

v3.4-ZKP-001 – ZK-Primitives, Circuit Libraries & Privacy Anchors

Document Title	ZK-Primitives, Circuit Libraries & Privacy Anchors
Version	v3.4
Document ID	v3.4-ZKP-001
Date	2025-03-22
Author	Take Back Your Data – ZK Sovereignty Lab
Document Type	Public / Certification / Internal

1. Purpose & Scope

This document defines the foundational primitives, circuit logic and privacy anchors that power Zero-Knowledge (ZK) functionality in MaxOneOpen. It ensures verifiable, sovereign-proof logic without leaking private data or requiring central verification.

2. ZK Primitive Logic

- All forks must implement core ZK primitives (commitment, range proof, membership, set equality)
- Primitives must be composable, field-tested, and circuit-compatible
- ZK usage must be deterministic, reproducible and schema-bound
- Forks may not rely on opaque cryptographic shortcuts or unverifiable optimizations

3. Circuit Library Design

Circuit Module	Function Scope	Validation Constraint
Schema Verifier	Type match + input sealing	Hash + field zero check
Capability Prover	Policy trace + token proof	Set equality circuit
Timestamp Masker	Time window blind logic	Range check circuit
Identity Anchor	Decoupled ZK-PKI proof	Signature + root match

4. Certification Hooks

- Forks must use circuit libraries from auditable, open, and peer-validated repositories
- All ZK circuits must be schema-bound and version-traceable
- Private input must never leak via metadata, fallback logic or inference

5. Certification Triggers

- Non-auditable ZK primitives or opaque circuit logic disqualify fork
- Use of shortcut validation, fixed key schemes or privacy leakage invalidates certification

6. Certification Relevance

All MaxOneOpen forks must include Zero-Knowledge verification as core trust mechanism. Only forks with auditable, reproducible and privacy-preserving ZK logic qualify for certification.