ROOT ZERO VAULT

# RSBIS Master Note

*Root Zero–Aligned Complete Technical Specification (Master-Note)*

Hosameldeen Saleh

## Executive Summary

This document provides a single, organized, normative Root Zero–aligned specification for RSBIS as a unified system. It integrates the math/tooling layer (WIPI) and the governance layer (Structural Trust) into one implementable reference, and it includes a fully instantiated Continuity Bundle example with real computed CVIDs, hashes, and signatures.

Non-negotiable separation: identifiers and coordinates are indexes; authority and ancestry are resolved exclusively from vault paths and deed chains. Numeric substrings in display identifiers are opaque labels and MUST NOT be interpreted as governance hierarchy.

### Normative language

This specification uses RFC 2119 terms MUST, SHOULD, and MAY as normative requirements.

## 1. Scope and Enforcement Boundary

RSBIS is a governance substrate that renders high-risk decisions decidable and independently verifiable within a governed domain $\mathbb{D}$. All strong enforcement guarantees are conditional on the Governed Domain Assumption (GDA): any action executed inside $\mathbb{D}$ MUST be routed through mediation.

## 2. System Objects and Invariants

RSBIS expresses governance as a finite sequence of canonical artifacts. Every meaningful state transition is a signed artifact that is canonicalized, content-addressed (CVID), coordinate-indexed, journal-anchored, and replayable offline using a Continuity Bundle.

| Object | Role | Core Invariant |
|---|---|---|
| Artifact | Human-readable decision/state record. | Canonicalizable; signed per policy. |

| | | |
|---|---|---|
| Canonical Bytes | Deterministic byte form. | Same semantics ⇒ same bytes. |
| CVID | Content identifier for canonical bytes. | Recomputable offline. |
| Coordinate Tuple | Numeric indices for ordering/proofs. | Deterministic; reversible under declared rules. |
| Vault | Scope container. | Scope truth from vault path + deed chain. |
| Deed | Authority declaration + lifecycle law. | Signed; versioned; upgrades are governed events. |
| Mediator | Terminating validator. | Outputs only ACCEPT/REJECT + reason code. |
| Journal | Append-only hash chain. | Tamper-evident historical induction. |
| Continuity Bundle | Portable offline proof packet. | Sufficient for replay without network/db. |

# 3. Canonicalization and CVIDs

Canonicalization is governance law. A canonicalization profile MUST specify encoding (UTF-8), Unicode normalization (NFC), structural rules, and map ordering. CVIDs MUST be computed from canonical bytes using the active hash algorithm and recorded in journals and bundles.

# 4. Multi-Coordinate Identity (Global/Local/WIPI)

RSBIS supports multiple coordinates per identifier:
• Global coordinate $G_1$ (mandatory): Unicode-derived canonical ordering.
• Local coordinate $G_2$ (optional): tenant-declared alphabet mapping.
• WIPI coordinate W (optional): fixed-width coordinate algebra (v,w) where width is semantic.

Coordinates MUST NOT be used to infer authority.

## 4.1 Root Zero conversion rules (injective, reversible)

To ensure injectivity for variable-length strings, Root Zero uses a zero-free digit encoding:
• Map each digit x to x+1 (digits are in 1..B, never 0).
• Use base (B+1).

Global $G_1$: normalize ID to NFC. Let B=0x110000. For each code point $cp_i$ set $d_i=cp_i+1$ and base b=B+1. Compute $G_1(ID)=\Sigma\, d_i \cdot b^{(n-1-i)}$. Decode by repeated div/mod by b and subtract 1.

Local $G_2$: require published Σ and bijection $\mu:\Sigma\rightarrow\{0,\ldots,k-1\}$; digits $\mu(s_i)+1$; base (k+1); same formula.

## 4.2 WIPI integration (Profile A)

When fixed-width semantics are required, WIPI coordinates are represented as (v,w) under a declared base b and digit mapping. WIPI is base-agnostic and can accommodate Unicode or tenant alphabets when a fixed-width encoding is declared by profile. All WIPI usage MUST declare base and mapping (directly or by CVID reference) to preserve reversibility.

# 5. Vaults, Deeds, and Authority Resolution

Authority and ancestry are resolved from vault paths and deed chains. Numeric tokens in display identifiers are opaque labels. A deed MUST declare scope, key requirements, lifecycle stages, conversion rules in force, and upgrade/override policies.

# 6. Mediation and Deterministic Validation

The mediator MUST terminate (non-Turing): finite predicates with acyclic dependencies. Given an event artifact, the mediator MUST:
1) parse; 2) canonicalize; 3) compute CVID; 4) compute required coordinates; 5) resolve authority; 6) evaluate predicates; 7) emit outcome and reason code; 8) journal-anchor.

# 7. Journals, Registries, and Tamper-Evidence

Journals are append-only and hash-chained. Every mediated decision yields a journal entry anchoring event CVID, outcome, reason code, signatures, and chain linkage. Registries are optional mirrors and MUST NOT affect validity.

# 8. Continuity Bundles (Normative Schema and Invariants)

A Continuity Bundle is a finite artifact set sufficient to recompute ACCEPT/REJECT and reason code offline. Bundles MUST be self-contained: no external databases, vendor services, or network calls are permitted for verification.

## 8.1 Schema

| Field | Type | Req. | Purpose / Constraint |
|---|---|---|---|
| bundle_version | string | MUST | Bundle schema identifier (e.g., RZ.CONTINUITY.1). |
| profile_id | string | MUST | Rule profile in force at decision time. |
| profiles | object | MUST | CVIDs (and optionally embedded texts) for canonicalization, sort spec, conversion rules, validator constraints. |
| event_artifact | text/bytes | MUST | Decision artifact. |
| event_canon_bytes | bytes | MUST | Canonical bytes of event_artifact under the stated profile. |
| event_cvid | string | MUST | CVID(event_canon_bytes). |
| vault_path | string | MUST | Resolved legal scope path. |

| | | | |
|---|---|---|---|
| deed_chain | list<object> | MUST | Ordered deeds sufficient to resolve authority; each includes CVID and signatures. |
| journal_context | object | MUST | At minimum: prev_entry_hash and entry_hash; include any turn/order state required. |
| outcome | enum | MUST | Recorded ACCEPT/REJECT. |
| reason_code | string | MUST | Recorded reason code (OK or E-*). |
| public_keys | list<object> | MUST | Verification keys required by policy. |
| signatures | list<object> | MUST | Signatures required by policy. |
| coordinate_proofs | object | MAY | Optional recomputation traces for $G_1/G_2/W$; if present MUST verify. |
| overrides | object | MAY | If used, MUST include override authorization and proof override is allowed by profile. |
| implementation_fingerprint | object | SHOULD | Optional build/runtime fingerprint for dispute resolution; not required if profiles suffice. |

## 8.2 Verifier invariants

Invariant I1 (CVID determinism): event_cvid MUST equal CVID(event_canon_bytes).
Invariant I2 (Profile determinism): profiles MUST identify the exact canonicalization/sort/conversion

rules used.
Invariant I3 (Authority determinism): deed_chain + vault_path MUST be sufficient to resolve authority without external state.
Invariant I4 (Outcome determinism): offline recomputation MUST match outcome and reason_code.
Invariant I5 (No hidden dependencies): verification MUST not require network access or mutable databases.

### 8.3 Bundle profiles

Minimal Bundle: required fields only; profile texts may be referenced by CVID if already archived.
Forensic Bundle: embeds full rule texts, full deed artifacts, optional coordinate proofs, and dispute metadata for courtroom-grade replay.

## 9. Positioning and Related Systems

RSBIS composes known primitives (canonicalization, content addressing, hash chaining, signatures) into a terminating governance stack with offline replay. The novelty claim is compositional and contractual: a mediator contract (ACCEPT/REJECT + reason codes) plus a continuity bundle contract for long-horizon recomputation.

### 9.1 Transparency logs (Certificate Transparency)

CT provides append-only public logs and inclusion/consistency proofs for certificates. RSBIS generalizes the ‘public verifiability’ principle to arbitrary governance decisions and adds a terminating mediator contract and continuity bundle replay.

### 9.2 Blockchains and consensus ledgers

Blockchains typically encode validity as a function of global shared state and consensus. RSBIS does not require consensus to decide validity: validity is deterministically computed from published profiles and bundle contents within a governed domain.

### 9.3 Git signed histories and supply-chain frameworks

Git and related frameworks enable content addressing and signed histories, but governance outcomes often remain procedural. RSBIS specifies a terminating mediator with canonical reason codes and portable replay bundles, and can treat Git objects as governed artifacts.

# 10. Fully Instantiated Example (No Placeholders)

This example provides concrete computed values to demonstrate reproducibility. Values below were computed from the canonical bytes shown.

## 10.1 Event artifact (YAML)

event_type: "DEPLOY_APPROVAL"
model_id: "LOVE"
stage: "DEPLOY"
timestamp: "2025-12-29T12:34:56Z"
turn: 42
vault_path: "/RootZero/Deployments/PROD"

## 10.2 Canonical bytes and CVID

Canonicalization: UTF-8 + NFC, keys sorted lexicographically, LF line endings.

event_canon_bytes (base64):
ZXZlbnRfdHlwZTogIkRFUExPWV9BUFBST1ZBTCIKbW9kZWxfaWQ6ICJMT1ZFIgpzdGFnZTogIkRFUExPWSIKdGltZXN0YW1wOiAiMjAyNS0xMi0yOVQxMjozNDo1NloiCnR1cm46IDQyCnZhdWx0X3BhdGg6ICIvUm9vdFplcm8vRGVwbG95bWVudHMvUFJPRCIK

event_cvid: cvid:sha256:ed8cb2db83b285f39274fc8308ebc6038a8cfc5eae5f0573ccb702e63392bb4f

## 10.3 Signature material (ed25519)

public_key_raw (base64): BtR6zq7bzkdFaGZ1Km1PpLdthpK4epRbaAIZXKEAiHI=

signature(event_cvid) (base64):
9exre9cenTV9RUycv/gi8LH+l8LMEeem0wS6j7rf7SrTbSRpXXh5I+2xD7xvLfzkjHhNh+xh1jD8z1O/BuX5DQ==

## 10.4 Journal entry and hash chaining

journal_entry_canonical_json:
{"event_cvid":"cvid:sha256:ed8cb2db83b285f39274fc8308ebc6038a8cfc5eae5f0573ccb702e63392bb4f",
"outcome":"ACCEPT","prev_entry_hash":"0000000000000000000000000000000000000000000000000000

000000000000000","reason_code":"OK","sig_alg":"ed25519","signature_b64":"9exre9cenTV9RUycv/gi8LH+l8LMEeem0wS6j7rf7SrTbSRpXXh5I+2xD7xvLfzkjHhNh+xh1jD8z1O/BuX5DQ==="}

prev_entry_hash: 0000000000000000000000000000000000000000000000000000000000000000

entry_hash: 067a200babdcb790318822bb3785d891a70becf70744078b03f44efec7f5d2f3

## 10.5 Minimal continuity bundle (illustrative JSON)

{
 "bundle_version": "RZ.CONTINUITY.1",
 "deed_chain": [
  {
   "deed_cvid":
"cvid:sha256:61c0feeb56fe1d252d722affb9a0554f8e134afcb06d1b654e4cea12d7df6e97",
   "deed_text": "deed_version: 1\nscope: /RootZero\nsignature_policy: ed25519-only\n",
   "scope": "/RootZero"
  }
 ],
 "event_artifact": "event_type: \"DEPLOY_APPROVAL\"\nmodel_id: \"LOVE\"\nstage:
\"DEPLOY\"\ntimestamp: \"2025-12-29T12:34:56Z\"\nturn: 42\nvault_path:
\"/RootZero/Deployments/PROD\"\n",
 "event_canon_bytes_b64":
"ZXZlbnRfdHlwZTogIkRFUExPWV9BUFBST1ZBTCIKbW9kZWxfaWQ6ICJMT1ZFIgpzdGFnZTogIkRFUExPWSIKdGltZXN0YW1wOiAiMjAyNS0xMi0yOVQxMjozNDo1NloiCnR1cm46IDQyCnZhdWx0X3BhdGg6ICIvUm9vdFplcm8vRGVwbG95bWVudHMvUFJPRCIK",
 "event_cvid":
"cvid:sha256:ed8cb2db83b285f39274fc8308ebc6038a8cfc5eae5f0573ccb702e63392bb4f",
 "journal_context": {
  "entry_hash": "067a200babdcb790318822bb3785d891a70becf70744078b03f44efec7f5d2f3",
  "prev_entry_hash": "0000000000000000000000000000000000000000000000000000000000000000"
 },
 "outcome": "ACCEPT",
 "profile_id": "ROOTZERO.PROFILE.1",
 "profiles": {
  "canonicalization_profile": "ROOTZERO.CANON.YAML.1",
  "conversion_rules": "ROOTZERO.CONV.MULTICOORD.1",
  "hash_alg": "sha256"
 },

```
 "public_keys": [
   {
    "alg": "ed25519",
    "public_key_raw_b64": "BtR6zq7bzkdFaGZ1Km1PpLdthpK4epRbaAIZXKEAiHI="
   }
 ],
 "reason_code": "OK",
 "signatures": [
   {
    "alg": "ed25519",
    "signature_b64":
"9exre9cenTV9RUycv/gi8LH+l8LMEeem0wS6j7rf7SrTbSRpXXh5I+2xD7xvLfzkjHhNh+xh1jD8z1O/B
uX5DQ==",
    "signed": "event_cvid"
   }
 ],
 "vault_path": "/RootZero/Deployments/PROD"
}
```


# RSBIS — Recursive Stage-Based Identifier System

Master Note — Root Zero–Aligned Complete Technical Specification (10/10 Paste-Ready)

Core Thesis

RSBIS replaces fragile operational trust (servers, pointers, mutable logs, discretionary authorities) with structural trust: canonical bytes, deterministic conversion mathematics, cryptographic evidence chains, and offline recomputability.

Root Zero Alignment (Non-Negotiable): In the Root Zero profile, numeric blocks are opaque labels. Authority, ancestry, and governance scope are never inferred from numeric digits. They are resolved only via Vault YAML path resolution + Deed chain. Numeric coordinates exist for ordering, indexing, and invariants—not for implicit hierarchy.

Abstract

Most identity systems trade off between (i) global uniqueness and (ii) local governance semantics. In high-stakes domains (identity issuance, financial authorization, infrastructure control, AI deployment), that trade fails at audit time: validity cannot be recomputed offline; policy outcomes are not deterministic; hidden overrides exist.

RSBIS defines identity and governance as a decidable property of a finite proof bundle: canonical bytes, conversion rules, cryptographic bindings (CVID), deterministic mediation, and tamper-evident recordkeeping. Root Zero provides the defensible “paper-grade” governance substrate: canonical YAML, CanonicalSortSpec ordering, explicit conversion rules (global Unicode + local subset base), deed/vault binding, dual signatures, and journal/registry double-entry accountability.

## PART I — FOUNDATIONS AND PROBLEM FORMULATION

### 1. The Fundamental Tension

Digital governance must satisfy both:

Global Consistency Requirement

globally unique and collision-resistant anchors

cryptographically verifiable

stable across systems and time

recomputable by auditors without vendor infrastructure

Local Sovereignty Requirement

local alphabets and conventions (Latin, Arabic, Unicode)

explicit lifecycle stages and governance rules

jurisdictional scoping via deeds/vaults

upgrades without breaking auditability

RSBIS Resolution (Root Zero Form):

Immutable mathematical anchors + declarative, human-readable governance—both canonical, both offline-recomputable.

2. Six Critical Problems Addressed

Problem 1 — Non-Canonical Representation

Unicode variance, ordering variance, formatting variance, and type ambiguity can produce byte-distinct artifacts for the same meaning.

RSBIS/Root Zero response: canonicalization + CanonicalSortSpec ensures semantically equivalent artifacts map to identical canonical bytes → identical CVIDs.

Problem 2 — Non-Deterministic Policy

Runtime-dependent engines can diverge, not terminate, or allow human discretion.

Response: non-Turing, terminating mediation; deterministic evaluation graph; explicit reason codes.

Problem 3 — Live-Infrastructure Dependency

Verification requiring databases/APIs collapses sovereignty.

Response: continuity bundles: all verification inputs are included; offline recomputation is mandatory.

Problem 4 — Mutable Audit Trails

Logs can be rewritten, selectively withheld, or forged.

Response: hash-chained journal with inclusion proofs as optional enhancement; detectable tampering by induction.

Problem 5 — Hidden Override (“Secret Zero”)

Unlogged master keys/backdoors make published rules advisory.

Response: recomputation invariant—any forced ACCEPT inconsistent with the Vault+Deed reality is detectable.

Problem 6 — Opaque Governance

Executable policy is not human-readable, stable, or court-defensible.

Response: static declarative vault logic (canonical YAML), content-verified by CVID; interpreted deterministically.

3. Formal Definitions (Normative Semantics)

Definition 3.1 — Decision (Conceptual Object)

```
decision:
  type: "governance_decision"
  action: "deploy_model"
  payload:
    model_hash: "hash:sha3-256:PLACEHOLDER"
    target_environment: "production"
  requester: "user:alice"
  justification: "PLACEHOLDER"
  signatures:
    policy: "dual"
    required_algorithms: ["ed25519", "pqc_dilithium3"]
    values:
      - "sig:ed25519:PLACEHOLDER"
      - "sig:pqc_dilithium3:PLACEHOLDER"
  ts: "2025-01-15T10:30:00Z"
```

Definition 3.2 — Structural Trust ($\mathbb{T}_s$)

$\mathbb{T}_s$ holds if there exists a finite bundle such that verification is deterministic, offline, terminating, and admits no hidden bypass in the governed domain.

```
structural_trust:
  requires:
```

- "finite_bundle"
- "offline_verification"
- "deterministic_verification"
- "terminating_mediation"
- "no_hidden_bypass_in_domain"

Definition 3.3 — Governed Domain Assumption (GDA)

If a high-risk action executes inside domain $\mathbb{D}$, it MUST have passed mediation under the domain's effective Vault logic.

gda:

statement: "High-risk actions in domain require mediated ACCEPT"

scope_note: "Institutional constraint; actions outside domain are out of scope"

Definition 3.4 — Canonical Representation

A deterministic, bijective mapping from semantic content to canonical bytes.

canonical_representation:

rule: "semantic_equivalence <-> byte_identity"

Definition 3.5 — CVID (Root Zero Profile)

Root Zero uses a fixed CVID convention:

cvid:

format: "cvid:blake3:<hex>"

definition: "CVID(content) = BLAKE3(CanonBytes(content))"

notes:

- "Algorithm agility occurs only via new profile versions"
- "Within a profile, the CVID algorithm is fixed"

PART II — CANONICALIZATION (ROOT ZERO NORMATIVE)

4. Canonical YAML Profile (Normative)

Root Zero artifacts are canonical YAML; canonicalization MUST reject ambiguous YAML features.

```
canonical_yaml_profile:
  profile_id: "ROOTZERO.CANON.YAML.1"
  encoding: "UTF-8"
  unicode_normalization: "NFC"
  line_endings: "LF"
  indentation_spaces: 2
  flow_style: "forbidden"
  anchors: "forbidden"
  aliases: "forbidden"
  tags: "forbidden"
  comments: "forbidden"
  duplicate_keys: "forbidden"
  scalars:
    strings: "double_quoted_or_plain_as_profile_allows"
    integers: "base10_only"
    floats: "forbidden"
    booleans: "lowercase"
    nulls: "lowercase"
```

5. CanonicalSortSpec (Normative Ordering)

CanonicalSortSpec defines deterministic ordering for serialization. For the Root Zero profile, key order is explicitly specified per artifact class, not left to implementation.

5.1 Journal Entry Canonical Key Order

```
canonical_sort_spec:
  artifact: "JournalEntry"
  key_order:
    - "ts"
    - "type"
    - "signer"
    - "signature_policy"
    - "signatures"
    - "payload"
    - "hash"
```

5.2 Registry Entry Canonical Key Order

```
canonical_sort_spec:
  artifact: "RegistryEntry"
  key_order:
    - "ts"
    - "type"
    - "signer"
    - "signature_policy"
    - "signatures"
    - "payload"
```

- "hash"

Rule: CanonBytes MUST serialize mappings in the relevant CanonicalSortSpec order for the artifact class; otherwise reject with E-CANON.

PART III — IDENTIFIER GRAMMAR (ROOT ZERO)

6. Display ID Formats (Normative)

Root Zero supports versioned display IDs. Digits are not ancestry. They are stable labels.

id_formats:

v1:

pattern: "LLLL-NNNN-NNNN"

regex: "^[A-Z]{4}-[0-9]{4}-[0-9]{4}$"

v2:

pattern: "LLLL-NNNN-NNNN-NNNN"

regex: "^[A-Z0-9]{4}-[0-9]{4}-[0-9]{4}-[0-9]{4}$"

rules:

- "Numeric blocks are opaque labels; do not infer hierarchy"

- "Any ancestry/scope must be resolved from Vault YAML path + Deed chain"

7. Coordinate Fields (Root Zero Semantics)

Root Zero distinguishes:

Codepoint / Unicode scalar data (string-based global determinism)

Coordinate (integer conversion result)

Class (artifact class/type indicator)

field_taxonomy:

fields:

- "class"
- "codepoint"
- "coordinate"
- "cvid"
- "id"
- "issued_by"
- "vault_logic_ref"

PART IV — CONVERSION RULES (ROOT ZERO NORMATIVE)

Root Zero mandates explicit conversion rules. These rules are how strings become stable coordinates.

8. Conversion Rule G1 — Global Unicode Coordinate (Canonical Across All Scripts)

Purpose: a universal, deterministic coordinate for any Unicode string (after NFC), providing a canonical ordering independent of local alphabets.

Let:

B = 1114112 (0x110000; full Unicode scalar space)

For a normalized string s of length n with code points cp(i) in logical order:

$G_{global}(s) = \Sigma ( cp(i) \times B^i ) \text{ for } i = 0..n-1$

conversion_rules:

G1_global_unicode:

base_B: 1114112

input: "Unicode string (NFC normalized)"

formula: "G = Σ(code_point(symbol_i) * B^i)"

ordering: "logical order (not visual); includes all code points"

notes:

- "Uses Unicode scalar values, not UTF-16 surrogate pairs"

- "All characters count, including ZWJ/ZWNJ/ZWSP and bidi marks"

- "Combining sequences must be NFC-normalized before conversion"

9. Conversion Rule G2 — Local Subset Coordinate (Tenant-Declared Alphabet)

A tenant may declare a local alphabet and compute a base-k coordinate. This is convenience and efficiency, not global truth.

conversion_rules:

G2_local_subset:

requires:

- "tenant declares alphabet Σ and base k = |Σ|"

- "tenant declares symbol->value mapping"

formula: "G = Σ(value(symbol_i) * k^i)"

note: "Only valid under declared tenant policy; must be published in Vault"

Example (Illustrative): “LOVE” with Base-26 Alphabet

conversion_example_local_subset:

alphabet: "ABCDEFGHIJKLMNOPQRSTUVWXYZ"

base_k: 26

mapping: { A: 0, B: 1, C: 2, D: 3, E: 4, F: 5, G: 6, H: 7, I: 8, J: 9,

K: 10, L: 11, M: 12, N: 13, O: 14, P: 15, Q: 16, R: 17, S: 18, T: 19,

U: 20, V: 21, W: 22, X: 23, Y: 24, Z: 25 }

string: "LOVE"

values: [11, 14, 21, 4]

coordinate: 221410

note: "Illustrative tenant-local coordinate; not global truth"

10. Conversion Rule Precedence (Critical Clarification)

coordinate_precedence:

authoritative_order:

1: "G_global_unicode (canonical across scripts)"

2: "Resolved Vault/YAML path scope + Deed chain (authority truth)"

3: "Any local subset coordinate (tenant convenience)"

verification_requirements:

- "All claimed coordinates must recompute from canonical inputs"

- "Mismatch -> E-CANON or E-CONV (policy-defined), never ACCEPT"

11. Unicode Edge Cases (Normative Handling)

unicode_edge_cases:

combining_characters:

rule: "Apply NFC before coordinate computation"

example: "U+00E9 equals U+0065 U+0301 after NFC"

surrogate_pairs:

rule: "Use scalar value, not UTF-16 surrogate pairs"

bidirectional_text:

rule: "Compute on logical order; visual rendering is irrelevant"

note: "Bidi marks are included if present"

zero_width_characters:

rule: "Included (they are code points)"

examples: ["U+200B", "U+200C", "U+200D"]

PART V — WIPI MATHEMATICS (PRESERVED, ROOT ZERO–SAFE)

RSBIS retains WIPI for graded positional semantics without violating Root Zero.

12. WIPI: Width-Indexed Positional Identifiers (Optional Coordinate Algebra)

WIPI treats width as semantic. Leading zeros are meaningful within a declared WIPI coordinate context.

wipi:

base_b: 10

element: "(v,w)"

constraints:

- "w >= 1"

- "0 <= v < b^w"

note:

- "WIPI may be used for capacity/turn semantics"

- "Root Zero forbids using numeric blocks to infer authority/ancestry"

Parent/Extension Operators (Mathematical, Not Governance Authority)

wipi_operators:

parent:

rule: "parent(v,w) = (floor(v/b), w-1)"

requires: "w >= 2"

extend:

rule: "ext_d(v,w) = (b*v + d, w+1)"

digit_domain: "0..b-1"

warning:

- "These operators do not define governance ancestry in Root Zero"

- "Governance ancestry is Vault YAML path + Deed chain only"

PART VI — THE FIVE STRUCTURAL COMPONENTS (ROOT ZERO FORM)

RSBIS Structural Trust requires five components:
$\mathscr{R}$ Root, $\mathscr{A}$ Authenticity, $\mathscr{L}$ Recordkeeping (Journal+Registry), $\mathscr{C}$ Continuity, $\mathscr{M}$ Mediation.

13. Root ($\mathscr{R}$) — Sovereign Origin via Vault Path + Deeds

A Root is the sovereign scope origin defined in the Vault's YAML structure, delegated by deeds.

root:

truth_source:

- "vault_yaml_path"

- "deed_chain"

forbidden:

- "numeric_hierarchy_inference_from_digits"

14. Authenticity ($\mathscr{A}$) — Canonical Bytes ↔ CVID

authenticity:

canonicalization: "ROOTZERO.CANON.YAML.1"

cvid: "cvid:blake3:<hex>"

property: "Canonical bytes fully determine CVID; recomputable offline"

15. Recordkeeping ($\mathscr{L}$) — Journal + Registry Double-Entry

Root Zero distinguishes:

Journal: local authoritative record (execution truth inside domain)

Registry: public/transparent mirror for reportable events (accountability truth)

recordkeeping:

journal:

role: "local_authoritative_record"

required: true

registry:

role: "public_transparency_record"

required: "policy-dependent"

rule:

- "Journal write is prerequisite for any registry mirror"

16. Continuity ($\mathcal{C}$) — Offline Decidability Bundle

continuity:

requires_bundle_contains:

- "canonical_profile_id"

- "canonical_sort_specs"

- "conversion_rules_used"

- "vault_chain (content + CVIDs)"

- "deed_chain (content + CVIDs)"

- "event artifact (canonical bytes reference)"

- "journal context (prior hash, current hash)"

- "registry context if applicable"

- "all required signatures + pubkeys"

must_not_require:

- "databases"

- "network"

- "vendor services"

17. Mediation ($\mathcal{M}$) — Deterministic, Terminating Enforcement

mediation:

outputs: ["ACCEPT", "REJECT"]

reject_requires: ["reason_code", "explanation"]

constraints:

- "deterministic"

- "terminating"

- "non_turing"

- "acyclic_dependency_graph"

PART VII — VAULT + DEED GOVERNANCE ARCHITECTURE (ROOT ZERO)

18. Vault Structure (Declarative Governance Container)

Vault logic is human-readable, canonical, content-verified, and hierarchically resolved by YAML path.

vault:

vault_version: "1"

canonical_profile_id: "ROOTZERO.CANON.YAML.1"

canonical_sort_spec_refs:

- "JournalEntry"

- "RegistryEntry"

```
scope:
  resolution: "yaml_path"
  note: "Scope truth comes from YAML location + deed binding"
conversion_rules:
  global_unicode: "G1_global_unicode"
  local_subset_optional:
    enabled: true
    alphabet_id: "TENANT_ALPHA_1"
    base_k: 26
    mapping_policy: "declared_in_vault"
stage_model:
  stages: ["draft","review","staging","production","retired"]
  transitions:
    draft: ["review"]
    review: ["staging","draft"]
    staging: ["production","review"]
    production: ["retired"]
    retired: []
turn_order:
  policy: "STRICT_MONOTONIC"
signature_policy:
  required: "dual"
  algorithms: ["ed25519","pqc_dilithium3"]
```

```
registry_policy:
  enabled: true
  reportable_actions:
    - "cross_jurisdiction_transfer"
    - "public_accountability_event"
    - "regulatory_reporting_required"
  conflict_handling:
    journal_accept_registry_reject: "REJECT_FOR_REPORTABLES"
    registry_unavailable: "DEFER_OR_REJECT (policy-defined)"
rules:
  - rule_id: "R.SIG.DUAL"
    predicate:
      type: "dual_signature_required"
      algorithms: ["ed25519","pqc_dilithium3"]
    reject_code: "E-SIG"
  - rule_id: "R.STAGE.TRANSITION"
    predicate:
      type: "stage_transition_allowed"
    reject_code: "E-STAGE"
  - rule_id: "R.TURN.ORDER"
    predicate:
      type: "turn_order_policy"
      policy: "STRICT_MONOTONIC"
```

reject_code: "E-TURN"

19. Vault Inheritance / Resolution (Deterministic)

Vaults resolve from root → leaf by YAML path; merges are deterministic and cycle-rejected.

vault_resolution:

chain_order: "root_to_leaf"

merge_rules:

scalars: "child_overrides"

mappings: "recursive_merge"

sequences: "replace_or_extend (explicitly defined per key)"

cycle_policy:

action: "reject"

reject_code: "E-CHAIN"

20. Deed Structure (Authority Binding to Vault Logic)

A deed binds scope to a specific vault logic reference and enforces signature policy. Deeds are frozen at issuance.

deed:

class: "DEED"

deed_version: "1"

deed_id: "cvid:blake3:PLACEHOLDER"

id: "ABCD-0007-0003-0012"

codepoint:

note: "Canonical string/codepoint representation used in conversion"

value: "ABCD-0007-0003-0012"

```
coordinate:
  global_unicode: "PLACEHOLDER_INTEGER"
  local_subset_optional: "PLACEHOLDER_INTEGER_OR_NULL"
vault_logic_ref:
  vault_cvid: "cvid:blake3:PLACEHOLDER"
issued_by:
  issuer_id: "AUTHORITY.ROOT"
  pubkeys:
    - "pk:ed25519:PLACEHOLDER"
    - "pk:pqc_dilithium3:PLACEHOLDER"
validity:
  valid_from: "2025-01-15T00:00:00Z"
  valid_until: "2026-01-15T00:00:00Z"
signature_policy:
  required: "dual"
  algorithms: ["ed25519","pqc_dilithium3"]
signatures:
  - "sig:ed25519:PLACEHOLDER"
  - "sig:pqc_dilithium3:PLACEHOLDER"
immutability:
  - "signature_policy_frozen"
  - "vault_logic_ref_frozen"
  - "turn_order_constraints_frozen (as applicable)"
```

21. Crypto Policy Evolution (No Retroactive Mutation)

Root Zero treats deed policy as immutable; upgrades happen by reissuance.

```
signature_policy_evolution:
  principle:
    - "Once issued, deed signature_policy is frozen"
  upgrade_mechanism:
    - "Issue new deed with new policy"
    - "Old deed marked superseded_by"
    - "New deed references prior deed for continuity"
  example:
    old_deed:
      signature_policy: "dual"
      algorithms: ["ed25519","pqc_dilithium3"]
      deed_id: "cvid:blake3:OLD"
    new_deed:
      signature_policy: "triple"
      algorithms: ["ed25519","pqc_dilithium3","falcon512"]
      supersedes: "cvid:blake3:OLD"
      deed_id: "cvid:blake3:NEW"
```

PART VIII — JOURNAL + REGISTRY SCHEMAS (ROOT ZERO NORMATIVE)

22. Journal Entry (Canonical YAML)

```
journal_entry:
```

```
ts: "2025-01-15T10:30:02Z"
type: "decision"
signer: "MEDIATOR.NODE.1"
signature_policy: "dual"
signatures:
  - "sig:ed25519:PLACEHOLDER"
  - "sig:pqc_dilithium3:PLACEHOLDER"
payload:
  decision_cvid: "cvid:blake3:PLACEHOLDER"
  id: "ABCD-0007-0003-0012"
  vault_cvid: "cvid:blake3:PLACEHOLDER"
  outcome: "ACCEPT"
  reason_code: "NONE"
  reason_text: "All predicates satisfied"
  prior_hash: "hash:blake3:PLACEHOLDER"
hash: "hash:blake3:PLACEHOLDER"
```

Canonical order MUST follow JournalEntry CanonicalSortSpec (ts,type,signer,signature_policy,signatures,payload,hash).

23. Registry Entry (Public Mirror)

Registry entries mirror reportable journal outcomes and include a journal reference.

```
registry_entry:
  ts: "2025-01-15T10:30:03Z"
  type: "registry_mirror"
```

```
  signer: "REGISTRY.NODE.1"
  signature_policy: "dual"
  signatures:
    - "sig:ed25519:PLACEHOLDER"
    - "sig:pqc_dilithium3:PLACEHOLDER"
  payload:
    journal_ref:
      journal_hash: "hash:blake3:PLACEHOLDER"
      journal_ts: "2025-01-15T10:30:02Z"
    decision_cvid: "cvid:blake3:PLACEHOLDER"
    id: "ABCD-0007-0003-0012"
    outcome: "ACCEPT"
    reportable_reason: "regulatory_reporting_required"
  hash: "hash:blake3:PLACEHOLDER"
```

24. Registry Triggering & Conflict Rules (Critical Clarification)

```
registry_requirements:
  reportable_actions:
    - "cross_jurisdiction_transfer"
    - "public_accountability_event"
    - "regulatory_reporting_required"
  triggering_logic:
    if: "decision.action in reportable_actions"
    then: "registry_mirror_required"
```

```
  else: "journal_only_sufficient"
conflict_handling:
  journal_accept_registry_reject: "REJECT_FOR_REPORTABLES"
  journal_reject_registry_accept: "impossible (journal prerequisite)"
  registry_unavailable: "defer_or_reject (policy-defined)"
```

## PART IX — VALIDATION & CONFORMANCE (ROOT ZERO)

### 25. Validator Flow (Normative, Conformance-Testable)

A Root Zero validator MUST implement a conformance pipeline that is deterministic and testable.

```
validator_flow:
 - step: "Parse artifact as YAML"
   reject_code: "E-PARSE"
 - step: "NFC normalize all strings under profile"
   reject_code: "E-CANON"
 - step: "Enforce ROOTZERO.CANON.YAML.1 restrictions"
   reject_code: "E-CANON"
 - step: "Enforce CanonicalSortSpec for artifact class"
   reject_code: "E-CANON"
 - step: "Validate display ID against versioned regex"
   reject_code: "E-ID"
 - step: "Recompute CVID (BLAKE3 over canonical bytes)"
   reject_code: "E-CVID"
 - step: "Resolve Vault chain by YAML path; reject cycles"
```

```
  reject_code: "E-CHAIN"
- step: "Resolve Deed chain; verify dual signatures"
  reject_code: "E-SIG"
- step: "Recompute conversions (G1 global, optional G2 local)"
  reject_code: "E-CONV"
- step: "Evaluate deterministic mediation rules"
  reject_code: "E-RULE"
- step: "Write Journal entry and verify hash linkage"
  reject_code: "E-JOURNAL"
- step: "If reportable, mirror to Registry per policy"
  reject_code: "E-REG"
- step: "Return outcome"
  outcome: ["ACCEPT","REJECT"]
```

26. Reason Code Taxonomy (Canonical)

```
reason_codes:
  E-PARSE: "YAML parse failure"
  E-CANON: "Canonicalization / CanonicalSortSpec failure"
  E-ID: "ID format / regex failure"
  E-CVID: "CVID mismatch"
  E-CHAIN: "Vault/Deed chain invalid or cyclic"
  E-SIG: "Signature policy failure"
  E-CONV: "Conversion rule mismatch"
  E-STAGE: "Invalid stage transition"
```

E-TURN: "Turn-order violation"

E-RULE: "Vault rule violation"

E-JOURNAL: "Journal linkage invalid"

E-REG: "Registry mirroring failure (reportables)"

PART X — STRUCTURAL TRUST THEOREM + PROOFS (DEFENSIBLE)

27. Structural Trust Theorem (Root Zero Form)

Given components $\{\mathscr{R}, \mathscr{A}, \mathscr{L}, \mathscr{C}, \mathscr{M}\}$, for any decision D within governed domain $\mathbb{D}$:

D is decidable (ACCEPT or REJECT)

D is recomputable offline from the continuity bundle

invalid D is structurally rejected in-domain (under GDA)

accepted D yields tamper-evident evidence (journal/registry)

Each component is necessary; together they are sufficient.

28. Necessity Proof (Failure Modes)

Without $\mathscr{R}$: no authoritative scope truth (anyone can mint “valid-looking” IDs).

Without $\mathscr{A}$: semantics become ambiguous; byte attacks possible; non-repudiation fails.

Without $\mathscr{L}$: history can be rewritten without detection.

Without $\mathscr{C}$: verification collapses to live authority; sovereignty fails.

Without $\mathscr{M}$: rules become advisory; bypass becomes possible.

29. Sufficiency Proof (Sketch)

$\mathscr{A}$ ensures CanonBytes ↔ CVID determinism.

$\mathscr{R}$ ensures authority truth via Vault YAML path + deed chain.

$\mathscr{M}$ ensures terminating deterministic evaluation.

$\mathscr{L}$ ensures immutable record by hash-chain + optional mirroring.

$\mathcal{C}$ ensures all inputs exist offline.

Therefore, Structural Trust holds in $\mathbb{D}$.

30. Recomputation Invariant (“Secret Zero” Detection)

recomputation_invariant:

statement: "If journal records ACCEPT, offline recomputation MUST yield ACCEPT"

mismatch_indicates:

- "tamper"

- "override"

- "non-canonical artifact"

- "policy divergence"

31. Root Zero Constraint: Ban Numeric Hierarchy Inference (Critical)

rootzero_constraints:

ban_numeric_hierarchy_inference: true

require_yaml_path_resolution: true

allowed_numeric_use:

- "ordering"

- "indexing"

- "stable labels"

forbidden_numeric_use:

- "authority inference"

- "ancestry inference"

- "governance scope inference"

PART XI — IMPLEMENTATION SPECIFICATION (COMPLETE)

32. Core Data Structures (Implementation-Grade)

32.1 Canonical Byte Builder

```
implementation:
  canon_bytes_builder:
    input: "YAML artifact"
    applies:
      - "ROOTZERO.CANON.YAML.1"
      - "CanonicalSortSpec (by class)"
      - "NFC normalization"
    output: "canonical bytes"
```

32.2 Conversion Engine

```
implementation:
  conversion_engine:
    computes:
      - "G1_global_unicode"
      - "G2_local_subset (if declared)"
    rejects_on:
      - "non-NFC inputs"
      - "undeclared alphabet"
      - "symbol outside tenant alphabet"
```

32.3 Vault Resolver (YAML Path Truth)

```
implementation:
  vault_resolver:
    truth_source: "vault_yaml_path"
    operations:
      - "resolve chain root->leaf"
      - "merge deterministically"
      - "reject cycles"
    output: "effective_vault"
```

32.4 Deed Resolver (Authority Chain)

```
implementation:
  deed_resolver:
    operations:
      - "resolve deed chain"
      - "verify dual signatures"
      - "enforce immutability constraints"
    output: "effective_authority_binding"
```

32.5 Journal/Registry Writers (Double-Entry)

```
implementation:
  recordkeeping_writers:
    journal:
      required: true
      hash_chain: "hash:blake3"
    registry:
```

```
    required: "policy-dependent"
    mirrors: "reportable actions only"
```

33. Network & Distribution Protocols (Minimal, Sovereign)

Root Zero does not require a blockchain. It requires survivable distribution.

```
distribution:
  continuity_bundle_transport:
    supported:
      - "offline media (USB, print, cold storage)"
      - "HTTPS object storage"
      - "air-gapped transfer"
  registry_distribution:
    models:
      - "append-only public feed"
      - "gossiped mirror network"
      - "periodic snapshots"
```

34. Security Threat Model (Complete)

```
threat_model:
  canonicalization_attacks:
    mitigations: ["NFC", "disallow YAML aliases/tags/comments", "CanonicalSortSpec"]
  tampering_attacks:
    mitigations: ["hash-chained journal", "bundle recomputation"]
  equivocation_attacks:
    mitigations:
```

- "registry mirroring for reportables"
- "public hash commitments"

key_compromise:

mitigations:

- "dual signatures (classical + PQC)"
- "reissuance + supersedence"

denial_of_service:

mitigations:

- "terminating mediation"
- "bounded rule graphs"

rollback_attacks:

mitigations:

- "journal prior_hash linkage"
- "registry snapshots / checkpoints"

35. Performance & Limits (Practical Guidance)

performance_notes:

global_unicode_coordinate:

complexity: "O(n) over code points"

storage: "big integer for long strings"

recommendation:

- "use global coordinate mainly for canonical ordering + verification"
- "use local subset coordinates for operational indexing where declared"

mediation:

complexity: "O(|V|+|E|) for DAG evaluation"

journaling:

complexity: "O(1) append; verification O(depth) by hash chain"

PART XII — APPLICATIONS (COMPLETE)

36. Digital Sovereignty & Court-Grade Audit

bundles enable independent verification without vendor access

canonical bytes + CVID allow legal defensibility

37. Disaster Recovery & “Basement Calculator Test”

A system passes if a verifier can validate everything offline long after infrastructure loss.

basement_calculator_test:

procedure:

- "obtain continuity bundle at time t"

- "assume all infrastructure destroyed"

- "verify canonical bytes, CVIDs, signatures, vault/deed resolution, mediation outcome, journal linkage"

passes_if:

- "verification succeeds offline"

38. Cross-Jurisdiction Accountability

local journal remains authoritative

registry provides public attestations for reportable events

39. AI Governance & Model Deployment

enforce deterministic release gates

immutable audit trail for who approved what, under which vault logic

40. Long-Horizon Infrastructure Control

stable identifiers

non-repudiable governance decisions

survivable evidence chains

PART XIII — APPENDICES (WORKED EXAMPLE + BUNDLE)

Appendix A — End-to-End Worked Example (Root Zero Profile)

A.1 Inputs (Vault + Deed + Event)

```
example_inputs:
  id: "LOVE-0007-0003"
  vault_declares:
    canonical_profile_id: "ROOTZERO.CANON.YAML.1"
    cvid_rule: "cvid:blake3:<hex>"
    conversion_rules:
      global_unicode: true
      local_subset:
        enabled: true
        alphabet: "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
        base_k: 26
    signature_policy:
      required: "dual"
      algorithms: ["ed25519","pqc_dilithium3"]
```

```
  registry_policy:
    reportable_actions: ["regulatory_reporting_required"]
 deed_binds:
  id: "LOVE-0007-0003"
  vault_cvid: "cvid:blake3:VAULT_PLACEHOLDER"
  signatures: ["sig:ed25519:...", "sig:pqc_dilithium3:..."]
 event:
  action: "regulatory_reporting_required"
  requester: "user:alice"
  signatures: ["sig:ed25519:...", "sig:pqc_dilithium3:..."]
  ts: "2025-01-15T10:30:00Z"
```

A.2 Conversion Outputs

```
example_conversions:
 global_unicode_coordinate:
  base_B: 1114112
  computed_from: "NFC('LOVE-0007-0003')"
  value: "PLACEHOLDER_BIGINT"
 local_subset_coordinate_for_token:
  token: "LOVE"
  base_k: 26
  value: 221410
```

A.3 Journal + Registry Writes

```
example_outputs:
```

```
journal_entry:
  ts: "2025-01-15T10:30:02Z"
  type: "decision"
  signer: "MEDIATOR.NODE.1"
  signature_policy: "dual"
  signatures: ["sig:ed25519:...", "sig:pqc_dilithium3:..."]
  payload:
    id: "LOVE-0007-0003"
    outcome: "ACCEPT"
    reason_code: "NONE"
    prior_hash: "hash:blake3:PREV"
  hash: "hash:blake3:CURR"
registry_entry:
  ts: "2025-01-15T10:30:03Z"
  type: "registry_mirror"
  signer: "REGISTRY.NODE.1"
  signature_policy: "dual"
  signatures: ["sig:ed25519:...", "sig:pqc_dilithium3:..."]
  payload:
    journal_ref:
      journal_hash: "hash:blake3:CURR"
      journal_ts: "2025-01-15T10:30:02Z"
    id: "LOVE-0007-0003"
```

```
    outcome: "ACCEPT"
    reportable_reason: "regulatory_reporting_required"
  hash: "hash:blake3:REG"
```

Appendix B — Continuity Bundle (Complete Offline Proof Packet)

```
continuity_bundle:
 bundle_version: "1"
 canonical_profile_id: "ROOTZERO.CANON.YAML.1"
 canonical_sort_specs:
  JournalEntry: ["ts","type","signer","signature_policy","signatures","payload","hash"]
  RegistryEntry: ["ts","type","signer","signature_policy","signatures","payload","hash"]
 conversion_rules:
  G1_global_unicode:
    base_B: 1114112
    formula: "Σ(code_point(symbol_i) * B^i)"
  G2_local_subset:
    enabled: true
    alphabet: "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
    base_k: 26
 vault_chain:
  - vault_cvid: "cvid:blake3:VAULT_PLACEHOLDER"
    vault_artifact: "INLINE_OR_POINTER"
 deed_chain:
  - deed_id: "cvid:blake3:DEED_PLACEHOLDER"
```

```
    deed_artifact: "INLINE_OR_POINTER"
  event:
    artifact: "INLINE_OR_POINTER"
    canonical_bytes_reference: "INLINE_OR_POINTER"
    event_cvid: "cvid:blake3:EVENT_PLACEHOLDER"
  recordkeeping:
    journal:
      prior_hash: "hash:blake3:PREV"
      current_hash: "hash:blake3:CURR"
      entry: "INLINE_OR_POINTER"
    registry:
      required: true
      current_hash: "hash:blake3:REG"
      entry: "INLINE_OR_POINTER"
  signatures:
    pubkeys: "INLINE_OR_POINTER"
    proofs: "INLINE_OR_POINTER"
```

FINAL SUMMARY (Root Zero–Aligned)

RSBIS in Root Zero form is a complete identity + governance architecture where:

Canonical YAML + CanonicalSortSpec produce canonical bytes.

CVID = cvid:blake3:<hex> anchors every artifact immutably.

Conversion rules explicitly define global Unicode coordinates and optional tenant-local coordinates.

Vault YAML path + Deed chain define authority truth.

Mediation is deterministic and terminating, producing ACCEPT/REJECT with reason codes.

Journal is authoritative; Registry is public mirror for reportables, enforcing double-entry accountability.

Continuity bundles enable offline recomputation, making hidden overrides detectable (recomputation invariant).

End of Master Note.

# References


• IETF RFC 2119: Key words for use in RFCs to Indicate Requirement Levels.

• Unicode Consortium. Unicode Normalization Forms (UAX #15, NFC).

• IETF RFC 8032: Edwards-Curve Digital Signature Algorithm (EdDSA).

• IETF RFC 6962 / RFC 9162: Certificate Transparency.

• BLAKE3 authors. The BLAKE3 Hashing Framework.