Discussion Section 9

Distributed Systems
Spring 2014

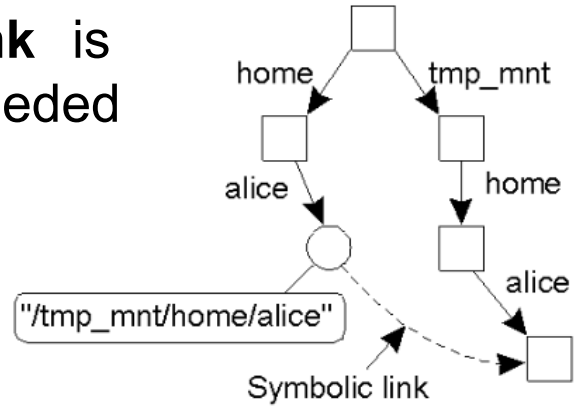
Problems Chapter 11

Automounting and symbolic links

Q: Using an automounter that install symbolic links makes it harder to hide the fact that mounting is transparent. Why?

A: If you recall, for performance and overhead reasons, directories are mounted in on-demand, by means of an **automounter**, instead of mounting all possible files when a user logs in.

The problem now is that the real directories are not what you think they are ("/tmp_mnt/home/alice" instead of "/home/alice") and a **symbolic link** is created between the two. Support from OS is needed to hide this.



Problems Chapter 11

Write denial in NFS

Q: Suppose that the current denial state of a file in NFS is WRITE. Is it possible that a previous client successfully opened that file and then request a write lock?

Ans: Yes, as locking is completely independent from the share reservations.

For example, the first client can request read/write access (value=BOTH) and with no denials (value=NONE). Then a second client request write access (value=WRITE), which will be granted as the first client had no denials. If the first client requests and acquires a write lock, the second client will not be able to write until the lock is released.

Cache Coherence Protocols

- With **read-only** caches, update operations are performed only at the server
- With **write-through** caches, the client can modify cached data and then forward the updates to the server
- With **write-back** caches, the client delays the propagation of updates by performing several writes before informing the servers

Problems Chapter 11

Cache Coherence

Q: What kind of cache coherence protocol does NFS implement?

Ans: Write-back Cache

Because multiple write operations can take place before cached data is flushed to the server, NFS clients implement a write-back cache.

Problems Chapter 11

Coda File System

Q: Explain how Coda solves read-write conflicts on a file that is shared between multiple readers and only a single writer.

Ans: Recall that Coda **transfers** an entire **copy** of a file to the client's machine **every time** a client **successfully opens a file**. Coda relaxes the ACID properties of real transactions by allowing multiple readers but a single writer. It uses the notion of **sessions**, which are treated in a **transaction** fashion.