Managing Version-Skew on a Shared /home

Scope: How to keep your data safe when multiple roots (openSUSE Tumbleweed/Aeon + test distros) mount the same Btrfs `/home`.

---

1 Why Skew Happens

- Rolling distributions ship the *same* apps days apart.
- A newer build may upgrade config files in `~/.config`, breaking an older build.
- Older kernels might not understand Btrfs features enabled by newer kernels.

---

2 Tumbleweed ⇄ Aeon Synchronisation

Pipeline Stage	**Tumbleweed**	**Aeon Desktop**
Build & QA	Daily snapshot after openQA pass	Image built **immediately** from the same snapshot
Delivery	`zypper dup` when you choose	`transactional-update.timer` auto-stages in 24 h
Typical Lag	0 days (if you run `dup`)	**0–2 days**

Shared /home – Risk Assessment

Risk Area	Why Skew Matters	Reality with TW + Aeon
Desktop configs	Newer version rewrites INI/JSON	Same snapshot ± 48 h → safe
Flatpak data	Uses per-user runtimes	Usually identical; Aeon may even be newer
Tool-chain caches	Forward compatible	No issues
Kernel-space user trees	Rare now	Not an issue

Bottom line: keep both roots updated weekly and you almost never see config breakage.

Keeping Them in Lock-Step

Strategy	How	Result
Let Aeon lead	Leave `transactional-update.timer` enabled; add a nightly `zypper dup` timer on TW	<24 h skew
Let TW lead	Disable timer on Aeon; run `transactional-update dup` after each `zypper dup`	Manual but synchronous

---

3 When Skew *Can* Bite

- Major desktop jumps (e.g. GNOME 46→47) — if you freeze one root for weeks.
- Opt-in RPMs present only on one root.
- Work-arounds: update, use per-root users, or isolate configs.

---

4 General Skew Management for *Other* Test Distros

4.1 Risk Map

Risk Zone	What Could Go Wrong	Impact
Files-on-disk	Kernel lacks new Btrfs flag	Disk fails to mount / data corruption
Configs	Newer app rewrites settings	Old build crashes or misbehaves
UID/GID	Installer picks UID 1001	Mixed file ownership
Accidental deletion	Unfamiliar UI	Data loss

4.2 Hardening Strategy

Pre-Flight Safeguards

Action	Command
Create read-only snapshot	`snapper -c home create -d "pre-test"`
Push snapshot to backup disk/ cloud	`btrbk send-receive` or `restic backup`
Freeze Btrfs feature set (optional)	`btrfs property set /home set-feature-compat-non-free no`

Layout Tricks (pick one)

Trick	Benefit
Per-distro sub-volumes	Dot-files isolated, bulk data shared
Per-distro users	Simpler; swap with symlinks to shared data
Consistent UID	Prevents mixed ownership
Containerised apps	Flatter config differences

Per-Boot Discipline

Habit	Why
Mount `/home` **ro** first boot	Confirm kernel & flags
Snapshot before each upgrade	Quick rollback
Aggressive snapshot pruning	Prevent NVMe fill-up

---

5 Restoration Playbook

Scenario	Steps
Config broken (files intact)	`snapper diff N..N-1` → `snapper undochange N`
Whole test distro trashed `/home`	Boot good root → `snapper rollback` latest healthy snapshot
Btrfs metadata corrupted	Restore send-stream from backup drive/cloud

---

6 TL;DR

1. **TW + Aeon** ship within 0–2 days → safe for single `/home` if both updated weekly.
2. For *other* test distros: snapshot before you boot, isolate configs (sub-volume or user), prune snapshots.
3. Off-disk backups (Btrfs send or Restic) guarantee a last-resort restore path.

With these safeguards you can multi-boot experimental roots on a single laptop without risking long-lived data—or your evening. 🚀