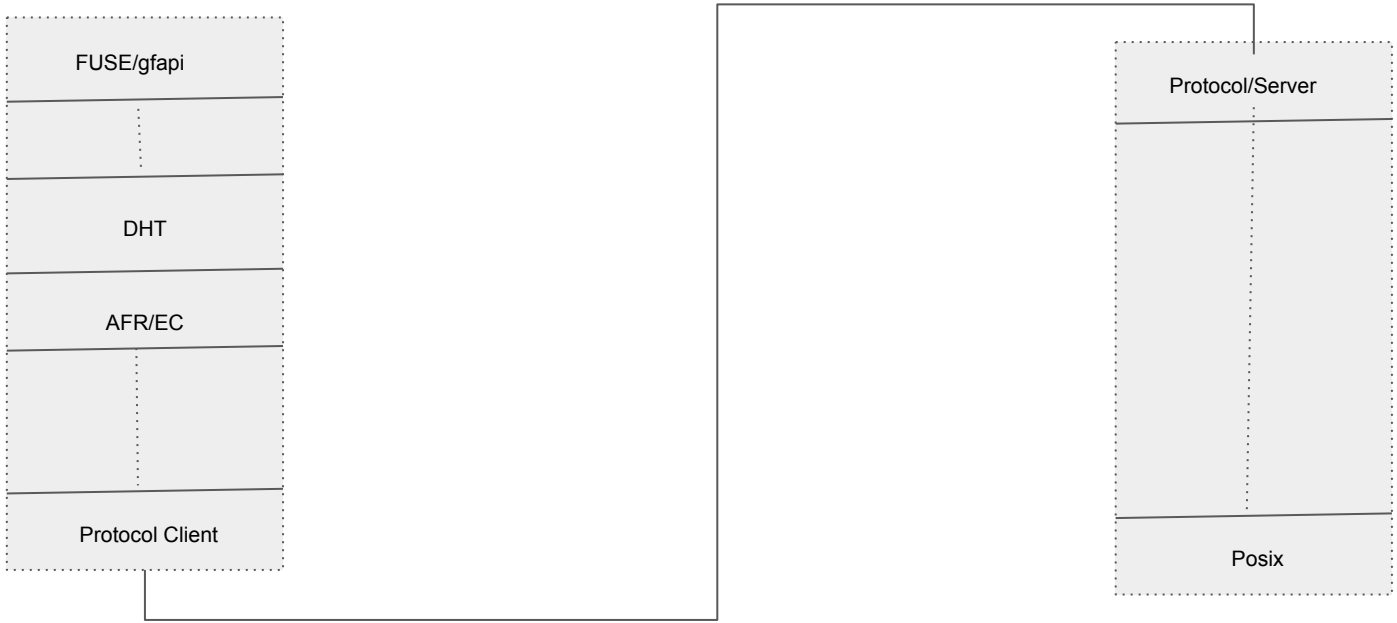
# Native Clients with GFProxy

Poornima G
Vijay Bellur

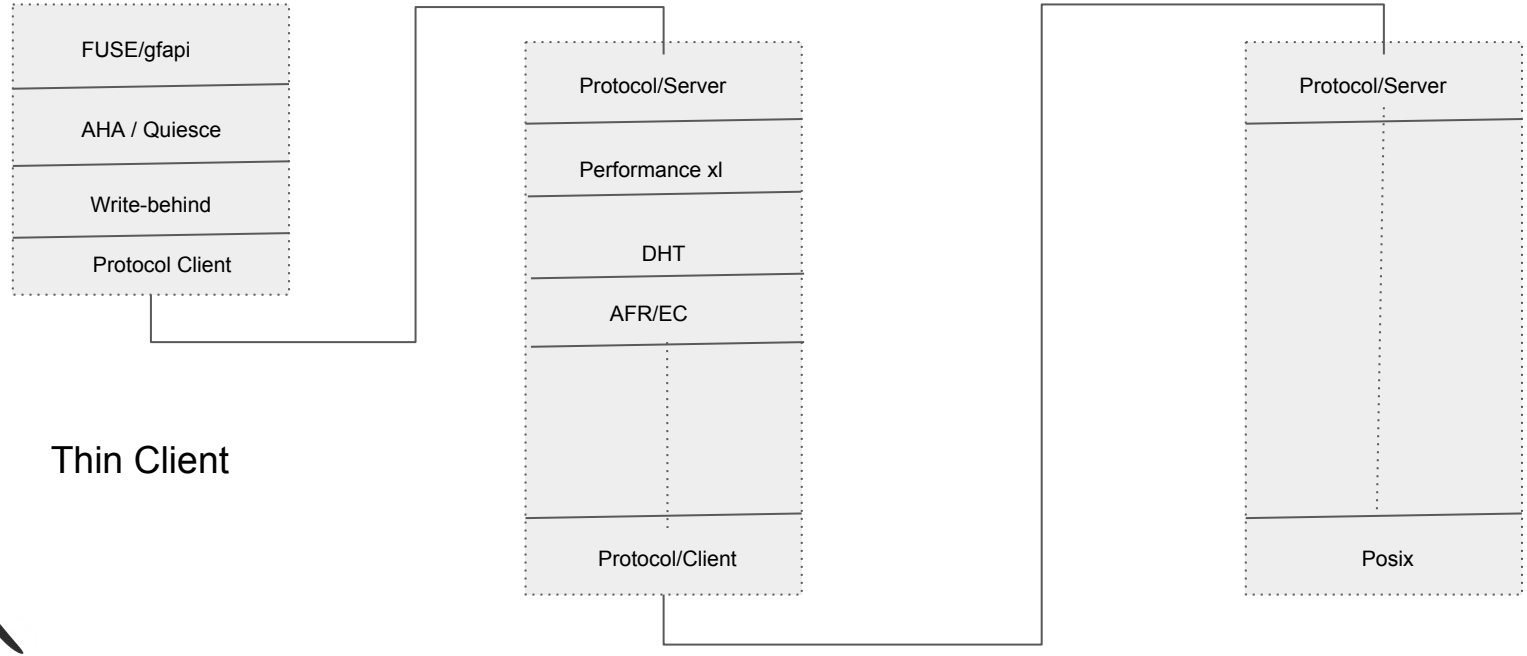
# Current Architecture




# Cons of client driven distribution and replication

- Upgrade on all client machines is difficult
- In case of multiple clients on the same system, the resource consumption of client processes is duplicated. Eg: Samba, Containers native storage, Qemu etc.
- Bandwidth consumption on the client.

# Hence GFProxy




# Current state of GFProxy

- Already done:
  - Glusterd changes - volfile generation, daemon management, port mapping.
  - AHA / Quiesce for High Availability.
  - Targeted for 3.13 as an experimental feature.
- To be done:
  - Dynamic graph switch
  - Volume multiplexing
  - Glusterd2 integration - volfile gen, daemon management, port mapping, gfproxy on subset of nodes etc.
  - Reduce the memory and thread consumption on thin clients.
  - Allow gfproxyd to be run on non-trusted storage pool nodes, for performance reasons.

Issue #242 for details on each of these items

# Native Clients with GFProxy

- GFProxy provides a thin client xlator stack
- Easy to have more native clients - for Mac, Windows and …!
- Fuse for Windows - https://github.com/billziss-gh/winfsp
- Fuse for Mac - https://osxfuse.github.io

Questions?