

Recovery and operator lifecycle under R4

These operations implement the existing R4 requirements. They do not establish completion of the remaining release criteria. Fixed membership, coordinated maintenance and authenticated non-Byzantine voters are the supported scope.

Backup and verification

Stop the source voter, then run:

```
sqlodin backup SOURCE-NODE.json NEW-BACKUP-DIRECTORY
sqlodin verify-backup NEW-BACKUP-DIRECTORY
```

The source must use storage format 5. Its stable voter lock excludes a running service; the other two voters may continue serving. The destination must not exist. The operation copies the applied application through SQLite's backup API, including the empty database, and syncs the image, manifest and directory before success. Verification checks the bounded binary manifest, engine/configuration identity, image SHA-256 and length, application metadata, integrity and foreign keys. Acceptor-local tables, symlinks and unexpected database sidecars are rejected. Preserve failed destinations for diagnosis and use a new destination for retries.

This is a consistent point-in-time backup, not a quorum certificate or a promise that it includes later acknowledgements. For a lossless recovery cut, stop new client work, resolve in-flight outcomes, obtain a fresh quorum read on the backup source, stop it and take the backup. Keep the old deployment fenced thereafter. A successful local status request does not establish a fresh read or readiness.

Fenced replacement and restore

For a lost voter or disk, the supported replacement procedure recovers the fixed group into a new namespace. Preserve surviving source directories and the backup. Fence every old process and its restart automation. Provision new configuration files with a globally unused cluster name, valid fixed membership, credentials and new, nonexistent data directories. Restore the identical verified backup on every voter before starting the group:

```
sqlodin restore BACKUP NEW-NODE.json --new-cluster
sqlodin serve NEW-NODE.json
```

Verify readiness with a fresh `SELECT 1` through each endpoint. The CLI rejects the source cluster name and an existing destination. Globally unused names and fencing are operator obligations: a local command cannot discover every previous deployment. Do not copy a donor consensus journal or reuse an old acceptor identity with forgotten promises. Live same-namespace voter replacement and membership resizing are outside the fixed first-release scope.

Restore retains the application prefix `P`, retry session results/fences and transaction revision, while dropping disposable per-slot outcome cache rows. No donor promises, accepted votes or reserved IDs are imported. Fresh consensus starts above `P`. Keeping `P` ensures that retained retry outcomes do not refer to a future slot. A digest of the verified backup is stored in `_sqlodin_genesis` and bound into the store/configuration identity and peer fingerprint. Empty or inconsistent genesis states cannot communicate as compatible voters. The old namespace cannot rejoin the new one. Compatible new voters use ordinary Paxos for all later slots and certified snapshots for later history retirement.

The destination initially carries an incomplete identity. Restore verifies an exact byte copy, rebases application identity, syncs files and directories, then publishes readiness in a `FULL` consensus

transaction. Startup rejects an incomplete destination. Genesis survives compaction, generation publication and restart. Never interpret a partially restored directory as an empty new voter.

Coordinated upgrade and rollback

Retain the exact old binary and original configurations/directories. For a format-4 source, explicitly configure `storage_format: 4`; omitted format now selects 5. Quiesce clients, establish the recovery cut, stop all voters, then run `sqlodin migrate OLD-NODE.json NEW-DIRECTORY` separately for every voter. Source database and WAL bytes remain unchanged. Validate the new configurations and start the entire group with the new compatible binary and format-5 directories. Peer fingerprint changes require this coordinated upgrade; rolling compatibility is not promised.

Rollback to the old binary and original source is supported only before any new voter is activated. If the source resumes, its earlier migration copies become stale: never activate those copies. Stop the source again and migrate into fresh directories. After activation or acknowledgements from the upgraded group, do not roll back to stale source state. Use a compatible forward fix, or the verified backup and fenced new-namespace recovery procedure above with an explicit cut.

Certificate renewal, identity and readiness

Use exact configured DNS identities and matching CA, certificate and private-key files. Startup rejects an expired/not-yet-valid local leaf with a certificate-date hint. Peer/client validity and trust are checked at TLS handshake. Existing TLS connections do not automatically terminate when a certificate later expires.

For the supported coordinated renewal, quiesce clients and stop all voters and clients; this closes existing connections. Issue a new CA and leaf/key pairs with the same configured DNS principals, update the CA/certificate/key paths on every voter and client, then restart and require fresh quorum reads on all endpoints. Keep the data, namespace and membership unchanged. Remove the old trust anchor; old credentials must fail authentication. Preserving the authenticated client principal preserves its retry fence across renewal. Protect private keys and retain recovery material according to the deployment's backup policy.

Membership or engine identity mismatches refuse startup instead of rewriting stored identity. A listening minority can answer status but cannot pass a fresh quorum read. Automated enrollment and rolling upgrades remain deferred.

Argument, implementation and evidence

`RestoredGenesis.tla` models the fresh initial-state seam: compatible peers have one initial image/prefix, readiness follows complete durable state, and retries retain valid origins. Its four negative controls exercise old namespace reuse, mixed images, early readiness and prefix reset. This finite model is not a proof of the complete Paxos implementation. R2 covers the composed protocol argument.

The code map is `backup_manifest.odin/backup_verify.odin` for the manifest and image, `genesis.odin` for the bound anchor, `restore_copy.odin` for checked copying, `restore_publish.odin` for the final readiness transaction, and the generation build/recovery paths for persistent genesis. `transport/mtls/openssl.odin` validates local dates and authenticated handshakes. `service/config.odin` reports startup failures; fresh read barriers establish service readiness.

Evidence is under `benchmarks/results/verification-20260924/`: backup service checks and nine crash boundaries; restore service checks and eight crash boundaries; 146 Odin tests; five restored-genesis model configurations; certificate lifecycle checks; and retained-binary migration/rollback

checks. Reports preserve source/binary hashes, failures and the actual filesystem. SIGKILL evidence is process-crash testing, not physical power-loss certification. Durability assumes FULL journal commits and a filesystem/device honoring sync. The separated application database commits with WAL NORMAL; after power loss it is a committed prefix no older than its last checkpoint, and recovery replays the retained chosen suffix (SOD 0005 M4). No model proves the compiler, cryptography, OS or hardware. Final-candidate qualification and the release decision remain R7, without an elapsed soak gate.