Storage Systems

NPTEL Course

Jan 2012

(Lecture 41)

K. Gopinath

Indian Institute of Science

Lease Mgmt

- designed to minimize mgmt overhead at master
- a lease initially times out at 60 secs.
  - primary can request and typ receive extensions indef
  - msgs piggybacked on heartbeats
- master may need to revoke a lease before expiry
  - eg. disable mutations on a file that is being renamed
- even if master loses comm with a primary, safe to grant a new lease to another replica after old lease expires.

Record Appends

- client pushes data to all replicas of last chunk of file
- client sends request to primary
- primary checks if append exceeds max size (64 MB)
  - if so, pad chunk to max size
  - tell secondaries to do same, and reply to client to retry on next chunk
- if record fits within max size (common case), primary appends data to its replica, tells secondaries to write data at its own offset, reports success to client

Record Appends (contd)

- if record append fails at any replica, client retries op
  - replicas of same chunk may contain different data possibly including duplicates of the same record in whole or in part.
  - GFS does not guarantee that all replicas are bytewise identical
  - only guarantees that data written at least once as an atomic unit
    - for op to report success, data must have been written at same offset on all replicas of some chunk
    - all replicas at least as long as the end of record and any future record will be assigned a higher offset or a different chunk even if a different replica later becomes primary
- regions in which successful record append ops have written their data defined (hence consistent), whereas intervening regions inconsistent (hence undefined)

Snapshots

- make a copy of a file or a dir tree “instantly”
  - minimize interruptions of ongoing mutations
- std COW: when master receives a snapshot request, revokes any outstanding leases on chunks in files to be snapshot
  - any later write to these chunks: first find lease holder from master
  - master can now create a new copy of chunk
- master logs op to disk; applies this log record to its in-memory state by duplicating metadata of source file or dir tree
  - newly created snapshot files point to same chunks as source files

Snapshots (contd)

- new client write to a chunk C (after snapshot)
  - sends request to master to find current lease holder
  - master notices ref count of C>1
    - defers replying to client req and instead picks a new chunk handle C'
    - asks each chunkserver with curr replica of C to create a new chunk called C’
      - data copied locally, not over netw (disks 3x 100 Mb Eth)
    - request handling now same: master grants one replica a lease on C’ and replies to client, client writes chunk normally

Master Operation

- executes all namespace ops
- manages chunk replicas:
  - makes placement decisions, creates new chunks and hence replicas
  - coordinates various system-wide activities to keep chunks fully replicated
  - balance load across all chunkservers
  - reclaim unused storage
- Many ops take long time: eg. snapshot op has to revoke chunkserver leases on all chunks covered by it
  - Allow multiple ops active and use locks over regions of namespace to ensure proper serialization

GFS dir ops

- GFS does not have a per-directory data structure that lists all the files in that directory
  - instead: lookup table mapping full pathnames to metadata
  - With prefix compression, efficient in memory
- No hard or symbolic links
- Each node in namespace tree (absolute file/dir name) has an associated read-write lock
- Each master op acquires a set of locks before it runs
  - op on /d1/d2/.../dn/leaf: acquire read-locks on dir names /d1, /d1/d2, ..., /d1/d2/.../dn, and
  - either a read lock or a write lock on full pn /d1/d2/.../dn/leaf
  - allows concurrent mutations in same dir.
    - multiple file creations concurrent in same dir: each acquires a read lock on the directory name and a write lock on the file name

GFS dir ops (contd)

- Eg: prevent a file /home/user/foo from being created while /home/user is being snapshotted to /save/user
- snapshot creat snapshot
- /home R R /save R
- user W R user W
- foo W
- Since /home/user is locked R or W, only one of them can proceed

- Related HSM Problem: file being migrated in by perpetrator (locks inode a)
- victim: looks up file
  - parent of dir locked while look up goes on
  - but file locked by perpetrator, so hangs during migration in
- next victim: looks up parent of dir!

GFS Locks

- read-write lock objects allocated lazily
  - to handle large namespace with many nodes
  - deleted once not in use
  - locks taken in a consistent total order to prevent deadlock
    - ordered first by level in the namespace tree
    - next, lexicographically within the same level

Replica Placement

- hundreds of chunkservers and clients spread across many machine racks crossing many netw switches
  - also, bw into or out of a rack may be less than aggregate bw of all machines within rack
- need to maximize
  - data reliability and availability
    - spread chunk replicas across racks
    - data survives even if an entire rack damaged or offline
  - network bandwidth utilization
    - read traffic for a chunk: aggregate bw of multiple racks
    - write traffic: data to multiple racks, a tradeoff

Chunk Creation, Re-replication, Rebalancing

- master creates a chunk: choose loc for initial empty replicas
  - place new replicas on chunkservers with below-average disk space utilization
  - limit number of “recent” creations on each chunkserver
    - creation cheap but heavy write traffic soon
    - append-once-read-many workload typ becomes read-only once they have been completely written
    - spread replicas of a chunk across racks
- master re-replicates a chunk: # avlbl replicas below reqd
  - some policies: boost priority of any chunk blocking client progress, or lost too many replicas, limit # of active repls in cluster and in each chunkserver; chunkserver limits repl bandwidth by throttling read reqs to source chunkserver
- master rebalances replicas

Garbage collection

- files deleted gc'ed later on regular fs scans
  - till gc, avlbl under a hidden name
  - eager deletion problematic (eg. loss of delete msgs)
    - write errors create garbage chunks unknown to master
- similar regular scan of chunk namespace:
  - master identifies orphaned chunks (not reachable from any file) and erases its metadata
- distributed garbage collection a hard problem but
  - file-to-chunk mappings only by master
  - any chunk replica is a Linux file in designated dir on each chunkserver
    - any such replica not known to master is garbage

Stale Replica Detection

- replica may become stale if chunkserver fails and misses mutations
- master maintains a chunk version number to distinguish between up-to-date and stale replicas.
- when master grants a new lease on a chunk, incr chunk version # and informs up-to-date replicas
  - master and these replicas all log new version #
    - before any client notified and write to chunk.
- if a replica is not avlbl, its chunk version number not incr
  - master detects stale replica when chunkserver restarts and reports its set of chunks and associated version#
- if master sees a version # greater than its #, master assumes that it failed when granting lease and so takes higher version to be up-to-date.

Fault Tolerance and Diagnosis

- High Availability
  - Fast recovery:
    - do not distinguish betw normal and abnormal termination
    - servers routinely shut down just by killing process
    - clients and other servers that time out on their requests: reconnect to restarted server, and retry
  - Chunk replication
  - Master Replication:
    - Master's op log and checkpoints replicated on multiple machines
    - A mutation to state committed only after its log record flushed to disk locally and on all master replicas.
      - But one master process remains in charge of all mutations as well as background activities such as garbage collection that change system internally
    - When master fails, restart almost instant.
      - if its machine or disk fails, monitoring infrastructure outside GFS starts a new master process elsewhere with replicated operation log
    - Shadows “read only” servers; try to track master by applying log ops

(contd)

- Data Integrity
  - 32bit checksums in memory and logged persistently, separate from user data
  - impractical to detect corruption by comparing replicas across chunkservers
    - divergent replicas legal (esp with atomic record append
    - each chunkserver must independently verify integrity of its own copy by maintaining checksums
  - during idle periods, chunkservers scan and verify contents of inactive chunks
- Diagnostic tools
  - detailed diagnostic (asynch) logging

Summary

- First really large FS
  - Reliable
  - Scalable
  - Available
- No POSIX...