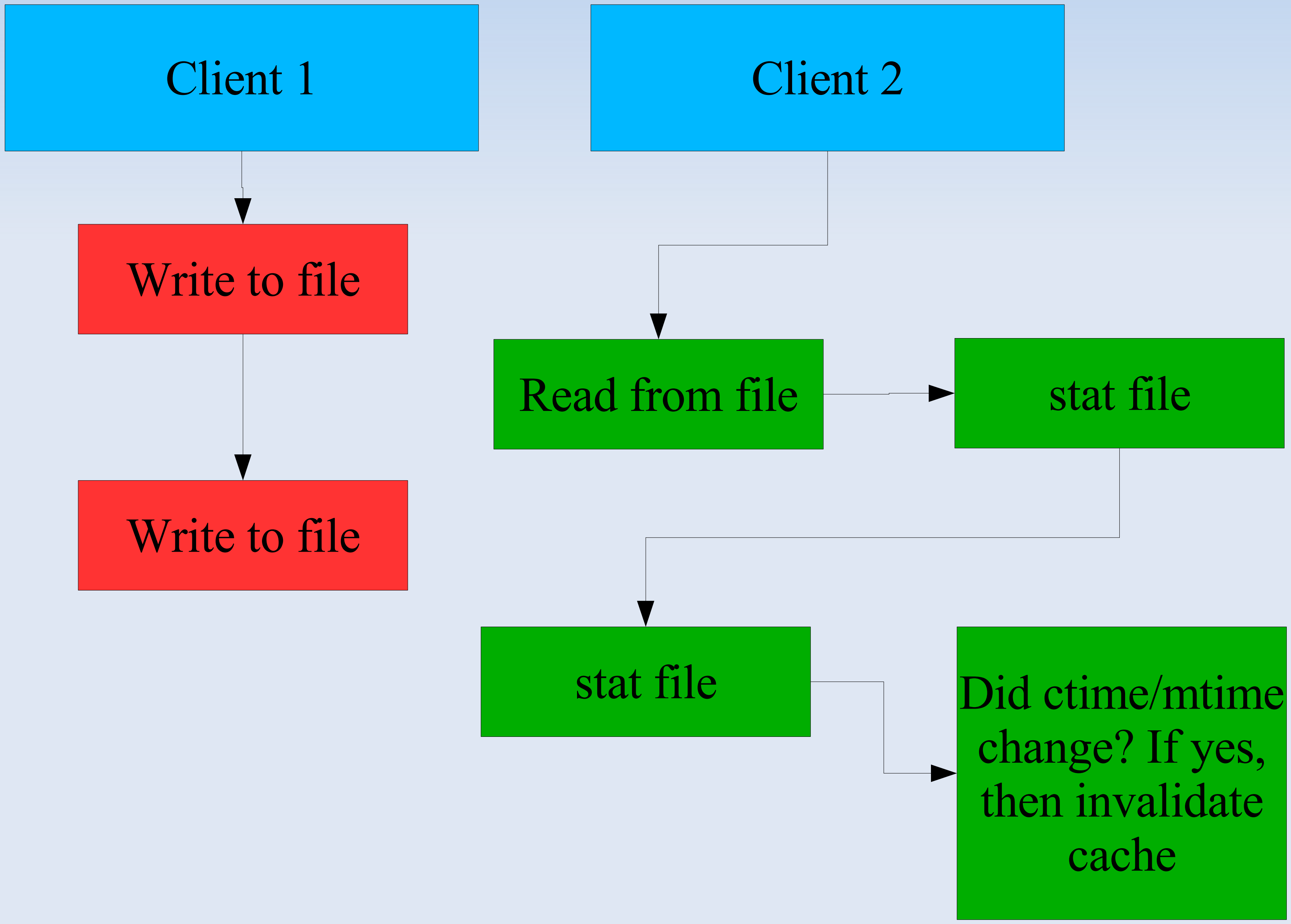
NFS topics in Linux

- Server issues
  - Filesystem support for a “change attribute”
  - Exporting cluster coherent locks
- Client issues
  - Scalability
  - Optimisations
  - Migration/replication
- ACLS

Server issues: Need for a “change attribute”



Change attribute

- Nanosecond time granularity does not resolve the race
  - Actual timesource is just jiffies
  - Would prefer to close the window permanently
- NFSv4 change attribute requirements
  - Must change whenever ctime changes
  - Must be consistent across server reboots
  - Need not be consistent across files
  - No units

NFSv4 ACLs

- The only ACL model supported by NFSv4 is “NFSv4 ACLs”
  - Based on Windows ACL model
  - Have more permission bits (append, r/w, ...)
  - Have ALLOW and DENY entries
  - Extremely general. POSIX acls can be mapped into NFSv4 but converse is not true.

NFSv4 ACLs server side

- Export POSIX ACLs as NFSv4 ACLs, map between the two. (What we currently do.)
- Support full NFSv4 ACLs in local filesystems. (Experimental ext3 patches exist.)
- Add POSIX ACLs to the NFSv4 protocol. (No other OS wants this. It would probably be linux-only.)

NFSv4 ACLs client side

- kernel exports only pure NFSv4 ACLs
- userspace has pure NFSv4 ACL editors, also optionally maps between POSIX and NFSv4 (using unmerged libacl patches).

Exporting cluster-coherent locks

- There already is a file ->lock operation; we need to:
  - modify NLM and NFS code to call it.
  - allow asynchronous return of results.

Exporting cluster-coherent locks

- Proposed asynchronous API:
  - static int callback(struct file_lock *, struct file_lock *, int) {
  - ...
  - }
  -
  - lock.fl_notify = callback;
  - error = vfs_lock_file(file->f_file, F_SETLK, &lock);
  - if (error == -EINPROGRESS)
  - callback() will be called with results

Client scalability

- readdir() scalability issues
  - Get rid of redundant storage
  - dentry+cookie lookup table?
- VFS: Add further intents?
  - Get rid of redundant lookups in operations like rename(), link()