ORACLE

Prototyping NFSv4 Migration

Chuck Lever
Consulting Member of Technical Staff

Sounding Our Recommendations

- Before advancing rfc3530-migration-update:
  - What part of this I-D has been prototyped and tested?
  - What has not been prototyped, and why not?
  - What interesting non-protocol issues have arisen in our prototypes?
    - After fixing the protocol, is I-D complete?
- Note that NFSv4.1 migration is largely untested, and is not in the scope of rfc3530-migration-update

rfc3530-migration-update

Prototype Coverage

- Implementation of specific recommendations
  - SETCLIENTID with Uniform Client Strings
  - Some aspects of server trunking detection
  - Location discovery accompanied by RENEW
  - FSID presence checking accompanied by RENEW
  - NFS4ERR_DELAY on FH-bearing operations

rfc3530-migration-update

Prototype Coverage

- Testing
  - TSM in common usage scenarios
  - Non-TSM in a few common usage scenarios
  - Migration of Kerberos-protected shares
  - Handling NFSv4 delegation during migration

rfc3530-migration-update

Prototype Gaps

- Lease merging on destination server
- Full implementation of trunking detection as described
- Multiple FSID presence tests per compound
- Using zero-length READ as TEST_STATEID substitute
- NFS4ERR_DELAY on RELEASE_LOCKOWNER

Why TSM is a Big Deal

- Data-only migration
  - After a migration, client performs state recovery to inform destination server of application lock state
  - Server SHOULD provide a grace period to prevent theft of lock state while recovery occurs
  - Time-limited grace may prevent full recovery or hold up application operation
- Transparent State Migration
  - Destination server merges migrated lease with client's existing lease, if there is one
  - The client can resume operation against the destination server without CLAIM_PREVIOUS OPENs or a grace period
  - *No window for lock state loss*

Client Implementation Challenges

- Trunking detection
  - Visited in previous presentations
- Redirecting the RPC transport
  - Or, how to provide NFSv4.1+ session semantics in NFSv4.0
- Referrals v. migration
  - NFS4ERR_MOVED triggers both, but how does a client tell them apart?
- Lock state representation

Handling NFS4ERR_MOVED

NFS4ERR_MOVED

Establishing contact

- With either migration or referral, destination server may be unfamiliar to client
  - Trunking detection may be needed
  - A fresh lease may be established
  - Security negotiation may be needed

NFS4ERR_MOVED

File system data structures

- *Referral*, like `mount`, materializes a local superblock
  - No other data or metadata yet exists
- *Migration recovery* must preserve existing superblock, inodes, and open/lock state data
  - Including delegated state
  - Non-TSM relies on client to preserve open/lock state

NFS4ERR_MOVED

Client's filename space

- *Referral* alters client's filename space (a.k.a. `mount`)
  - Occurs only on LOOKUP-like operations and READDIR
- *Migration recovery* does not change the namespace
  - Implementation handles only leaf exports for now
  - Client may not follow path from destination's pseudo-fs
    - Client may perform FSINFO or acquire lease expiration as part of following pseudo-fs path, won't necessarily do this automatically as part of migration recovery

NFS4ERR_MOVED

Recovery serialization

- *Referral* is handled synchronously in user context
  - Treated like an automount operation, not a recovery
- *Migration recovery* must be serialized with state recovery
  - Therefore it's handled in state recovery FSM
  - Blocks all operation against source server
  - Like state recovery, may occur on nearly any operation

Representing Lock State

Representing State

Current recommendations

- Data structure organization
  - RFC 3530bis, section 9.1.3: “stateids associated with a given client ID are associated with a common lease”
  - Thus, clients typically group state IDs under each client ID
  - Servers may also take this approach
- TSM moves state IDs for one FSID to another lease
  - May require significant data structure re-engineering

Representing State

Protocol implications

- To support TSM, state IDs should be unique across servers
  - 9.1.3: “Each stateid must be unique to the server. Many operations take a stateid as an argument but not a clientid, so the server must be able to infer the client from the stateid.”
  - Server discards migrated state if a state ID collision occurs

Representing State

Protocol: Good news, bad news

- In NFSv4.0, all state-bearing ops also carry a FH
  - NFSv4.1 FREE_STATEID is a fly in the ointment
  - Different state ID structure for NFSv4.0 is required
- NFSv4.0 state IDs:
  - Must contain a client ID
  - Can rely on operation arguments for FSID/FH
- NFSv4.1+ state IDs:
  - Can rely on SEQUENCE operation to identify the client
  - Must contain FSID/FH (if TSM is supported)

Transport Management

Transport Management

Challenges

- Migration recovery requires exclusive access to RPC transport
  - How do we suspend NFS tasks without adding overhead during normal operation?
  - How do we avoid deadlocks?
- Replacing a mount point's RPC client context is unsafe on Linux
  - Therefore, existing context is modified in place
  - Only the underlying transport is replaced

Transport Management

Order of operation

- 1. Operations receiving MOVED are parked
- 2. Location discovery is performed
- 3. RPC transport is switched to the destination
  - The part of the proof where “miracles occur”
- 4. Parked operations awoken, retried
- 5. Client then determines whether its state is still valid

Transport Management

Switching RPC transport

- Step 3 from the previous slide, in detail
  - a. Transport to source server is drained
  - b. Client acquires fresh transport to destination server for the migrating mount point
  - c. Trunking detection, lease establishment may be performed
  - d. FSINFO probe retrieves destination's lease time, *etc.*
  - e. Source transport is unblocked

Transport Management

Slot table implementation

- NFSv4 slot table used to suspend NFS operations
  - One slot table manages all operations for each server
  - No server feedback on table size, always 1024
  - Same performance overhead as NFSv4.1 session
  - Same drain/wake-up points in state recovery as NFSv4.1
- New RPC transport replaces source's RPC auth cache
  - GSS contexts are specific to a client-server pair
  - Security re-negotiation may occur

Take-away

- rfc3530-migration-update is well-covered
- Some important items remain untested
  - Lease merging (server)
  - Testing individual state IDs (may be unnecessary)
  - DELAY on RELEASE_LOCKOWNER (new)
- Significant client redesign is required to support migration recovery and TSM
- No new protocol-related issues were identified

Questions / Discussion