Serverless Network File Systems

Anderson, Dahlin, Neefe, Patterson, Roselli, Wang, 1996

What kind of paper?

- Presents a new idea.
- Describes a system.
- Present performance of the system.

Motivation

- High-speed switched LANs.
    - Switching implies that network bandwidth increases with machines (no shared net)
    - High-speed implies ability to move more data per user.
- Increased user demands.
- Performance and reliability limitations of a central server.

Technology

- Dynamically distribute control per-file.
- Use RAID and LFS for redundancy and performance.
- Cache cooperatively, sharing distributed memory.

Caveats

- Machines must have fast interconnect.
- Machines must trust each other (with each other's data).

Zebra

- Combines RAID, LFS, and networking.
- Each client accumulates data to form a segment.
- That segment is broken up into stripe units that are written to individual storage servers.
- Parity is computed locally at client.
- Distributed state updated atomically using deltas.

Key things that xFS addresses that Zebra did not:

- Zebra used a single file manager to map blocks to machines.
- Zebra used a single cleaner.)
- Zebra stripes across all storage servers.

Open Issues

- Scalable, distributed meta-data management and reconfiguration.
- Scalable subsetting.
- Scalable log cleaning.

The Manager Map

- Maps a file id (index number) to a particular manager.
- Globally replicated (all managers and all clients).
- Big table indexed by bits of the file ID.
- Entry specifies a specific machine.
- Helps do load balancing.
- Dynamically reconfigurable.
- Have an order of magnitude more entries than managers.
- Old managers send state to a new manager to distribute load.
- Relatively small and slowly changing.
- Managers keep track of cache consistency information and disk block pointers.

The IMap

- Maps a file ID to the disk address of the file’s index node.
- Same as the index map in LFS (look up this name).
- Distributed to managers along with the manager map (i.e. a manager manages the portion of the imap containing the files it manages).
- The IMap points to index nodes which resemble FFS inodes.

File Directories

- Maps file names to file IDs.
- Directories are regular files.
- New files managed by the client that created the file.

Stripe Group Map

- Maps disk address to a list of storage servers (containing the stripe).
- Globally replicated (small and slow-changing; prototype does not do dynamic modification).
- Necessity if number of servers is high.
- Contains group id, group members, group status.
- Dynamic reconfiguration achieved using obsolete groups.
- Cleaner cleans up obsolete groups.

Read Path

1. Directory Lookup yields file index number.
2. Lookup file/block in local cache; if found, done.
3. Else, send request to the files' manager.
4. The manager checks cache state and if the block is cache resident, the manager requests the caching machine to satisfy the request.
5. Else, we must get the block from disk.
6. The manager examines the index map to get the block address.
7. Using this block address, the manager consults the stripe group map to figure out from whom to request the block.
8. The request goes to the storage server and the storage server sends the data to the client that initially requested it.

Cache Consistency

- Token-based, per-block consistency.
- Machines attain ownership before writes.
- Managers track caching and ownership.

Simulation 1: Management Distribution Policies

- Compare First Writer to Central Server.
- File Creator becomes manager.
- Use 7-day NFS trace for 236 client machines.
- One day’s warming; six day simulation.
- 16 MB local cache; all clients are managers.
- Files extant before trace begins are randomly assigned (reassigned if later written).
- Metric is number of network hops.
- Most writes remained local (90%).

Cleaning

- Client’s maintain segment information for each segment they create.
- Simulation 2: shows that the segment information can be kept up to date with little communication overhead.
- S-files contain segment usage information.
- S-files per stripe group.
- Stripe group leaders initiative cleaning based on low-water mark or idleness.
- Leader dispatches S-files to cleaners; cleaners clean those s-files.
- Optimistic locking on cleaned files.

Recovery

- Very little implemented (essentially RAID recovery is implemented).
- Storage server recovery based on Zebra.
- Parallelized by splitting work amount managers.

Security

- Assume that machines within an administrative domain trust each other.
- Can restrict circle of trust by having core trusted servers that export NFS.

Prototype

- Implements most of Zebra and distributed meta-data management.
- Real users do not put data there (less mature than Sprite-LFS).
- Loadable module in Solaris.
- 32 machines; 8 2-processor SPARC 20's and 24 SS10's.
- 64 MB physical memory.
- In NFS tests; use one server w/prestoserve.
- in xFS, all machines are clients, servers, and managers.
- 8 machine groups (7 date, 1 parity).
- 64 KB stripe units.
- 256 MB file partitions.
- Connection: myrinet (80 MB/sec, 3.2 MB/sec w/ 8kb packets).

Performance

- Write test: Write 10 MB file, sync; scales up to 35 clients.
- Read test: Read 10 MB file; scales linearly to 15, slightly to 35.
- Small file write test: 2048 1 KB files. (different semantics)
- Storage server scalability: add servers, get more throughput (probably works with NFS too).
- Manager scalability: scales up to about five managers.