QoreStor Refcount Script – Technical Function Reference & Execution Flow

# Purpose

This document provides a detailed, function-by-function description of the QoreStor refcount migration script, including the order of execution in each mode (scan-only, dry-run, live) and the rationale for that ordering. It covers inputs, outputs, side effects, error handling, and design decisions such as layout detection, logging, service control, rsync usage, and config editing.

# Execution Flow by Mode

## Common Startup (All Modes)

1) parse\_args → 2) setup\_logging → 3) detect\_layout\_once  
These steps establish runtime flags, bind stdout/stderr to the log, and decide Standard vs. Alternate layout (affecting REFCNT\_SUBPATH and repo resolution).

## Scan-only Mode (`--scan-only`)

main → parse\_args → setup\_logging → detect\_layout\_once → scan\_refcnt\_sizes → print\_summary → exit  
  
Rationale: The script performs only read-only inspection to estimate sizes and directories. No mountpoint, service control, or config changes are needed.

## Dry-run Mode (`--dry-run`)

main → parse\_args → setup\_logging → detect\_layout\_once → decide\_dryrun\_target → verify\_ready\_to\_stop → wait\_for\_service\_stop → copy\_all\_refcnt (simulated if no target) → verify\_all\_refcnt (dry) → dry\_run\_preview → (optional) start\_services (simulated) → print\_summary → exit  
  
Rationale: Dry-run proves feasibility and computes counts using rsync stats without modifying data. If no target is available, it downgrades to a plan-only path while still presenting scan and config preview to de-risk the live run.

## Live Mode (default)

main → parse\_args → setup\_logging → detect\_layout\_once → setup\_mountpoint → confirm\_live\_run → make\_backup → verify\_ready\_to\_stop → wait\_for\_service\_stop → copy\_all\_refcnt → (optional) verify\_all\_refcnt (if --checksum-verify) → apply\_changes → start\_services → print\_summary  
  
Rationale: Live mode is gated by a confirmation preview and a config backup. Services are stopped before data movement to ensure consistency; verification can be checksum-strong if requested. Config edits are applied only after a successful copy and (optional) verify, then services are restarted and health-checked.

# Function Reference

## usage

Purpose: Prints CLI usage/help.

Inputs: Reads none.

Outputs/Side Effects: Writes to stdout; no side effects.

## banner(text, color)

Purpose: Pretty banner printer with ANSI colors.

Inputs: text, color optional.

Outputs/Side Effects: Writes to stdout.

## decide\_dryrun\_target

Purpose: Interactive prompt (dry-run) to determine if destination is mounted.

Inputs: Reads DRY\_RUN; prompts user; may call setup\_mountpoint.

Outputs/Side Effects: Sets DRY\_HAS\_TARGET, DRY\_SKIP\_SERVICES, MOUNTPOINT; logs via debug\_log.

## capture\_system\_info

Purpose: Shows basic system info using `system --show`.

Inputs: Requires `system` command.

Outputs/Side Effects: Writes filtered info to stdout; no mutation.

## detect\_layout\_once

Purpose: Detects Standard vs Alternate layout from /etc/oca/oca.cfg.

Inputs: Reads CONFIG\_FILE; greps for PLATFORM\_DS\_REFCNTS\_ON\_SSD=1 and TGTSSDDIR=${R3\_DISK\_JOURNAL\_PATH}; extracts R3\_DISK\_JOURNAL\_PATH.

Outputs/Side Effects: Sets ALT\_LAYOUT, REFCNT\_SUBPATH, R3\_JOURNAL\_PATH; debug logs decision.

## detect\_alt\_repo\_from\_config

Purpose: Pure helper to parse ALT repo root from config.

Inputs: Parses CONFIG\_FILE for flags and R3\_DISK\_JOURNAL\_PATH.

Outputs/Side Effects: Prints path if satisfied; return 0/1.

## get\_repo\_location

Purpose: Resolves repo root path.

Inputs: If ALT\_LAYOUT=true → uses R3\_JOURNAL\_PATH; else parses `system --show`.

Outputs/Side Effects: Prints path; returns non-zero on failure; debug logs to stderr.

## to\_bytes(s)

Purpose: Best-effort parser to bytes from human strings.

Inputs: String with digits and IEC suffixes.

Outputs/Side Effects: Prints integer bytes.

## human\_bytes(b)

Purpose: Formats bytes using IEC suffixes.

Inputs: Integer bytes.

Outputs/Side Effects: Prints human-readable string.

## run\_with\_bar cmd...

Purpose: Runs rsync (or similar) and renders a progress bar.

Inputs: Command array; detects -n for dry-run and simulates progress.

Outputs/Side Effects: Manages temp files; returns wrapped command exit code; preserves shell strictness.

## simulate\_bar

Purpose: Simple progress bar generator for dry-run UX.

Inputs: None.

Outputs/Side Effects: Runs until killed; used for simulated rsync scans.

## debug\_log / debug\_printf

Purpose: Debug helpers that print to FD3 (original stderr).

Inputs: DEBUG\_MODE flag.

Outputs/Side Effects: Do not contaminate stdout in command substitutions.

## parse\_args

Purpose: Parses CLI flags and collects free args.

Inputs: Process argv.

Outputs/Side Effects: Sets DRY\_RUN, SCAN\_ONLY, VERIFY\_CHECKSUM, TEST\_MODE, DEBUG\_MODE, PARSED\_ARGS.

## setup\_mountpoint

Purpose: Normalizes and validates MOUNTPOINT; composes NEW\_LINE.

Inputs: PROMPTS if not provided; checks existence or mountpoint based on mode.

Outputs/Side Effects: Sets MOUNTPOINT, NEW\_LINE; may mkdir target /ssd in live mode.

## setup\_logging

Purpose: Initializes LOG\_FILE and logs both stdout and stderr separately to tee.

Inputs: Uses TIMESTAMP, LOG\_DIR; creates directory.

Outputs/Side Effects: Redirects FD1 and FD2; prints run banners; calls capture\_system\_info.

## get\_system\_state / get\_system\_reason

Purpose: Extracts system state and reason from `system --show`.

Inputs: Requires `system` command.

Outputs/Side Effects: Prints strings for other functions.

## verify\_ready\_to\_stop

Purpose: Pre-check info before stopping service.

Inputs: Reads states.

Outputs/Side Effects: Prints status; in dry-run records summary only.

## wait\_for\_service\_stop(service)

Purpose: Stops service and waits until System State=Stopped or timeout.

Inputs: Service name; STOP\_TIMEOUT.

Outputs/Side Effects: Calls systemctl stop; loops with status output; returns 0/1.

## start\_services

Purpose: Starts service and waits until fully operational.

Inputs: Service name; START\_TIMEOUT; relaxed reason string match recommended.

Outputs/Side Effects: Calls systemctl start; loops until Operational Mode and active.

## plan\_copy\_totals

Purpose: Counts files under each <id>/$REFCNT\_SUBPATH with find.

Inputs: Requires repo path and REFCNT\_SUBPATH.

Outputs/Side Effects: Prints total file count (present on source).

## safe\_rsync args...

Purpose: Wrapper enforcing -n in DRY\_RUN to prevent accidental writes.

Inputs: DRY\_RUN flag; rsync args.

Outputs/Side Effects: Calls rsync; returns 99 if dry-run missing -n.

## scan\_refcnt\_sizes

Purpose: Scans each integer directory and sums `du -sb` of $REFCNT\_SUBPATH.

Inputs: Repo location and REFCNT\_SUBPATH.

Outputs/Side Effects: Sets SCAN\_FOUND, SCAN\_TOTAL\_BYTES; prints per-dir sizes and total.

## check\_free\_space(target, need)

Purpose: Validates free space using `df -PB1`.

Inputs: Target base path and required bytes.

Outputs/Side Effects: Prints free/required; returns 1 if insufficient.

## rsync\_base\_flags / rsync\_verify\_flags

Purpose: Emit base rsync flags and verify flags.

Inputs: VERIFY\_CHECKSUM for verify\_flags.

Outputs/Side Effects: Used by copy/verify calls; include -aHAX, --numeric-ids, --sparse, -W, --dirs.

## copy\_one\_refcnt(SRC, DST, base)

Purpose: Copies a single refcnt subtree from SRC to DST.

Inputs: Auto-normalizes SRC/DST to append REFCNT\_SUBPATH if needed.

Outputs/Side Effects: Uses safe\_rsync in dry-run; run\_with\_bar + rsync in live; updates COPIED\_FILES and SUMMARY.

## verify\_one\_refcnt(SRC, DST)

Purpose: Verifies a single refcnt subtree.

Inputs: Uses rsync verify flags; optional --checksum.

Outputs/Side Effects: Non-zero on mismatch; summarizes files compared.

## copy\_all\_refcnt()

Purpose: Bulk copy of all refcnt files using --files-from list.

Inputs: Builds list with `find . -type f -path \*/$REFCNT\_SUBPATH/\*`.

Outputs/Side Effects: Dry-run: simulated bar; Live: run\_with\_bar; sets COPIED\_FILES; uses check\_free\_space first.

## verify\_all\_refcnt()

Purpose: Bulk verify of all refcnt trees.

Inputs: Loops per <id>; mkdir -p dest refcnt path; uses rsync verify flags.

Outputs/Side Effects: Totals verified files; summary lines.

## config\_preview\_live()

Purpose: Shows what config edits will be applied in live mode.

Inputs: Uses NEW\_LINE, REFCNT\_OLD/NEW.

Outputs/Side Effects: Prints intent; no file changes.

## dry\_run\_preview()

Purpose: Generates a temp preview of config edits and shows colored diff.

Inputs: In dry-run, may use placeholder for NEW\_LINE if no mountpoint.

Outputs/Side Effects: Ensures replace-or-insert behavior for TGTSSDDIR; dedup for PLATFORM\_DS\_REFCNTS\_ON\_SSD.

## make\_backup()

Purpose: Creates timestamped backup of config.

Inputs: CONFIG\_FILE, BACKUP\_DIR.

Outputs/Side Effects: Sets BACKUP\_FILE; prints path.

## apply\_changes()

Purpose: Applies config edits safely using awk transform.

Inputs: Replaces existing TGTSSDDIR or inserts after TGTDIR; toggles PLATFORM\_DS\_REFCNTS\_ON\_SSD to 1.

Outputs/Side Effects: Writes CONFIG\_FILE via .tmp; shows diff vs backup.

## print\_summary()

Purpose: Prints end-of-run summary, mode, options, and accumulated notes.

Inputs: SUMMARY array.

Outputs/Side Effects: No side effects besides printing.

## confirm\_live\_run()

Purpose: Interactive confirmation before live run.

Inputs: Shows scan totals, planned files, and config preview.

Outputs/Side Effects: Allows user to cancel before any write operation.

# Ordering Rationale (Why Functions Are Called in This Way)

• parse\_args before anything else to determine mode (scan-only, dry-run, live) and flags that influence later behavior.  
• setup\_logging early to capture the entire run’s output, including early failures.  
• detect\_layout\_once before any filesystem operations so REFCNT\_SUBPATH and repo root resolution are correct everywhere.  
• setup\_mountpoint (live mode / some dry-run situations) before planning copy and config edits, because NEW\_LINE depends on a validated mount path and the free-space check needs a concrete target.  
• verify\_ready\_to\_stop and wait\_for\_service\_stop before copy to ensure filesystem quiescence and consistent refcnt trees.  
• copy\_all\_refcnt before apply\_changes, because changing config first could redirect writes and misalign the migration.  
• verify\_all\_refcnt (optional checksum) immediately after copy to validate data integrity while still in a controlled state.  
• make\_backup immediately before apply\_changes to guarantee rollback capability.  
• apply\_changes only after a successful copy/verify to avoid partial migrations.  
• start\_services last, with a loop checking system state/health, to ensure the system returns to a reliable operational mode.  
• print\_summary at the end to provide a durable record of what happened (and what didn’t).

# Error Handling and Safety Considerations

• set -euo pipefail: aborts on unset variables and failed commands in pipelines.  
• safe\_rsync: enforces -n during dry-run to prevent accidental writes.  
• Separate stdout/stderr logging: avoids contaminating command substitutions, preserving correctness.  
• Backups and diffs: create a point-in-time backup of /etc/oca/oca.cfg and show diffs for traceability.  
• Timeouts and polling: service stop/start loops include timeouts to prevent indefinite hangs.  
• Confirmation gates: live run requires explicit 'yes' confirmation.

# Invariants & Design Decisions

• REFCNT\_SUBPATH is `.ocarina\_hidden/refcnt` (Standard) or `refcnt` (Alternate) and is used everywhere to avoid hardcoding.  
• Bulk copy uses a --files-from list of files under refcnt; rsync recreates necessary directories. --dirs is retained to ensure directory entries are synced when needed.  
• Progress bar UX is handled by run\_with\_bar, kept isolated from dry-run enforcement to preserve behavior.  
• Config editing uses a replace-or-insert strategy for TGTSSDDIR and avoids duplicate PLATFORM\_DS\_REFCNTS\_ON\_SSD lines.