File Systems

File Systems

NTFS

hadoop HDFS

Ext4 File System

NFS NETWORK FILE SYSTEM

PVFS

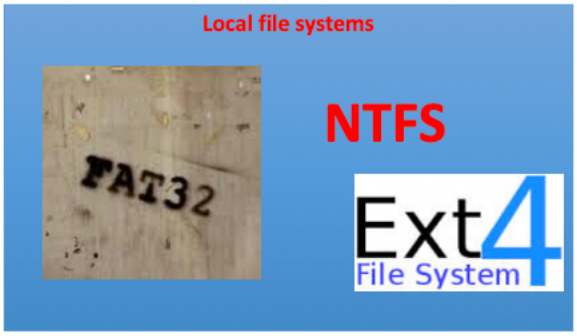
The Google File System

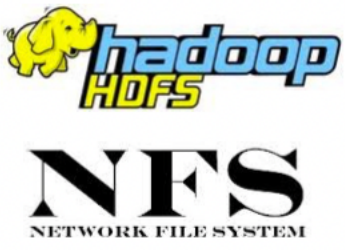
By Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung
(Presented at SOSP 2003)

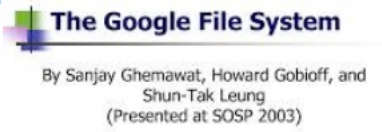
Overview

- Filenames:
  - File identity.
- Directories (folders):
  - Group of files in separate collections.
- Metadata:
  - Creation time, last access time, last modification time.
  - Security information (owner, group owner).
  - Mapping file to its physical location of file (e.g., location in storage devices).
- Computer file:
  - A resource for storing information.
  - Durable, remained available for access.
  - Data: sequences of bits.
- File system:
  - Control how computer files are stored and retrieved.
  - Main operators: READ, WRITE (offset, size), CREATE, DELETE.

Local vs distributed file systems







Distributed file system:

- File system:
  - Abstraction of storage devices.
- Distributed file system:
  - Available to remote processes in distributed systems.
- Benefits:
  - File sharing.
  - Uniform view of system from different clients.
  - Centralized administration.

Decentralized file system:

- IPFS.
- Web Torrent.

Goals: Network (Access) Transparency

- Network (Access) Transparency:
  - Users should be able to access files over a network as easily as if the files were stored locally.
  - Users should not have to know the physical location of a file to access it.
- Transparency can be addressed through naming and file mounting mechanisms.
  - Location Transparency: file name doesn't specify physical location.
  - Location Independence: Files can be moved to new physical location, no need to change references to them. (A name is independent of its addresses).
  - Location Independence → Location transparency, but the reverse is not necessarily true.

Goals: Availability

- Availability: Files should be easily and quickly accessible.
- The number of users, system failures, or other consequences of distribution shouldn't compromise the availability.
- Addressed mainly through replication.

Architectures

- Client-Server:
    - Sun Microsystem Network File System (NFS), Google File System (GFS).
    - Architecture:
        - One or more machines (file servers) manage the file system.
        - Files are stored on disks at the servers.
        - Requests for file operations are made from clients to the servers.
        - Client-server systems centralize storage and management; P2P systems decentralize it.
- Symmetric:
    - Fully decentralized; based on peer-to-peer technology.
    - E.g., Ivy (uses a Chord DHT approach).
    - E.g., IPFS and Web Torrent.

Design issues in distributed file systems

Design issues

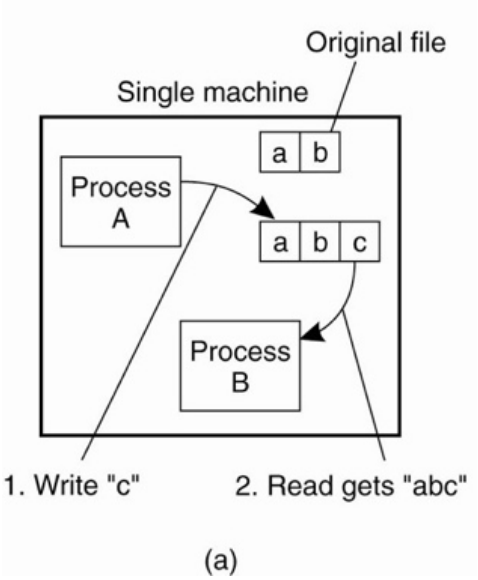
- Naming and name resolution.
- Semantics of file sharing.
- Caching.
- Replication.

Naming and name resolution

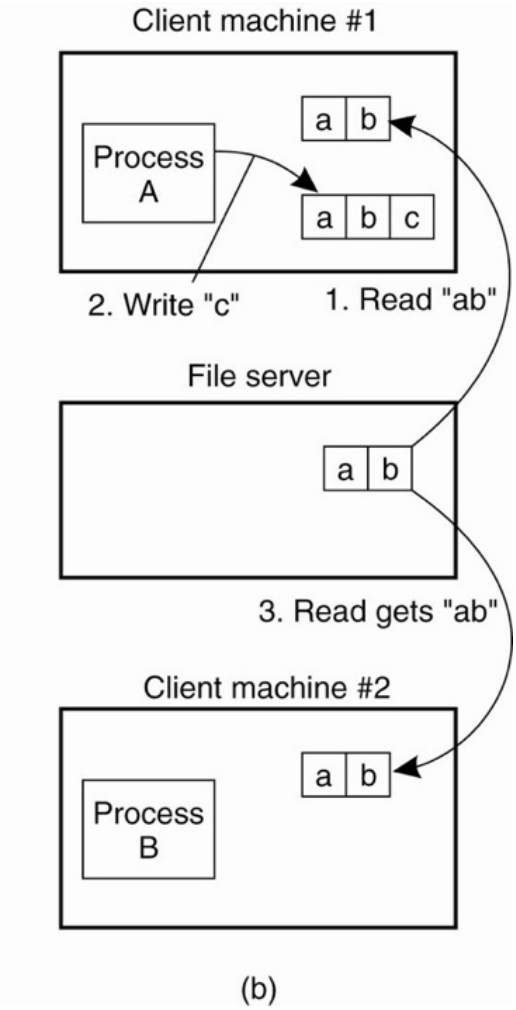
- A name space – collection of names.
- Name resolution – mapping of a name to an object.
- 3 traditional ways:
    - Concatenate the host name to the names of files stored on that host.
    - Mount remote directories onto local directories.
    - Provide a single global directory.

File Sharing Semantics

- Problem: When dealing with distributed file systems, we need to consider the ordering of concurrent read/write operations and expected semantics (=consistency).



(a)



(b)

- Assume open; reads/writes, close:
  - UNIX semantics:

        - Value read is the value stored by last write.
        - Write to an open file are visible immediately to others that have this file opened at the same time. Easy to implement if one server and no cache.
    - Session and semantics:
        - Write to an open file by a user is not visible immediately by other users that have files opened already.
        - Once a file is closed, the changed made by it are visible by sessions started later.
    - Immutable-Shared-Files semantics:
        - A sharable file cannot be modified.
        - File names cannot be reused, and its contents may not be altered.
        - Simple to implement.
    - Transactions:
        - All changes have all-or-nothing property.
        - W1, R1, R2, W2 not allowed where P1=W1; W2 and P2=R1; R2.

Caching

- Server caching: in main memory.
    - Cache management issue, how much to cache, replacement strategy.
    - Still low due to network delay.
    - Used in high-performance web-search engine servers.
- Client caching in main memory.
    - Can be used by diskless workstation.
    - Faster to access from main memory than disk.
    - Complete with the virtual memory system for physical memory space.
- Client-cache on local disk.
    - Large files can be cached.
    - The virtual memory management is simpler.
    - A workstation can function even when it is disconnected from the network.

Caching tradeoffs

- Reduces remote addresses → reduces network traffic and server load.
- Total network overhead is lower for big chunks of data (caching) than a series of responses to specific requests.
- Disk access can be optimized better for large requests than random disk blocks.
- Cache-consistency problem is the major drawback. If there are frequent writes, overhead due to the consistency problem is significant.

Replication

- File data is replicated to multiple storage servers.
- Goals:
    - Increase reliability.
    - Improve availability.

    - Balance the server's workload.
- How to make replication transparent?
- How to keep the replicas consistent?
    - A replica is not updated due to its server failure.
    - Network partitioned.

Table of Contents