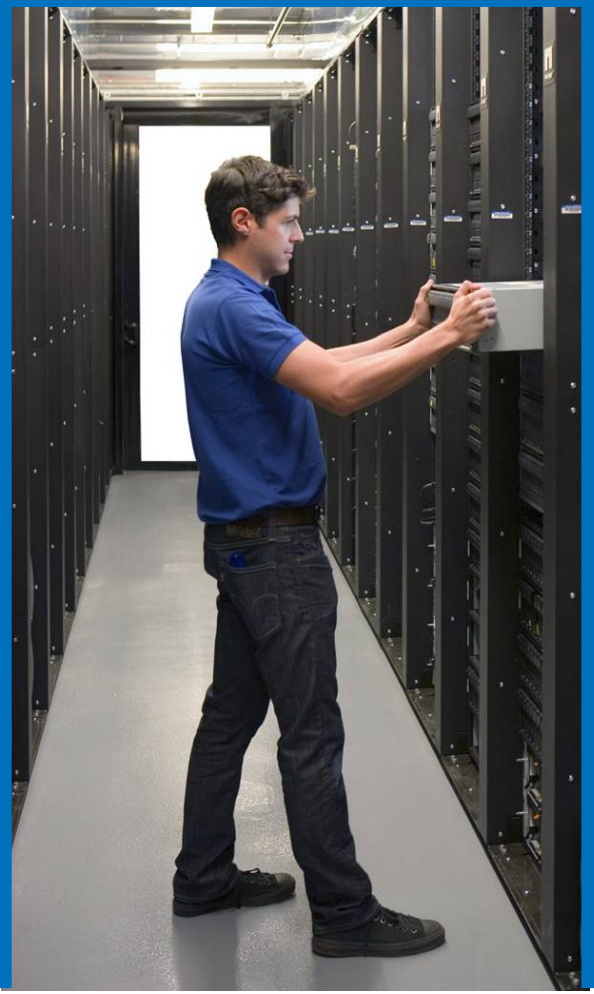
NetApp®

Go further, faster®

Layoutcommit and cache consistency

Trond Myklebust



Outline

- A few preliminaries
- Problem statement
- Client side solutions
- Server side solutions

A few preliminaries

- Cache consistency requires that any visible data changes made to the file MUST be accompanied by a change attribute update.
  - Otherwise, a client that OPENs the file for reading may believe that its page cache contents are still valid.
- In practice, the close-to-open cache consistency model should allow us to defer the change attribute update until the writer calls LOCKU, OPEN_DOWNGRADE or CLOSE.
  - pNFS relies on this behaviour, and requires the client to issue LAYOUTCOMMIT before CLOSE

Problem statement

- What happens if a client modifies the file via pNFS, but dies before it can issue the LAYOUTCOMMIT?
  - The file may have changed on the server, but close-to-open cache consistent clients may not be able to detect the change.
  - Backup programs may no longer work as expected.

Client side solutions

- Ditch the close-to-open cache consistency model, and only cache data when the client holds a delegation.
- Problems:
  - RFC5661 promises that pNFS supports close-to-open in section 13.10
  - Legacy NFSv2/v3 clients have no delegations
  - Legacy NFSv4 clients already rely on close-to-open, and would have to be modified to work with pNFS setups.
  - Eliminates the possibility of using cachefs-style persistent caches.

Server side solutions

- Upon receiving an OPEN, LOCK or WANT_DELEGATION request from a new client, if the MDS may check whether clients that have a layout and are holding the file open for writing have sent a LAYOUTCOMMIT and initiate recovery if they have not.
  - Note that this does not work with legacy stateless NFSv2/v3 clients: check on GETATTR and LOOKUP instead?

Server side solutions (cont)

- Recovery methods depend upon the nature of the data servers:
  - For block servers, you may need to always assume the file has changed if someone holds a layout
  - Object and file servers might be able to maintain change attributes on the data servers