Rsnapshot Restoration Plan

Rebuild an independent, testable backup system for the Arch host while using the predecessor configuration as evidence, not as a template.

Decision: retain the predecessor's proven hard-link snapshot model and 7/4/12/4 retention, but replace stale paths, unsafe live database copying, implicit remote trust, and the suspend-after-backup side effect.

1. Design Position

Keep

- rsnapshot plus rsync hard-link rotations on a separate local disk.
- Daily, weekly, monthly, and yearly recovery points.
- Filesystem backup of configuration, web data, Maildir, and application state.
- A dedicated logical database export captured before the snapshot.

Change

- Use a new snapshot root; preserve the predecessor archive untouched.
- Use systemd timers, mount guards, locking, capacity checks, and failure units.
- Replace the legacy MySQL dump with a consistent MariaDB 12 export.
- Require explicit approval before remote pulls, Tide data, or power actions.

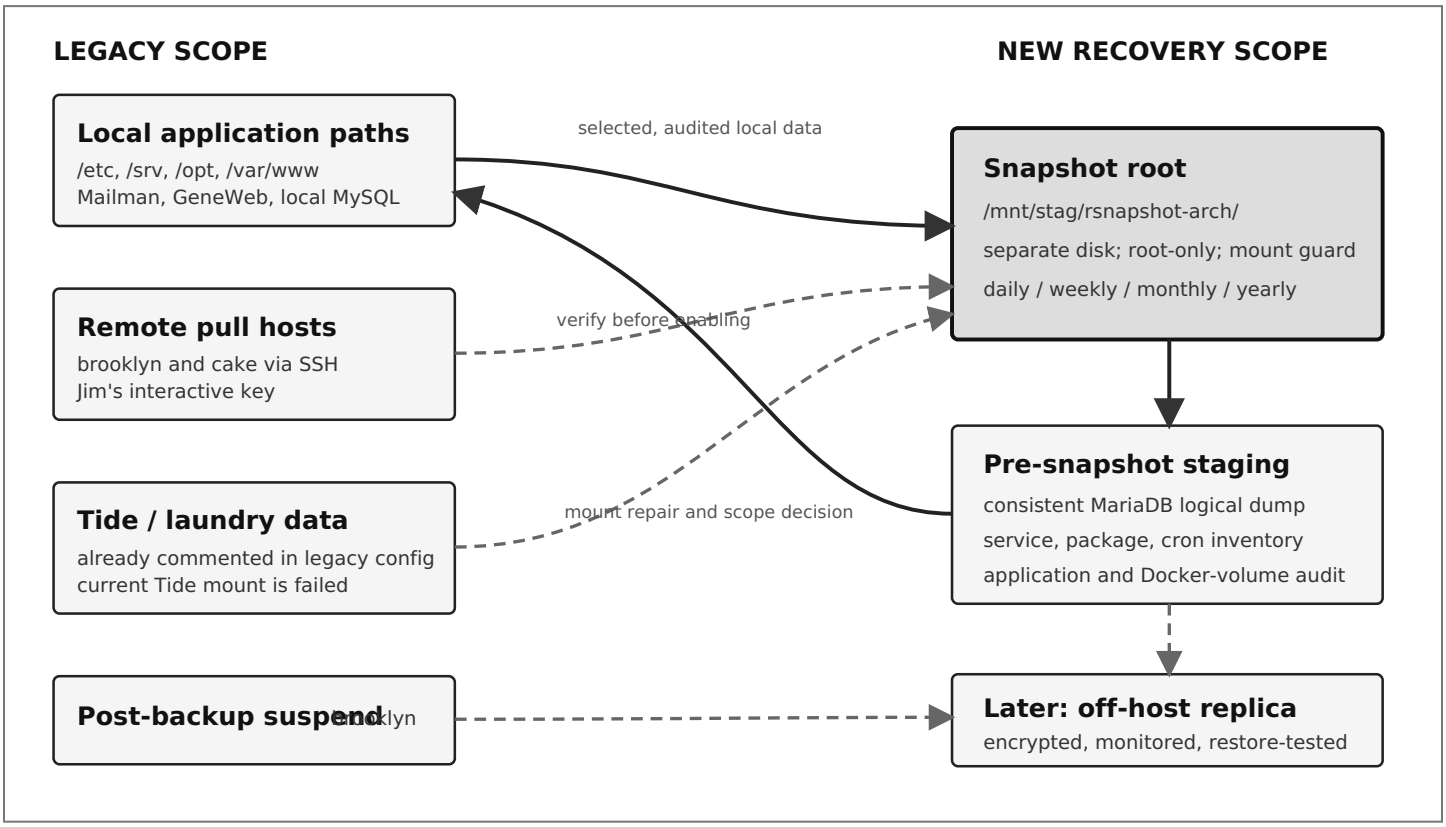
Not a substitute for Snapper: Snapper provides quick local Btrfs rollback. Rsnapshot provides independently stored, versioned file recovery on the separate `/mnt/stag` disk. A later off-host replica remains required for disaster recovery.

2. Evidence From The Legacy System

Legacy fact	Observed configuration	Modern interpretation
Snapshot root	`/mnt/stag/rsnapshot/`	Do not reuse it: it contains the retained predecessor history. Create `/mnt/stag/rsnapshot-arch/`.
Retention	7 daily, 4 weekly, 12 monthly, 4 yearly	**KEEP** Start with the same recovery horizon, then revisit after real capacity measurements.
Schedule	The packaged `/etc/cron.d/rsnapshot` sample existed but every run line was commented.	**CHANGE** Use explicit systemd timers with `Persistent=true`; Cronie remains available for user cron jobs.
Database hook	`mysqldump --all-databases`	**CHANGE** Use a restricted backup account and `mariadb-dump --single-transaction --routines --events`.
Post hook	SSH command suspended `brooklyn` after each snapshot.	**DO NOT PORT** A backup must never power-manage another host as an implicit side effect.
Remote pulls	Data was pulled from `brooklyn` and `cake` using Jim's SSH key.	**DEFER** Audit ownership, capacity, host reachability, and use a purpose-scoped backup key first.

3. Recovery Map

The diagram separates data already local to this host from optional remote/legacy scopes. Solid lines are planned for the first usable backup. Dashed lines require a separate approval.



4. First Backup Scope

Scope	Initial treatment	Reason
Configuration	**INCLUDE** `/etc`, `/root`, `/usr/local`, selected `/opt`	Required to rebuild services, cron, Apache, Keycloak, Postfix, and backup automation.
Content and mail	**INCLUDE** `/var/www`, `/srv`, `/var/mail`, application upload/data paths after audit	Contains restored web assets, static content, and Maildir.
MariaDB	**DUMP** into root-only staging, then snapshot the dump	Never use rsnapshot to copy live InnoDB data files as the database backup.
Docker	**AUDIT** bind mounts and named volumes; exclude overlay layers and images	Container images are reproducible. Persistent data is not.
Runtime/ cache	**EXCLUDE** `/proc`, `/sys`, `/dev`, `/run`, `/tmp`, caches, Snapper snapshots, destination tree	Prevents recursion, noise, unsafe pseudo-files, and unnecessary capacity use.
`/mnt/sternum`	**SEPARATE MONTHLY JOB**	It is a large, mostly static predecessor archive. Its first copy may consume about 600 GiB of the 1 TiB currently free on `/mnt/stag`.

5. Execution Phases

PHASE 0 Inventory active application data, Docker volumes, MariaDB size, source sizes, and the exact mount identity for `/mnt/stag`. Establish a free-space threshold before copying anything.

PHASE 1 Install rsnapshot and create `/mnt/stag/rsnapshot-arch/` as root-owned, mode 0700. Add a preflight that rejects an unmounted or low-space target rather than writing into an accidental local directory.

PHASE 2 Create a dedicated MariaDB backup credential and pre-snapshot service. Produce a consistent logical export with routines and events, validate its completion, and keep it readable only by root.

PHASE 3 Build the rsnapshot configuration from the audited local scope. Set `one_fs=1`; add mounted data deliberately rather than allowing traversal across all filesystems. Preserve relevant legacy exclusions and add current container/runtime exclusions.

PHASE 4 Add systemd `rsnapshot@daily`, weekly, monthly, and yearly services/timers. Use `Persistent=true`, `flock`, ordering after the database export, and independent log/failure reporting. Initial schedule: daily after Jim's 03:28 Maildir sync.

PHASE 5 Run `rsnapshot configtest`, a dry run, then the first daily snapshot. Measure actual storage before enabling the longer retention rotations.

PHASE 6 Perform a documented restore drill: recover a web file, Maildir message, configuration file, and MariaDB dump into an isolated validation target. Only then enable the full timer set.

6. Timer And Retention Policy

Interval	Retention	Planned cadence	Guardrails
daily	7	04:30 local time	After Jim's 03:28 Maildir sync; database export must succeed first.
weekly	4	Sunday 05:15	Never overlap with daily; same mount and capacity preflight.
monthly	12	Day 1, 06:00	Evaluate destination growth; include the separate legacy-archive run only after capacity approval.
yearly	4	January 1, 07:00	Restore sample before relying on the oldest tier.

7. Controls That Make This A Backup

1. **Independent storage:** the target is on `/mnt/stag`, not the root Btrfs disk.
2. **Consistency:** database exports are completed and checked before file snapshotting begins.
3. **Failure visibility:** timers log to journald and fail explicitly. Mail alerts remain secondary until outbound mail reputation is repaired.
4. **Capacity discipline:** no automatic first copy of the large predecessor archive; alert before the target approaches its approved threshold.
5. **Least privilege:** root-owned configuration, root-only database dump credentials, and dedicated noninteractive keys for any later remote source.
6. **Recovery proof:** a scheduled restore drill is a completion requirement, not a future aspiration.

8. Explicit Deferrals

Item	State	Condition to enable
Brooklyn and Cake remote pulls	DEFERRED	Confirm ownership, reachability, source paths, destination capacity, a dedicated key, and a restore requirement.
Tide/laundry sources	DEFERRED	Repair the failed Tide mount and decide which data is authoritative rather than duplicated.
Legacy `/mnt/sternum` archive	DEFERRED	Measure its baseline, reserve capacity, make one verified copy, then schedule monthly updates.
Post-snapshot Brooklyn suspend	RETIRED	Do not restore. Power state is outside backup completion semantics.
Off-host backup	NEXT DESIGN	Select encrypted external or remote storage after the local restore drill succeeds.

Prepared from the legacy `/mnt/debian/etc/rsnapshot.conf`, its database and post-snapshot hooks, current mounted-storage observations, and the restored service inventory. This document is a controlled implementation plan, not evidence that rsnapshot has been installed or scheduled.