1 checkpointing to N-N checkpointing by breaking each node's write operations out into a log file

- Improves HPC checkpoint performance by avoiding underlying filesystem bottlenecks
- PLFS's log structured writes fit the filesystem semantics provided by the HDFS cloud storage system

If PLFS could write its logs to HDFS, it could provide N-1 checkpoint semantics for HPC applications using HDFS for storage



HDFS API and Platform Issues

Direct mapping:

- access, chmod, rename, unlink, ...

File descriptor-based:

- open, close pread, write, fsync, ...

More Involved:

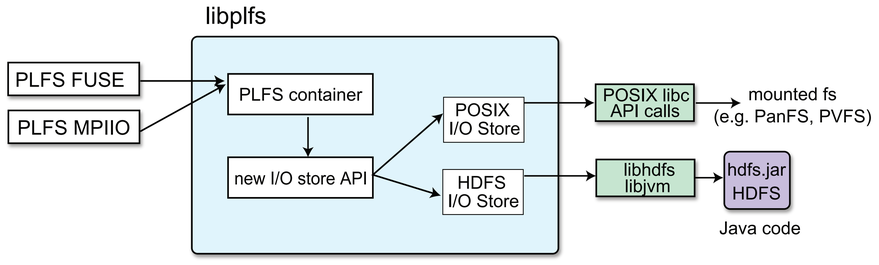
- owner/group strings vs uid/gid ints
- open/create/mkdir with mode
- diretory read
- mmap indx file (read-only)

Non-POSIX HDFS API semantics

- HDFS directory rename is really a move
- HDFS I/O on unlinked files genereates errors (unlike POSIX)

PLFS I/O Store Architecture

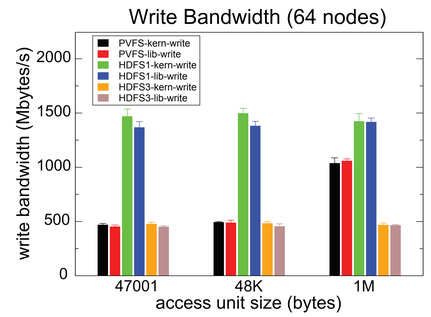
- Insert new I/O store layer into PLFS to allow multiple types of backing filesystems to be used (including non-POSIX ones)
- Current list of I/O stores: POSIX, HDFS, PVFS, IOFSL (in progress)
- HDFS I/O store module uses libhdfs API for log I/O
  - Hadoop's libhdfs uses Java Native Interface (JNI) to provide C/C++ access to HDFS Java methods

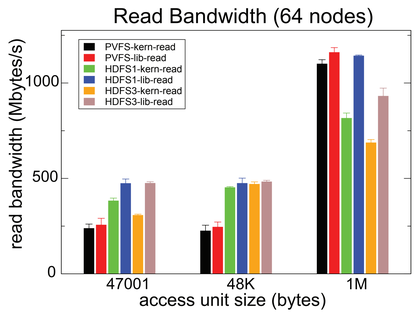


Results

Platform: Marmot PRObE cluster

- 1.6GHz AMD Opteron dual processor, 16GB memory, Gigabit Ethernet
- Hadoop HDFS 0.21.0, FUSE 2.8, PLFS, OrangeFS 2.8.4 (PVFS)
- LANL test_fs N-1 checkpoint benchmark with 47001, 48K, or 1M byte objects
- 6 test cases: PVFS, HDFS1 (no replication), HDFS3 (3 way replication) through a kernel mount point and a library API





PLFS/HDFS is roughly comparable to PVFS

- writes: HDFS1 always writes to local disk (fast, no network)
  - HDFS3 has 3x replication overhead
  - PVFS network limited with small access size
- reads: HDFS benefits from PLFS' log structured writes
  - Kernel buffer cache hurts 47001 reads due to page alignment