Distributed File Systems: Concepts and Examples

- Overview paper on challenges of distributed storage systems

- Interesting point: Departure from extending centralized file systems is necessary to fully leverage the benefits of distribution. The question to ask (perhaps at the end of the course is) to what extent is this true in a) research prototypes and b) commercial products
  - There was lots of research on distributed storage in ‘80s (Locus, AFS, NFS etc. etc.)

Important performance parameters

- Fault tolerance: Distributed storage is composed of a large number of distributed storage components (rather than a single storage component). Increasing the number of components affects fault tolerance. Distributing the components makes the system more fault prone (because of network disruptions).
- Scalability: There is opportunity for better scalability by distributing components.

Naming and Transparency

- This is the problem of locating where the objects are stored and exposing that to the end applications.
  - Transparent access is preferable (though harder to achieve)
  - Location transparency: name of file does not reveal its physical location
  - Location independence: name of file does not change when physical storage location changes
  - Location independence is a stronger property as compared to transparency

"We envision future DFS that supports location independence completely and exploits the flexibility that this property entails".

Naming schemes

- Explicit location <host:/location>
- Mounted first and then location transparent: NFS style
  - The naming need not be consistent across all machines
  - E.g. machine a may mount a file system as /home while machine b may mount it as /u
- Total integration of names so that it is uniform across the entire system: AFS style
  - Problem with local files (device files, binaries etc.)

Implementation

- Pathname translation
  - Translate each component separately (for example in /a/b/c/d/file)
  - Structured identifiers which uniquely identy and object in the system (for independence)
  - Hints that may point to where the object is (without any guarantees that it will be available on that particlar location)

Semantics of Sharing

- UNIX (single file system) like:
  - Every read sees the effects of all previous writes
  - Writes to an open file are visible immediately to others
  - Clients can share a file pointer to index into the same file
- Session semantics:
  - Writes to open files are visible immediately to local clients, but invisible to remote clients
  - Once file is closed, changes are visible in remove sites (but not to already open files)
- Immutable files:
  - All files are write-once. Updates creates new versions
- Transaction-like:
  - Operations are serializable

Remote access methods

- Remote service: All requests have to go to the remote server for service.
  - Easy to implement. Does not offer many benefits of distribution
- Caching: Maintain local copies so that you don't have to ask the remote site all the time.
  - Better performance. Maintaing cache consistency is challenging.
  - Issues: Cache unit size (blocks or files), Location (main memory, disk), Modification policy (delayed write, write through), Cache validation (Client-initiated [NFS], Server-initiated [call backs of AFS])

Fault tolerance issues

- Stateful vs stateless (how much information does the server maintain about the replicas or who maintains this information, the server or the client?)
- File is recoverable if it is possible to rever to an earlier, consistent state. File is robust if it is guaranteed to survive crashes of the storage device.
  - These issues are orthogonal
- Replication can improve availability at the expense of complex update protocols.

Scalability

- "AFS is the closest system to be classified as very large-scale system"
  - Negative examples show the attributes of non-scalable architecture. E.g. centralizing components is not scalable. Scalability is a complex topic.

NFS and AFS

- NFS: Traditional NFS is block oriented, write-through caching, client-initiated cache validation with path name translations at each stage.
- AFS: File oriented, last-writer wins semantics. Stateful server with callbacks to invalidate caches on clients. Cells are used to provide location independent storage.

UNIX United

- A number of UNIX filesystems to create a global file system.
- Logical name structure. Use /.. To get out of current context. All other names are relative to a given context (which resides on different servers)
- LOCUS:
  - Distributed storage with location transparency
  - Replication via primary copy scheme - we will see more of these later on
  - Access synchronization: UNIX semantics
  - Uses CSS and SS to centralized some compoents

Distributed storage

- Storage scope and requirements are exploding (courtesy: Garth Gibson, keynote FAST '04)
  - High performance computing: may require 100 GB/sec/TFLOP
  - Commercial media applications: 1.2 GB/sec
  - Consumer media market: TIVO, iPod etc.
- Legal requirements such as Sarbane-Oxley act defines liability for archival storage
- Typical desktops finally have plenty of usable storage that can be used via broadband connectivity by others

How much information are we generating

- Print, film, magnetic, and optical storage media produced about 5 exabytes of new information in 2002.
  - Ninety-two percent of the new information was stored on magnetic media, mostly in hard disks.
- telephone, radio, TV, and the Internet -- contained almost 18 exabytes of new information in 2002, three and a half times more than is recorded in storage media. Ninety eight percent of this total is the information sent and received in telephone calls - including both voice and data on both fixed lines and wireless
- 5 Exabytes: All words ever spoken by human beings

- Internet archive: (wayback)
  - Capture all pictures on the web for storage challenge
  - Currently about 1 PB
- Sanger Inst. - Genome data
  - ~ 2TB/wk

Seagate profile

- by 2006, the worldwide market for hard disc drives will surpass 350 million drives.
- In fiscal year 2004, Seagate shipped:
  - 6.6 petabytes of total storage
  - 6.3 million consumer electronics drives
  - 10.3 million Enterprise drives
  - 3.3 million 15K RPM drives
  - 3.6 million mobile drives
  - 59.0 million personal storage drives

Why distributed storage?

- Its increasingly difficult to deliver last amounts of storage to a number of clients. Distribution allows for scalability

- In this class, we will focus on autonomous and distributed storage (unlike storage area network style storage)

Important Challenges

- Naming and location
  - The scale of storage affects how objects occupy the namespace
- Consistency and replication
  - Tradeoff between consistency and replication performance
- Storage management
  - Self managing important when the number of component increases
- Security
- Peer-to-peer and sensor storage
- Other concerns (Energy, Archival storage)

Who should take this course?

- This is an advanced “systems” graduate level course. You should have a good graduate-level background in OS/distributed systems/Computer Networks or other related course
  - The course is organized around reading research papers and a course project
- You should take this course if you are interested in learning about large scale distributed storage.
- Are there any special topic wishes from the students?

Course logistics

- Course project (group): 60%
  - Ideally a project that is related to your own research. Ideally (with some extra work) can be turned around to a conference publication
  - Three milestones: Goals and objectives (due soon), mid-semester status report (complete with your predicted graphs etc.) and final presentation and report.
- Paper summaries (group): 30%
  - Due by 8:00 pm of the previous day. One page ASCII.
    - One superficial review = warning
    - Two superficial reviews - you need to see me
- Class participation: 10%

Grapevine

- Grapevine was a distributed mail store built in Xerox PARC in early 80'. In some ways, Grapevine is still ahead of where we are now.(e.g., Grapevine was aware of message ordering) Grapevine was also used as an RPC mechanism; using mail type messaging forces consistency problems
  - Their problem was that their servers were just not good enough (5 MB storage)
  - Many of these authors made fundamental contributions in systems research

Cedar

- Immutable, file level shared distributed file system
  - Remote disks
  - Remote blocks (NFS)
  - Remote files (Cedar)
  - No cache consistency problem because files are immutable, updates are via versions