THE ZERO PARADOX

ZP Addendum

The Choice-Free Core

Version 1.0 | June 2026

Build the file ZeroParadox/AxiomProfile.lean and read the Lean kernel's output. It reports that the central theorem of this framework — the Binary Snap, T-SNAP, the forced transition $\bot \to \varepsilon_0$ — depends on no axioms at all: not the Axiom of Choice, not even propositional extensionality. The lattice algebra (ZP-A) and the Quine-atom self-reference that is the framework's keystone (ZP-J) are likewise choice-free. This is a machine-checked fact, not a claim of the prose: anyone can run `lake build ZeroParadox.AxiomProfile` and see it.

Two boundaries are stated up front, because the claim is narrow and exact. The framework as a whole is not choice-free. Most of its theorems do depend on `Classical.choice`. But every place it appears is a place where the framework builds on Mathlib's classically-built analysis, order, and computability libraries — the layers that realize the snap inside standard analytic structures (p-adic topology, Hilbert space, ordinals, category theory), where the dependence is inherited from those libraries. It is not used by the core results above. And dependence is not necessity: that those realizations use choice as written does not show choice is required there (Section III).

Section I: The Choice-Free Core

The Lean kernel's `#print axioms` command reports the complete axiom dependency of any result. Run on the framework's central results, it returns the following. "Does not depend on any axioms" is the strongest possible report — stronger than "choice-free," since it uses not even propositional extensionality.

Verified axiom-free (does not depend on any axioms)
T-SNAP, the Binary Snap and its derivation (ZP-E):
t_snap_machine, t_snap_derived, t_snap_join, t_snap_irreversible,
da1_minimal_path, dp2_execution_distinguishability.
The lattice algebra (ZP-A): bot_le, the order laws, cc1.
The Quine-atom self-reference keystone (ZP-J): bot_is_quine_atom, cc1_derived,
t_exec, quine_atom_unique.

A second tier of results is choice-free but uses propositional extensionality and quotient soundness (`[propext, Quot.sound]`), both standard in Lean 4. These include the structural floor (ZPH_PowerSet.ps_structural_floor) and the wheel of fractions (WheelFrac.instWheel, inf_ne_bot). No `Classical.choice`.

Section II: Where Classical.choice Enters

The honest contrast. `Classical.choice` does appear across the framework — in the majority of its theorems — and the same `#print axioms` artifact shows exactly where. Every occurrence is in a layer that realizes the snap floor inside a standard analytic structure, and inherits choice from the Mathlib library that builds that structure classically.

Carries Classical.choice (inherited from Mathlib), e.g.
ZP-B c3_irreversible — p-adic topology (metric / ultrametric library).
ZP-D t4_snap_orthogonal — Hilbert space (inner-product library).
ZP-H fB_functor / fD_functor / fC_functor — TopCat / ModuleCat ℂ /
the Kleisli category of the probability monad.
and the ordinal (ZP-L/M), information (ZP-C), and computability (ZP-K) layers.
Footprint in each case: [propext, Classical.choice, Quot.sound].

The pattern is clean: the core states the result; the analytic layers realize it inside the standard frameworks, and that is where the library's classical foundations enter. The choice is in the plumbing, not in the claim.

Section III: Dependence Is Not Necessity

`#print axioms` proves dependence — that the proof as written uses an axiom. It does not prove necessity — that no choice-free proof exists. The framework has no proven-necessity case anywhere: it has never shown, by a reversal, that any result requires choice. So the analytic-layer dependence may be removable.

One layer has been classified directly. In the "choice-probe" experiment, the `Classical.choice` in the 2-adic tree construction (ZPB_PadicTree) decomposed into three sources: incidental tactic artifacts (removed, leaving those results choice-free); Mathlib's classically-proved connectivity API (routable by a path-uniqueness reformulation); and `sInf` on a complete lattice (routable by a redefinition). The verdict for that layer was "mostly not structurally required." Whether this generalizes — and whether the snap geometry forces choice anywhere — is an open question, tracked for the constructive validation layer (ONote/NONote, future ZP-N).

Remark — Why this matters

The Zero Paradox argues that the foundational axioms are not freely chosen but forced by the structure of the bottom element. It would be a tension if the framework's own central results leaned on the Axiom of Choice — the canonical free, non-constructive selection. They do not. T-SNAP is axiom-free; the keystone is choice-free. The "forced, not chosen" thesis is internally consistent at the level of what the framework actually asserts. Where choice appears, it is the supporting library's classical foundation showing through the realizations, not an assumption of the argument.

The Artifact

Checkable evidence

ZeroParadox/AxiomProfile.lean — a file of `#print axioms` commands. Section I prints the choice-free core; Section II prints the analytic-realization results that carry choice, as an honest contrast. Build with `lake build ZeroParadox.AxiomProfile` and read the kernel's report. The per-theorem map is in the project repository.

Choice-probe (the one-layer classification): branch `choice-probe`; the verdict and the three decomposed sources are recorded in the project notes.

Scope of the claim

The claim is exactly: the framework's central results — T-SNAP, the lattice, the Quine-atom self-reference — are choice-free, and T-SNAP is axiom-free. NOT claimed: that the whole framework is choice-free (it is not), nor that the analytic-layer choice is removable or necessary (open). The fact surfaced here is the verified one.

Endnote: This is a framework-wide note, machine-verified as of June 2026. The central theorem T-SNAP depends on no axioms; the conceptual core is free of the Axiom of Choice; `Classical.choice` appears only where the framework builds on Mathlib's classical analysis, order, and computability libraries, and whether it is necessary there remains open. All of this is checkable in ZeroParadox/AxiomProfile.lean.