

Appendix E: Schemas / Registries Roadmap

Published as Supplement S5 for Kingdom Conformance RFC v1.3.6. Implementer-facing interoperability roadmap (informative; adds no requirements). Aligned to Sections 5 (determinism), 7 (Conformance Packs), and 14 (registries).

Goal: Make cross-vendor replay practical by publishing a minimal, versioned set of machine-readable schemas and signed registries. An independent verifier should be able to parse a Conformance Pack, resolve required identifiers via registry snapshots, and deterministically reproduce PASS/FAIL/HOLD outcomes.

E1 Minimal artifact schemas (recommended publication set)

- Conformance Statement (kc.conformance_statement@1): level; declared scope and action class; covered receipt types and control points; audit window (or selection rule); bindings (Policy Pack / Validator / CanonicalFormID, plus optional registry snapshot refs); explicit exclusions; signature reference.
- Receipt (kc.receipt@1): receipt_type; action_class; scope; Policy Pack id/version; Validator id/version; CanonicalFormID; decision + reason_codes; evidence_handles; decision timestamp + declared freshness bounds; signature reference.
- Permit (L2+) (kc.permit@1): permit_id; minted_after (receipt/decision reference); scope + audience; issued_utc + expires_utc; signature reference.
- Policy Pack metadata (kc.policy_pack@1): policy_id; version; declared profile/predicates; signed change history reference; optional human-readable summary.
- Validator contract metadata (kc.validator_contract@1): validator_id; version; output contract; reason code registry handle (if published); declared determinism boundary; test vectors reference (including negative and freshness-boundary cases).
- Evidence-handle record (kc.evidence_handle@1, optional): digest/reference type; redaction/selective-disclosure descriptor; retrieval instructions (public-safe); access note; redaction and retention policy.

E2 Minimal registries (IANA-style: versioned, signed, replay-stable)

- Receipt Type Registry: allowed receipt types and meanings (DATA, TRAIN, EVAL, DEPLOY, OPERATE, TRANSFER).
- Control Point Registry: where permits are validated (compute admission; registry/promotion; transfer/egress; externalization; deployment change).
- Action Class Registry: stable action-class identifiers used in receipts and permits (recommended: short, dot-delimited tokens).
- CanonicalFormID Registry: canonicalization rulesets used for deterministic replay and signature verification.

E2 Minimal registries (continued)

- Evidence Handle Type Registry: acceptable evidence-handle forms and how they satisfy predicates.
- Reason Code Registry: stable reason-code identifiers for each policy/validator version; meanings must not change without a version bump.
- Policy Pack Profile Registry: program/domain profiles that declare required predicates, receipt types, and control points for a scope.

For L1+ claims, Conformance Packs should include registry snapshot references (as evidence handles) for any registry required to interpret identifiers in scope. If a verifier cannot resolve a required identifier, the safe outcome is HOLD (not PASS).

E3 Versioning and compatibility (recommended)

- Semantic versioning for registries/schemas: PATCH = editorial clarifications; MINOR = additive, backwards compatible; MAJOR = changes that can alter replay outcomes for the same declared inputs (including canonicalization).
- Never change the meaning of an existing identifier without a version bump. Prefer adding new identifiers and marking old ones deprecated.
- Parsers should ignore unknown optional fields, but treat missing required fields (per schema/profile) as HOLD or FAIL per policy.
- During replay, freshness predicates should be evaluated against the receipt's recorded decision timestamp (or explicitly recorded `evaluation_time`), not the verifier's wall-clock time.

E4 Reviewer fast checks (approval-route accelerators)

- Schemas: a verifier can validate Conformance Packs against the published schemas with no privileged payload access.
- Registries: required registry snapshots resolve; unknown required identifiers deterministically yield HOLD.
- Replay: offline replay reproduces recorded PASS/FAIL/HOLD and reason codes under pinned Policy Pack / Validator / CanonicalFormID.
- Negative vectors: missing snapshot, stale evidence beyond freshness bounds, and policy/version mismatch produce HOLD or FAIL as specified.
- Fail-closed posture: when evidence is missing, stale, inconsistent, unverifiable, or insufficient to decide deterministically, HOLD blocks sensitive side effects.

Contact: MeridianVerity@proton.me | Status: Public Draft (public-safe) | Text license: CC BY 4.0 (text). Patent rights are not licensed by this publication.