Distributed File Systems

Rick Cox

UW CSE

What

- Allow client machines to access files on a server.
- One of the most widely deployed distributed systems; every company has them.
- More interesting to study than some other distributed systems because DFS must support updates.

Why

- Management: Easy to maintain, easy to backup.
- Speed:
  - Can afford fast disks and RAID for file server.
  - File server has GBs of RAM to cache files - faster than local disk.
- Convenience: Users can access files from any machine and can share files easily.

How

RPC:

- Where do we put the client?
  - Generally, underneath the VFS layer.
- Server?
  - Can be either a user-level application (daemon) or a kernel-level service.

Basic Distributed Systems Issues

- Lack of global state
  - Provably impossible to know exactly what state all components are in.
  - So can’t directly know whether it is safe to update a file.
  - And in general, want to keep communication to a minimum.
- Security
  - Networks have no central controller (like a kernel) that can enforce global policy.
  - And no direct way to enforce policy dependent on global state.
  - Next lecture.

Issues, contd.

- Partial Failure
  - If one of the departmental file servers goes down, I want to keep working.
  - Preferably, if any go down, keep working.
- Scalability
  - Can (and do) connect lots of machine to a network.
- Naming
  - What is an object called on my computer?
  - How is an object located given a name?
- Transparency
  - Want the system to be invisible to the user.
  - Or do we?

Consistency

- What was the single-system solution?
  - Synchronization primitives.
- What makes that hard in a distributed system?
  - No obvious primitive atomic operations to build on.
  - Partial failure.

Consistency Solutions

- Reduce guarantees (do nothing)
  - NFS
- Prevent conflicts
  - AFS, Sprite FS
- Resolve conflicts
  - CVS, Coda
  - Can’t be guaranteed to work; some conflicts irreconcilable.

Availability

- We would like the system to be highly available in the face of failures.
- Replication is the key to availability.
  - Can also increase speed of reads - go to fastest/closest replica.
  - But introduces more consistency problems.

Partitions

- If the system is separated into two (or more) components.
- Network failures
- Mobile devices

CAP Theorem

Any given distributed system that supports non-trivial updates may be any two of: consistent, available, and correct in face of partitions.

- Consider different combinations.
- Almost always need correctness in partitions.
- So get to chose consistency or availability.