The Envoy File System:
Structural Overview

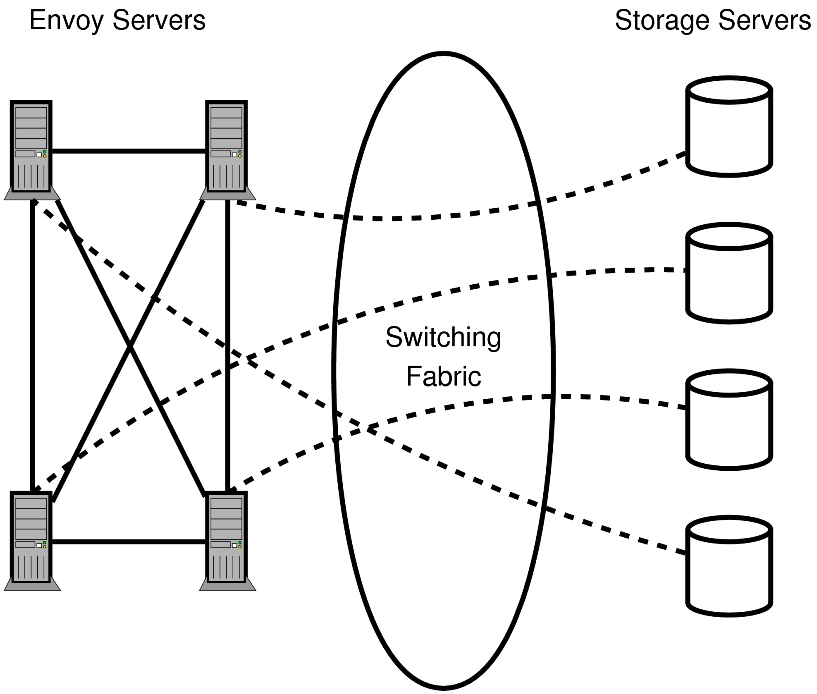
Russ Ross

russ@russross.com

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Big picture

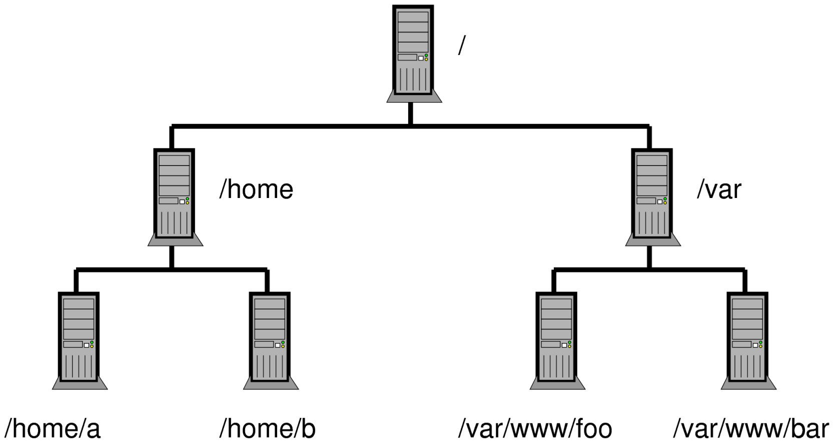
- Envoy servers cluster together along shared boundaries
- Storage servers are independent of each other
- Envoys always connect directly to relevant storage and envoy servers: no hierarchy at the network level



UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Envoy organisation

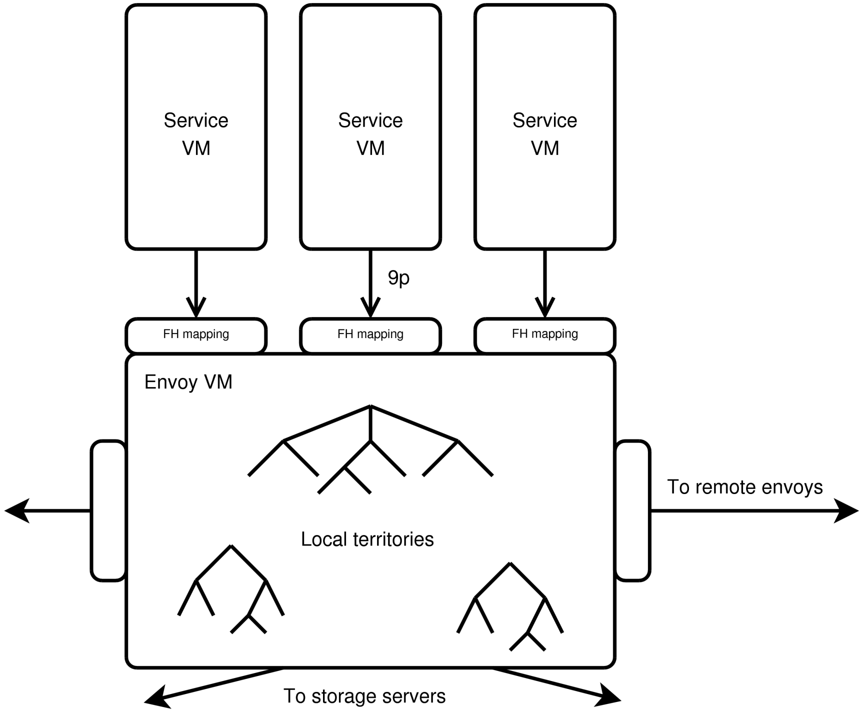
- Territories are branches of the global namespace
- Connections are maintained between envoys only when:
  - They are territory neighbours
  - One has a file open in another's territory
- Connections indicate sharing/overlapping interests



UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Envoy node

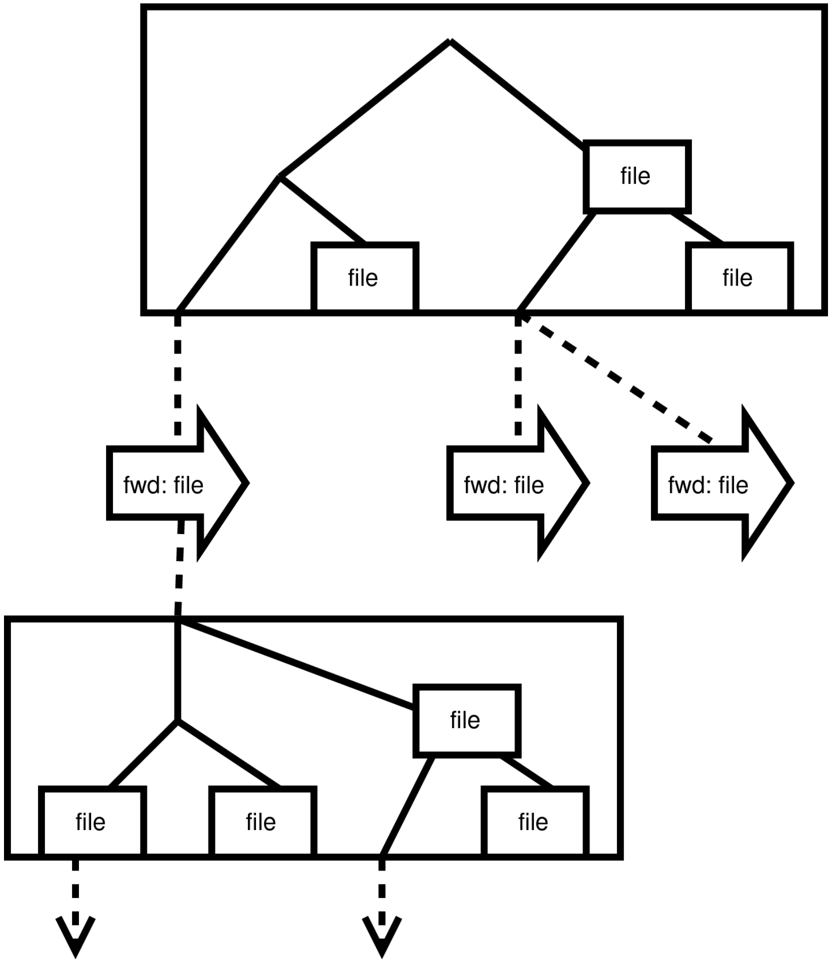
- One per machine
- Persistent cache
- Server to local VMs
- Manages state for all local file handles
- Owns territories
- Serves all requests for local territories
- Forwards requests for remote territories
- Connects to storage



UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Territories and claims

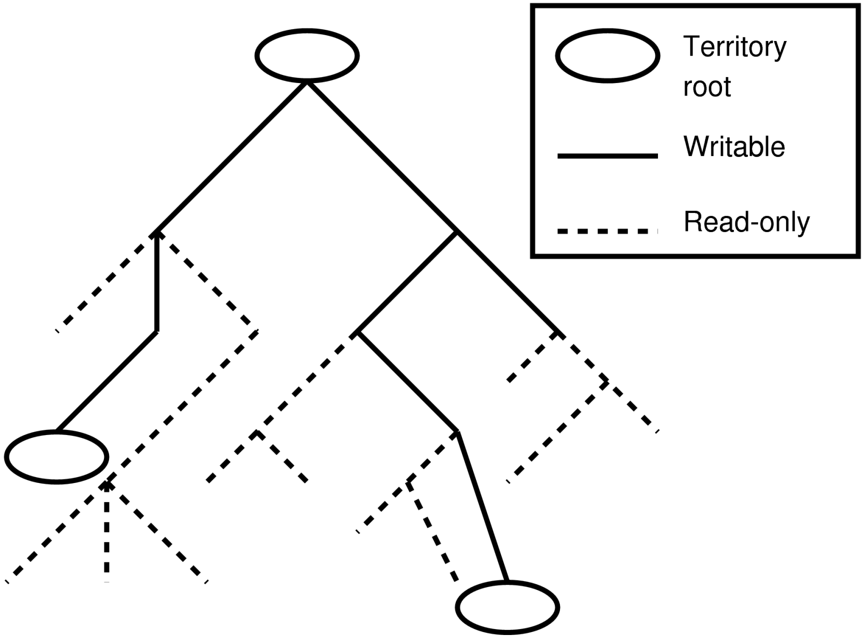
- A tree connecting claims (handles to active objects) overlays each local territory
- Open files and territory boundaries are considered active objects
- A claim is a globally unique gateway to an object: synchronizing object access is trivial



UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Copy-on-write

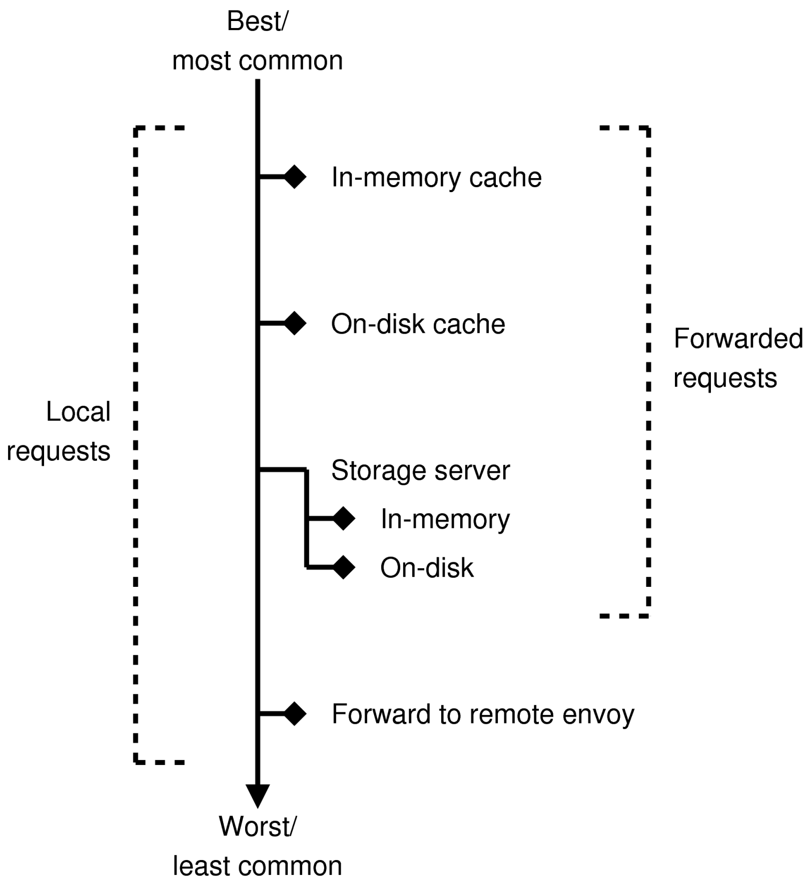
- Territory roots are always writable
- The path to any writable object is also writable
- Writable objects only have a single name
- Read-only objects may have multiple names from image forks



UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Data paths

- Most common data paths are also the shortest
- Longest paths only happen due to runtime sharing
- Territory change algorithm designed to shorten average system-wide path length



Summary

- The good:
  - Alignment of interests
  - Private images perform like local server
  - Shared images fastest for heaviest users
  - Sharing is consistent
  - No significant trust granted to users, but root-like image control
  - Fast response to major changes, avoid thrashing for minor changes
  - Minimal runtime links between envoy nodes
- The bad:
  - Nearest cache is across a protection boundary from client
  - Persistent cache may be slower than striping from storage servers
  - Clients are dependent on a local envoy instance, and fail when it goes down
  - Single, global root node