LARRI
VERIFICATION PROTOCOL

Ledger of Attested Records and Receipts for Integrity

Volume III of V

LARRI Profiles and Extensions Specification

Version 0.1.0

Purpose. Public semantic contracts for adding optional assurance conditions, registered artifacts, algorithms, carriers, evidence, additional trust inputs, result contracts, verification checks, bounded claims, and residual limitations without changing the base protocol or disclosing proprietary decision behavior.

Property	Value
Document type	Normative profiles and extensions specification
Publication set	LARRI 0.1.0, Volume III of V
Protocol	LARRI - Ledger of Attested Records and Receipts for Integrity
Version	0.1.0
Status	Experimental
Use boundary	Synthetic laboratory and supervised evaluation only
Publication date	27 July 2026
Official website	https://larriverification.org

Assurance boundary. Profiles add only the exact bounded public assurance conditions they define and verify under explicitly selected contracts and offline-supplied inputs. They do not establish factual truth, identity, authority, completeness, independently trusted time, domain correctness, legal or regulatory sufficiency, production readiness, universal absence of sensitive information, or correctness of the private decision process.

Normative specification

Contents

Section	Title
1	Authority, scope, closed selection, and stage boundary
2	Profile record and result-contract binding
3	Profile-combination contract
4	Extension contract and added trust
5	Base registries and artifact-type integration
6	Lineage and derivation extension capacity
7	Checkpoint and inclusion extension capacity
8	External time-evidence capacity
9	Witness and transparency evidence capacity
10	Privacy-screening evidence capacity
11	Alternative carrier extension capacity
12	Multi-signature and cryptographic agility
13	Application and domain semantic profiles
14	Claims, limitations, and refusal behavior
15	Enablement and conformance boundary
A	Semantic contract templates
B	Base registry state
C	Profile authoring checklist

Document relationship. Volume I controls assurance meaning, roles and trust boundaries, verification targets, the eleven-stage dependency order, result meaning, and non-goals. Volume II controls exact base fields, schemas, lexical forms, canonical bytes, signature messages, trust objects, artifact classification, refusals, and result bytes. This volume owns optional profile and extension semantics. Volume IV owns conformance evidence, vector identity, reports, claims, and publication gates. Volume V is informative.

1. Authority, scope, closed selection, and stage boundary

LARRI - Ledger of Attested Records and Receipts for Integrity - uses profiles to add bounded public assurance conditions to the `profiled-bundle` target. An extension introduces one separately registered artifact, algorithm, carrier, evidence form, trust form, signature suite or policy, result contract, commitment role, or public check. Profiles and extensions preserve the base protocol rather than rewriting it.

Profiles and extensions define only public observable inputs, validation rules, checks, refusals, claims, result bindings, and residual limitations. They do not prescribe source layout, modules, orchestration, private evidence handling, product workflow, models, prompts, thresholds, policies, or proprietary decision mathematics.

LARRI-010-PROF-SCOPE-001 A profile MUST state every additional artifact type, required extension, operator-controlled trust or policy input, public check, applicable result contract, permitted claim, mandatory limitation, resolved limitation, and profile-specific refusal required for its scope.

LARRI-010-PROF-SCOPE-002 A profile or extension MUST NOT permit artifact-contained material to establish, replace, or override the external trust used to verify that artifact.

LARRI-010-PROF-SCOPE-003 A profile or extension MUST NOT waive a failed core requirement, alter a core verification target, or reinterpret core PASS or REFUSE.

LARRI-010-PROF-SCOPE-004 Normative requirement identifiers in this volume MUST be globally unique across the LARRI 0.1.0 publication set. A duplicate requirement definition is a publication defect.

1.1 Closed profile selection

Profiles are selected by exact identifier under the semantic contracts identified by `contracts_sha256`. There is no implicit inheritance, automatic dependency closure, runtime constraint merging, inferred profile, base-profile alias, fallback from `profiled-bundle` to `bundle`, or selection by filename, path, media type, or content sniffing.

LARRI-010-PROF-SELECT-001 The receipt and bundle targets MUST emit an empty profiles array. A selected profile is evaluated only under the profiled-bundle target.

LARRI-010-PROF-SELECT-002 The profiled-bundle target MUST receive at least one explicitly selected profile identifier. Stage 3 MUST normalize and resolve that selection, and the normalized result MUST record the resolved profile identifiers in canonical profile order.

LARRI-010-PROF-SELECT-003 An unknown, reserved, deprecated, disabled, inapplicable, or unsupported selected profile MUST prevent PASS and MUST produce CONTRACT_RESOLUTION_REFUSED with PROFILE_UNSUPPORTED at Stage 3.

LARRI-010-PROF-SELECT-004 A verifier MUST NOT silently substitute a base-profile alias, drop a selected profile, infer a missing profile, or fall back to the unprofiled bundle target.

1.2 Semantic-contract boundary

Profile status, rank, applicability, canonical semantic-record digests, exact combinations, required extensions, added trust contracts, result contracts, checks, claims, limitations, and refusal codes are resolved only from the semantic contract layer identified by `contracts_sha256`. Network discovery, latest-version lookup, implementation-private aliases, vector-set identity, and release metadata are outside Stage 3 resolution.

LARRI-010-PROF-CONTRACTS-001 Every selected profile, exact profile combination, required extension, required artifact type, required check, additional trust-input contract, result contract, claim, limitation, and profile-specific refusal code MUST resolve to exactly one applicable enabled semantic record in the pinned contracts package.

1.3 Stage 3 resolution and Stage 10 evaluation

Stage 3 contract_resolution	Stage 10 commitments and selected profiles
Resolves every selected profile and the canonical profile set.	Evaluates only conditions belonging to records that resolved successfully.
Resolves one exact enabled combination when more than one profile is selected.	Discovers required profile artifacts by manifest artifact_type and evaluates their registered checks.
Resolves every required extension, artifact type, check, trust-input contract, result contract, claim, limitation, and refusal code.	Produces only profile-condition PASS, REFUSE, SKIP, NOT_PRESENT, or UNVERIFIED behavior already authorized by the resolved records.
Owns missing, unknown, disabled, deprecated, inapplicable, unsupported, and ambiguous selection failures.	Does not rediscover profiles, merge records, select alternatives, or emit selection failures.

LARRI-010-PROF-RESOLVE-001 At Stage 3, contract_resolution MUST resolve the selected profiles, the exact canonical profile set, the applicable enabled profile-combination record, every required extension, every required artifact type, every required public check, every additional trust-input contract, every result contract, and every profile-specific refusal code.

LARRI-010-PROF-RESOLVE-002 A missing, unknown, reserved, deprecated, disabled, inapplicable, unsupported, or ambiguous record required by profile selection MUST produce CONTRACT_RESOLUTION_REFUSED at Stage 3 with the applicable enabled detail code. PROFILE_SELECTION_REQUIRED, PROFILE_UNSUPPORTED, PROFILE_COMBINATION_UNSUPPORTED, AMBIGUOUS_PROFILE_COMBINATION, REQUIRED_EXTENSION_UNSUPPORTED, UNKNOWN_REQUIRED_IDENTIFIER, and DISABLED_OR_INAPPLICABLE_IDENTIFIER are used only as defined by Volume II.

LARRI-010-PROF-RESOLVE-003 Stage 10 MUST evaluate only the artifacts, evidence, commitments, checks, claims, limitations, and refusal conditions of profiles and extensions that resolved successfully at Stage 3. It MUST NOT perform profile selection, alternative selection, dependency merging, or selection-failure reporting.

2. Profile record and result-contract binding

A `larri/profile-record` is a closed semantic contract for one additional assurance. The canonical profiles registry binds `profile_id` to status, rank, applicable target, and the SHA-256 digest of the exact canonical semantic profile record. Conformance evidence is bound separately by Volume IV.

SCHEMATIC - NOT MACHINE-VALID

```
{
  "schema_id": "larri/profile-record",
  "schema_version": "0.1.0",
  "protocol_id": "larri",
  "protocol_version": "0.1.0",
  "profile_id": "example.org/example-profile-v1",
  "status": "reserved",
  "verification_target": "profiled-bundle",
  "required_extensions": [],
  "required_artifact_types": [],
  "required_checks": [],
  "required_trust_inputs": [],
  "result_contract_id": "larri/verifier-result",
  "permitted_claims": [],
  "mandatory_limitations": ["NO_COMPLETENESS_GUARANTEE"],
  "resolved_limitations": [],
  "refusal_codes": []
}
```

Member	Semantic contract
schema_id	Constant `larri/profile-record`.
schema_version	Constant `0.1.0`.
protocol_id / protocol_version	Constants `larri` and `0.1.0`.
profile_id	Stable namespaced semantic identifier.
status	enabled, reserved, deprecated, or disabled; only enabled is selectable.
verification_target	Exactly `profiled-bundle` in 0.1.0.
required_extensions	Complete extension dependency list.
required_artifact_types	Complete profile-artifact type list.
required_checks	Complete Stage 10 public check list.
required_trust_inputs	Complete operator-controlled added trust or policy-input list.
result_contract_id	One enabled closed result contract applicable to the target and profile.
permitted_claims	Closed bounded statements available only after dependent checks PASS.
mandatory_limitations	Limitations that remain after profile PASS.
resolved_limitations	Only limitations directly replaced by verified narrower properties.
refusal_codes	Every profile-specific Stage 10 blocking refusal enabled by the record.

LARRI-010-PROF-RECORD-001 A profile record MUST be closed and MUST contain every member required by the larri/profile-record schema, including protocol identity and result_contract_id.

LARRI-010-PROF-RECORD-002 Every set-like profile array MUST reject duplicates and use the ordering declared by the applicable registry or, where no registry rank exists, exact ASCII order.

LARRI-010-PROF-RECORD-003 The profiles registry MUST bind each profile_id to the SHA-256 digest of the exact canonical semantic profile record, its status, rank, and applicable target. Evidence-layer and release-layer identities are not members of that binding.

LARRI-010-PROF-RECORD-004 A permitted profile claim MUST identify the exact profile identifiers, verification target, contracts_sha256, applicable result_contract_id, outcome, and mandatory limitations. Added trust identity and actual-use information MUST be retained as required by the selected result contract.

LARRI-010-PROF-RECORD-005 A profile MUST NOT list a resolved limitation unless its required public evidence and checks directly establish the narrower property that replaces that limitation.

LARRI-010-PROF-RECORD-006 A semantic profile record MUST NOT contain vector_set_sha256, fixture digests, expected-result digests, conformance-report identities, release-envelope identities, implementation identifiers, or private decision configuration.

2.1 Deterministic ordering

Member	Required order
required_extensions	extension-registry rank, then exact ASCII identifier
required_artifact_types	artifact-type registry rank, then exact ASCII identifier
required_checks	check-registry rank
required_trust_inputs	trust-contract registry rank, then exact ASCII identifier
permitted_claims	claim-registry rank, then exact ASCII identifier
mandatory_limitations / resolved_limitations	limitation-registry rank, then exact ASCII identifier
refusal_codes	refusal-registry rank, then exact ASCII code

2.2 Result-contract selection

LARRI-010-PROF-RESULT-001 Every profile record and profile-combination record MUST name exactly one enabled result_contract_id that is applicable to profiled-bundle and capable of representing the complete selected public result context.

LARRI-010-PROF-RESULT-002 A profile MAY use larri/verifier-result only when its existing closed fields truthfully retain every selected input identity and actual-use fact required by the profile.

LARRI-010-PROF-RESULT-003 When added trust, policy, carrier, or other selected identities cannot be retained by larri/verifier-result, the profile MUST select a distinct registered closed result schema or other closed result contract. It MUST NOT widen larri/verifier-result with unknown members.

3. Profile-combination contract

More than one profile may be selected only through one closed `larri/profile-combination` record. The record is not a merge instruction. It is the complete combined semantic contract for one exact canonical ordered profile set.

SCHEMATIC - NOT MACHINE-VALID

```
{
  "schema_id": "larri/profile-combination",
  "schema_version": "0.1.0",
  "protocol_id": "larri",
  "protocol_version": "0.1.0",
  "combination_id": "example.org/example-combination-v1",
  "status": "reserved",
  "verification_target": "profiled-bundle",
  "profiles": ["example.org/profile-a-v1", "example.org/profile-b-v1"],
  "required_extensions": [],
  "required_artifact_types": [],
  "required_checks": [],
  "required_trust_inputs": [],
  "result_contract_id": "larri/verifier-result",
  "permitted_claims": [],
  "mandatory_limitations": ["NO_COMPLETENESS_GUARANTEE"],
  "resolved_limitations": [],
  "refusal_codes": []
}
```

3.1 Exact-set resolution

Step	Rule
1	Normalize selected profile identifiers by profile-registry rank and then exact ASCII.
2	Match that exact canonical ordered set; input order has no semantic effect.
3	Require no more than one enabled matching combination under the selected contracts_sha256.
4	Use PROFILE_COMBINATION_UNSUPPORTED when no enabled match exists.
5	Use AMBIGUOUS_PROFILE_COMBINATION when more than one enabled match exists.

LARRI-010-PROF-COMBINE-001 A multiple-profile selection MUST normalize to one exact ordered profile set and MUST match exactly one enabled profile-combination record. A missing match MUST produce CONTRACT_RESOLUTION_REFUSED with PROFILE_COMBINATION_UNSUPPORTED; more than one enabled match MUST produce CONTRACT_RESOLUTION_REFUSED with AMBIGUOUS_PROFILE_COMBINATION.

LARRI-010-PROF-COMBINE-002 A combination record MUST state the complete combined required extensions, artifact types, checks, trust inputs, result contract, claims, mandatory limitations, resolved limitations, and refusal codes. Every set-like member MUST reject duplicates and use the applicable registry order or exact ASCII order. A verifier MUST use those exact lists and MUST NOT derive them through runtime merging.

LARRI-010-PROF-COMBINE-003 A semantic combination record MUST NOT contain vector-set, fixture, expected-result, report, implementation, or release-envelope identities. Volume IV binds conformance evidence to the canonical combination-record digest separately.

LARRI-010-PROF-COMBINE-004 Profile order in the normalized result MUST follow profile-registry rank and then ASCII identifier; input order MUST NOT change verification semantics.

4. Extension contract and added trust

A closed `larri/extension-record` introduces one material public addition to the protocol surface. An extension does not execute merely because its bytes are present. It executes only when Stage 3 resolves it as required by the selected profile or combination and all prerequisites permit its checks.

SCHEMATIC - NOT MACHINE-VALID

```
{
  "schema_id": "larri/extension-record",
  "schema_version": "0.1.0",
  "protocol_id": "larri",
  "protocol_version": "0.1.0",
  "extension_id": "example.org/example-evidence-v1",
  "extension_class": "evidence",
  "status": "reserved",
  "applicable_targets": ["profiled-bundle"],
  "contract_id": "example.org/example-evidence-contract-v1",
  "contract_sha256": "<64 lowercase hex>",
  "introduced_artifact_types": [],
  "introduced_checks": [],
  "introduced_trust_inputs": [],
  "introduced_result_contracts": [],
  "introduced_algorithms": [],
  "introduced_signature_suites": [],
  "introduced_signature_policies": [],
  "introduced_carriers": [],
  "introduced_commitment_roles": [],
  "refusal_codes": []
}
```

Extension class	Minimum public semantic definition
artifact	Schema, identity, signed members, target applicability, permitted content roles, trust treatment, and check behavior.
evidence	Subject binding, required availability, verification method, trust, claim, result contract, and residual limitations.
check	Prerequisites, stage, target applicability, public status, refusal codes, safe details, and result position.
trust	Closed input schema, independent supply, canonical identity, validation, authority or policy meaning, use stage, and result traceability.
result	Closed result schema or projection, canonicalization, identity retention, ordering, and compatibility with selected profiles.
carrier	Logical subject mapping, carrier identity, deterministic encoding where claimed, safety, limits, signature context, artifact_type behavior, and claim limits.
canonicalization or digest algorithm	Exact inputs, outputs, encoding, parameters, domain construction, error behavior, applicability, and evidence required before enablement.
signature suite or policy	Exact messages, key and signature encodings, strict acceptance, membership, counting, ordering, trust, downgrade behavior, and actual-use traceability.
commitment role	Exact relationship semantics, namespace, applicable subjects, and claim limitations.

LARRI-010-PROF-EXT-001 An extension capable of affecting accepted bytes, trust, required checks, PASS, REFUSE, primary-refusal selection, normalized output, claims, or limitations MUST be registered and MUST define every observable semantic behavior that can produce that effect. Volume IV evidence is required before the record may be enabled.

LARRI-010-PROF-EXT-002 Unknown or unselected extension bytes MAY remain subject to generic manifest path, artifact-type, content-role, length, and digest checks when permitted, but they MUST NOT execute extension checks, satisfy a profile, contribute semantic assurance, authorize a claim, or count as extension conformance.

LARRI-010-PROF-EXT-003 Requiredness is determined only by the selected profile or exact profile combination. An unknown, disabled, deprecated, inapplicable, or unsupported required extension MUST produce CONTRACT_RESOLUTION_REFUSED with REQUIRED_EXTENSION_UNSUPPORTED at Stage 3.

LARRI-010-PROF-EXT-004 An extension MUST NOT add unregistered members to a closed base object. It uses a separately registered artifact schema, a separately registered carrier or result schema, or another exact extension mechanism permitted by Volume II.

LARRI-010-PROF-EXT-005 Only one enabled applicable extension record may satisfy a required extension identifier. Reserved, deprecated, disabled, ambiguous, or inapplicable records do not satisfy a required extension in 0.1.0.

LARRI-010-PROF-EXT-006 A semantic extension record MUST NOT contain vector_set_sha256, fixture or expected-result digests, conformance-report identities, implementation identities, publication metadata, or release-envelope identities.

4.1 Added trust and result traceability

Required definition	Meaning
Closed input schema	The exact expressible trust or policy object.
Independent supply channel	How the operator supplies it outside the artifact being verified.
Validation and canonical identity	Strict decoding, canonicalization, identity, and failure behavior.
Target and profile binding	The exact target, profile, subject, and policy scope authorized.
Trust or policy semantics	What a key, fingerprint, authority, or policy relation does and does not establish.
Use stage and check	The exact point at which the input is required and the public check that records its use.
Closed result traceability	The exact identity and actual-use fields retained in the selected result contract.

LARRI-010-PROF-EXTTRUST-001 An extension that adds operator-controlled trust or policy input MUST define the closed input schema, independent supply channel, strict validation, canonical identity, target and profile binding, key, fingerprint, authority, or policy semantics, and the exact stage and public check at which the input is used.

LARRI-010-PROF-EXTTRUST-002 An added-trust extension MUST NOT be enabled until the selected profile or combination names a closed result contract that retains the exact added input identity and the actual-use information required to interpret PASS or REFUSE. Hidden trust and unrecorded policy dependencies are prohibited.

5. Base registries and artifact-type integration

The base five-volume publication defines the mechanism without manufacturing a concrete optional assurance. The `profiled-bundle` target remains enabled, but the base semantic registries contain no enabled profile, no enabled profile combination, and no enabled extension. A companion contracts package may add records only by satisfying this volume and the evidence requirements of Volume IV.

Base registry	Enabled state in the five-volume publication
profiles	No enabled entries.
profile_combinations	No enabled entries.
extensions	No enabled optional extension entries.
profiled-bundle target	Enabled target; without a selected enabled profile, Stage 3 refuses with PROFILE_SELECTION_REQUIRED.

LARRI-010-PROF-BASE-001 A base receipt or bundle PASS MUST NOT imply lineage, inclusion, trusted time, transparency, privacy screening, semantic correctness, completeness, or any companion-profile property.

LARRI-010-PROF-BASE-002 A profile PASS is valid only when every prerequisite core check and every required profile check passes; a profile cannot waive or replace a core REFUSE.

LARRI-010-PROF-BASE-003 The profiled-bundle target MUST produce CONTRACT_RESOLUTION_REFUSED with PROFILE_SELECTION_REQUIRED at Stage 3 when no enabled profile is selected. It MUST NOT invent a profile or defer the failure to Stage 10.

LARRI-010-PROF-BASE-004 Healthcare, PHI-handling, clinical, insurer, hospital, and other domain-specific profiles are separately versioned companion contracts. Time, witness, transparency, and similar optional assurances may be added only through separately registered profiles or extensions and MUST NOT add fields or enabled identifiers to the base five-volume publication.

LARRI-010-PROF-BASE-005 The base publication MUST NOT enable a profile, profile combination, or extension solely to manufacture a positive profiled-bundle path or demonstrate the mechanism. An enabled record requires a real, nonredundant, fully specified, domain-neutral offline assurance and complete public result traceability.

5.1 Path-independent profile-artifact discovery

Every required profile artifact is classified by the manifest entry member `artifact_type`. The `path` member identifies location only. A filename, suffix, directory convention, media type, path prefix, heuristic parser, or content sniffing rule cannot select profile semantics.

LARRI-010-PROF-ARTIFACT-001 A profile or extension MUST discover each required bundle artifact only through the manifest entry artifact_type. It MUST NOT select or infer an artifact from its path, filename, extension, directory, media type, content bytes, or heuristic parsing.

LARRI-010-PROF-ARTIFACT-002 Every registered profile or extension artifact type MUST declare its permitted content roles, required cardinality, duplicate and multiple-match behavior, subject identity, applicable checks, and failure behavior.

5.2 Opaque extension files

An unknown application-namespaced or unselected extension file may still change generic bundle integrity because its declared path, role, length, or digest may fail. That generic integrity effect is not execution or verification of the extension assurance.

Assurance distinction. Generic manifest integrity answers whether the observed prepared directory matches its signed declaration. Extension assurance answers whether selected registered semantics executed and passed. The first may apply to opaque bytes; the second never does.

6. Lineage and derivation extension capacity

This section defines authoring requirements only. It does not enable a lineage scheme. A lineage extension may bind identified source and output subjects to an identified transform under separately specified trust without disclosing proprietary transformation logic.

LARRI-010-PROF-LINEAGE-001 A derivation record MUST reference every source and output by content digest and MUST NOT modify a signed source artifact in place.

LARRI-010-PROF-LINEAGE-002 A free-form transform description MUST NOT substitute for a registered transform identifier when semantic interoperability is claimed.

LARRI-010-PROF-LINEAGE-003 A lineage extension contract MUST define exact source and output identities, transform identifier, source and output ordering, duplicate behavior, parameter commitment and canonicalization, graph-size limits, cycle behavior, trust treatment, artifact_type integration, and residual limitations.

LARRI-010-PROF-LINEAGE-004 A lineage PASS MUST NOT be represented as proof that the transform was correct or that every relevant source was disclosed.

Bounded claim. A passing lineage check may establish only the exact signed relationship among the identified source identities, output identities, transform identifier, committed parameters, and selected trust. It does not establish correctness, legality, quality, reversibility, completeness, or disclosure of every source.

7. Checkpoint and inclusion extension capacity

This section defines authoring requirements only. It does not enable a checkpoint, tree, accumulator, or proof system. A concrete scheme must define exact bytes and trust without relying on conventional or library-default behavior.

Required definition	Minimum semantic content
Leaf encoding	Subject identity, exact byte encoding, and domain separator.
Collection semantics	Ordered set, unordered set, or multiset; duplicate and size behavior.
Root calculation	Exact node encoding, domain separation, tree or accumulator rule, and odd-element behavior.
Proof encoding	Canonical inclusion or consistency proof bytes and strict rejection rules.
Empty collection	Invalid or one exact registered root.
Size binding	How collection size or tree size enters the commitment and proof.
Trust treatment	Signature, witness, or added trust contract for the checkpoint statement.

LARRI-010-PROF-CHECKPOINT-001 A checkpoint PASS establishes only the registered commitment and any successfully verified proof; it does not establish completeness of the producer's source universe.

LARRI-010-PROF-CHECKPOINT-002 The checkpoint algorithm MUST define exact leaf encoding, collection semantics, duplicate handling, root calculation, domain separation, empty-set behavior, odd-element behavior, proof malleability, canonical proof order, and size binding.

LARRI-010-PROF-CHECKPOINT-003 A proof MUST bind the exact checkpoint root, subject identity, collection or tree size where applicable, registered algorithm identifier, domain separator, and proof type.

8. External time-evidence capacity

Issuer `created_at`, outer `asserted_at`, and equivalent fields remain issuer assertions. This section does not enable a timestamping, certificate, clock, or network service. A concrete profile must be verifiable offline from supplied evidence and independently supplied trust inputs.

LARRI-010-PROF-TIME-001 A verifier MUST NOT elevate an issuer-asserted created_at, asserted_at, or equivalent field to trusted time without a selected time-evidence profile and successfully verified external evidence.

LARRI-010-PROF-TIME-002 Absent optional time evidence MUST produce NOT_PRESENT; absent required time evidence MUST produce the profile's registered blocking refusal.

LARRI-010-PROF-TIME-003 A time-evidence profile MUST state whether it establishes an upper bound, lower bound, interval, relative order, or other exact time relation and MUST NOT imply a stronger relation.

LARRI-010-PROF-TIME-004 A time-evidence extension MUST define the exact subject binding, evidence form and encoding, independent trust-input contract, verification algorithm, bounded time relation, use stage and check, result traceability, and residual clock, policy, coverage, retention, revocation, and long-term-validation limitations.

9. Witness and transparency evidence capacity

This section defines authoring requirements only. It does not require a live witness, public log, ledger, network lookup, or external service. Evidence and trust needed by an enabled profile must be supplied as offline protocol inputs.

Evidence form	Permitted bounded claim
External witness	The identified witness key attested to the identified subject bytes under the selected witness contract.
Transparency inclusion	The identified subject commitment appears under the exact verified log statement and proof.
Transparency consistency	The supplied statements satisfy the exact registered consistency relation.

LARRI-010-PROF-WITNESS-001 Witness and transparency trust inputs MUST be supplied independently of artifact-contained evidence, MUST bind the exact subject and selected profile, and MUST be retained through the selected closed result contract.

LARRI-010-PROF-WITNESS-002 A witness or transparency PASS MUST NOT be represented as identity, authority, factual correctness, completeness, or trusted time unless a separate selected profile establishes that narrower property.

LARRI-010-PROF-WITNESS-003 A witness or transparency extension contract MUST define the exact statement, subject binding, proof encoding, independent trust key or policy, checkpoint or tree-size binding, consistency relation where applicable, strict failure behavior, result traceability, and freshness limitations.

10. Privacy-screening evidence capacity

A privacy-screening profile may establish only that one identified ruleset or executable contract completed identified checks over one identified byte or artifact scope and produced one identified bounded public result. No profile may transform that completion statement into proof that sensitive information, PHI, secrets, credentials, regulated data, or any other class of information is universally absent.

Required public binding	Minimum meaning
ruleset_id and ruleset_digest	Identity of the exact screening contract applied.
scope identity	Exact bytes or artifact set screened.
coverage contract	Formats, encodings, parsers, inaccessible, encrypted, unsupported, and unparsed content treatment.
result	One closed bounded result code with deterministic failure behavior.
report commitment	Optional external report identity; no raw private matches in public results.

LARRI-010-PROF-PRIVACY-001 A privacy-screening PASS MUST be worded as completion of the identified screening checks and MUST NOT be represented as proof that the artifact contains no sensitive information.

LARRI-010-PROF-PRIVACY-002 Normalized results and public refusal descriptions MUST NOT expose matched private values, raw evidence, host paths, ruleset secrets, or free-form exception text.

LARRI-010-PROF-PRIVACY-003 A privacy-screening profile MUST define the exact screened scope, format and encoding coverage, treatment of inaccessible, encrypted, unsupported, and unparsed content, closed result codes, safe public details, and deterministic failure behavior.

LARRI-010-PROF-PRIVACY-004 Public counts, ruleset identifiers, result codes, and report commitments MUST be omitted or separately protected when their disclosure is sensitive or guessable.

11. Alternative carrier extension capacity

The base bundle subject is the prepared directory defined by Volume II. This section does not enable an archive or other byte-stream carrier. A byte-stream carrier extension must map its received bytes to the prepared-directory logical subject or another explicitly registered subject without changing base bundle semantics.

Carrier requirement	Required definition
Subject mapping	Exact mapping to the prepared-directory logical subject or another registered subject.
Dual identity	Received carrier identity and logical subject identity.
Encoding	Deterministic bytes only where explicitly claimed; complete duplicate-field and round-trip rules.
Safety	Path, extraction, link, allocation, decompression, and carrier-specific resource limits.
Signature context	Exact authenticated carrier or logical-subject context and downgrade behavior.
artifact_type	How contained artifacts preserve path-independent classification.
Claims	Exact limits on cross-carrier equality, equivalence, and reproducibility.

LARRI-010-PROF-CARRIER-001 Use of an alternative carrier MUST NOT imply generic interoperability with the broader carrier ecosystem unless that interoperability is separately specified and tested.

LARRI-010-PROF-CARRIER-002 Cross-carrier equivalence MUST be claimed only when one registered mapping and its Volume IV evidence demonstrate equivalent logical signed content and define whether carrier bytes, signatures, logical content, or only selected subject identities are equivalent.

LARRI-010-PROF-CARRIER-003 A carrier extension MUST bind its carrier identifier into the applicable signature or subject context and MUST define received carrier identity, logical subject identity, duplicate-field behavior, numeric and text encoding, unknown-member treatment, round-trip behavior, path and extraction safety, carrier-specific limits, artifact_type behavior, and strict failure semantics.

LARRI-010-PROF-CARRIER-004 Base bundle verification does not require byte-identical carrier construction. Any reproducible-carrier or byte-reproducible-packaging claim MUST be defined only by a separately registered carrier extension and MUST NOT be represented as Volume IV Reproducible Release.

12. Multi-signature and cryptographic agility

The baseline `larri/ed25519-sha256-v1` suite and `larri/one-ed25519-v1` policy remain unchanged. This section preserves optionality; it does not select or enable a post-quantum, hybrid, threshold, or alternative classical construction.

Required semantics	Additional suite or policy contract
Encodings	Exact public-key, signature, and policy-member bytes.
Message	Exact signed or prehashed message bytes and domain separation.
Strict rejection	Malformed, noncanonical, invalid, unsupported, duplicate, and extra-attestation behavior.
Membership and counting	Closed key set, roles, thresholds, duplicate-key treatment, and deterministic attestation order.
External trust	Closed trust form, target permissions, fingerprints or pins, and actual-use traceability.
Downgrade	No heuristic negotiation, alias, fallback, or silent substitution.
Evidence before enablement	Complete Volume IV positive, negative, boundary, downgrade, and multi-fault vectors.

LARRI-010-PROF-MULTISIG-001 A signature-policy identifier MUST be part of signed content and MUST resolve through the closed signature-policies registry.

LARRI-010-PROF-MULTISIG-002 A profile requiring an additional signature policy MUST resolve every required suite and policy at Stage 3 and MUST produce deterministic refusal at the applicable signature stage when a required attestation, key, suite, membership, or threshold condition is absent or invalid.

LARRI-010-PROF-MULTISIG-003 A multi-signature policy MUST define exact membership, key-role constraints, duplicate-key behavior, canonical attestation ordering, treatment of extra valid or invalid attestations, threshold counting, external trust, and result actual-use traceability.

LARRI-010-PROF-CRYPTO-001 A new digest, canonicalization, signature suite, or signature policy MUST use a distinct registered identifier and complete public verification semantics. Required Volume IV conformance vectors MUST exist before enablement; heuristic aliases and silent algorithm substitution are prohibited.

LARRI-010-PROF-CRYPTO-002 A profile that permits more than one cryptographic suite MUST define deterministic suite selection, downgrade refusal, ordering, and result traceability and MUST NOT use heuristic negotiation or fallback.

13. Application and domain semantic profiles

A separately versioned semantic profile may define a public assertion vocabulary, required commitments, public evidence, and bounded validation rules for one application domain. It must keep protocol integrity distinct from domain correctness and list every material claim that remains outside scope.

Proprietary boundary. An application may bind private policies, models, prompts, thresholds, source material, internal evidence, or decision outputs by digest. Base interoperability and ordinary profile verification do not require disclosure or re-execution of those materials unless the selected public claim expressly depends on separately defined public evidence.

LARRI-010-PROF-SEM-001 A semantic profile MUST distinguish protocol integrity checks from domain-correctness checks and MUST state every material claim that remains outside its scope.

LARRI-010-PROF-SEM-002 A semantic profile MUST define one authoritative namespace, exact assertion types, public value or commitment rules, required artifacts and evidence, public checks, result contract, permitted claims, residual limitations, and refusal behavior. Conformance evidence is owned separately by Volume IV.

LARRI-010-PROF-SEM-003 Base LARRI conformance MUST remain achievable by independently engineered implementations that share no proprietary decision engine, source code, internal data model, or software architecture.

LARRI-010-PROF-SEM-004 A domain profile is a separately versioned companion contract. Healthcare, PHI handling, HIPAA-oriented controls, clinical evidence, insurer or hospital workflow, and similar requirements MUST NOT silently add an identifier, member, or rule to the base five-volume publication.

14. Claims, limitations, and refusal behavior

Every profile defines the exact bounded statement permitted after successful verification. Profile claims are additive only within `profiled-bundle`. A claim does not erase a core or profile limitation unless the selected semantic record directly resolves that limitation and every dependent public check passes.

Condition	Normalized behavior
Required profile artifact absent or invalid	Stage 10 REFUSE under the profile-specific code resolved at Stage 3.
Required added trust absent or invalid	REFUSE under the stage and check defined by the selected trust extension and result contract.
Optional profile artifact absent	NOT_PRESENT.
Signed optional assertion or commitment cannot be recomputed	UNVERIFIED.
Profile check blocked by failed or unsafe prerequisite	SKIP.
Required profile check passes	PASS for that check; overall PASS only when every required base and profile check passes.
Unselected extension file present	Generic byte integrity only; no semantic claim.

LARRI-010-PROF-CLAIM-001 A user-facing profiled-bundle PASS statement MUST name profiled-bundle, the exact selected profiles, contracts_sha256, the applicable result_contract_id, the PASS outcome, and every mandatory limitation produced by the selected profile or combination.

LARRI-010-PROF-CLAIM-002 A permitted profile claim MAY be made only when every check on which that claim depends has status PASS.

LARRI-010-PROF-CLAIM-003 A mandatory limitation MUST remain unless one selected passing profile or combination lists it in resolved_limitations and its required public evidence directly establishes an explicitly bounded replacement property. Factual-truth, authority, completeness, trusted-time, privacy-absence, and domain-correctness limitations MUST NOT be removed by broader wording or inference; universal privacy absence is never a permitted claim.

LARRI-010-PROF-CLAIM-004 Every profile-specific refusal code MUST resolve and be enabled at Stage 3, MUST belong to Stage 10 profile-condition evaluation, MUST use deterministic refusal-registry rank, and MUST preserve all earlier base refusal precedence. A profile MUST NOT define a private competing order.

15. Enablement and conformance boundary

Volume III defines semantic completeness required before a profile or extension may be enabled. Volume IV owns vector objects, fixtures, frozen expected results, `vector_set_sha256`, reports, conformance claims, and publication gates. A semantic record never embeds its evidence identity.

Coverage class required before enablement	Required consequence
Positive behavior	Every enabled profile, extension, result contract, and exact combination has one valid path where applicable.
Refusals	Every profile-specific refusal and every Stage 3 resolution failure is exercised directly.
Boundaries	Size, count, graph, time, proof, threshold, encoding, and carrier boundaries defined by the contract.
Multi-fault precedence	Earlier base refusals, safe sibling collection, and downstream SKIP.
Claim safety	Exact permitted wording, mandatory limitations, no overclaim, and privacy-absence prohibition.
Limitation transitions	Every retained and directly resolved limitation state.
Opaque extensions	Generic integrity without semantic execution or conformance.
Interoperability	Required only when the corresponding optional claim is made.

LARRI-010-PROF-CONF-001 A profile conformance claim MUST inherit the complete Volume IV scope tuple and bind every selected semantic profile-record digest, required extension contract digest, and applicable combination-record digest. Volume IV binds the separate vector_set_sha256 evidence identity.

LARRI-010-PROF-CONF-002 A profile or extension is conforming only when the implementation also conforms for every prerequisite core target and required base contract.

LARRI-010-PROF-CONF-003 Before a profile, extension, combination, added-trust contract, or result contract is enabled, every normative branch, refusal, status transition, boundary, claim boundary, limitation transition, opaque-extension behavior, and relevant multi-fault interaction MUST have direct executable Volume IV coverage.

LARRI-010-PROF-CONF-004 Unknown extension preservation without execution MUST be reported as opaque or unsupported as required and MUST NOT be counted as conformance to that extension assurance.

LARRI-010-PROF-CONF-005 No profile conformance claim may PASS when the selected base contracts, semantic profile or combination records, extension contracts, registries, evidence index, vectors, or frozen expected outputs prescribe inconsistent behavior for the same scope.

LARRI-010-PROF-EXAMPLE-001 Every example containing ellipses, placeholders, synthetic hashes, incomplete signatures, or omitted required members MUST be labeled SCHEMATIC - NOT MACHINE-VALID. Only a complete example that validates against every applicable public schema and registry may be labeled machine-valid.

Reference implementation boundary. Reference implementation output is not an oracle. Authoring tools may help produce candidate fixtures, but Volume IV expected results must be derived from the approved public rules, independently checked, frozen, and compared without regeneration during ordinary conformance execution.

A. Semantic contract templates

A.1 Profile-record required members

SCHEMATIC - NOT MACHINE-VALID

```
schema_id, schema_version, protocol_id, protocol_version,
profile_id, status, verification_target,
required_extensions, required_artifact_types, required_checks,
required_trust_inputs, result_contract_id, permitted_claims,
mandatory_limitations, resolved_limitations, refusal_codes
```

A.2 Profiles-registry entry

SCHEMATIC - NOT MACHINE-VALID

```
profile_id, status, rank, applicable_target,
profile_record_sha256
```

A.3 Profile-combination required members

SCHEMATIC - NOT MACHINE-VALID

```
schema_id, schema_version, protocol_id, protocol_version,
combination_id, status, verification_target, profiles,
required_extensions, required_artifact_types, required_checks,
required_trust_inputs, result_contract_id, permitted_claims,
mandatory_limitations, resolved_limitations, refusal_codes
```

A.4 Extension-record required members

SCHEMATIC - NOT MACHINE-VALID

```
schema_id, schema_version, protocol_id, protocol_version,
extension_id, extension_class, status, applicable_targets,
contract_id, contract_sha256,
introduced_artifact_types, introduced_checks,
introduced_trust_inputs, introduced_result_contracts,
introduced_algorithms, introduced_signature_suites,
introduced_signature_policies, introduced_carriers,
introduced_commitment_roles, refusal_codes
```

Status	0.1.0 effect
enabled	May be selected and may satisfy a required semantic record.
reserved	Identifier allocated but no enabled semantics; selection refuses.
deprecated	Recognized but not selectable; selection refuses.
disabled	Known but unavailable; selection refuses.

Semantic/evidence separation. None of these semantic templates contains vector_set_sha256, fixture digests, expected-result digests, report identities, publication metadata, or release-envelope identities. Those belong to Volume IV evidence and release layers.

B. Base registry state

The base five-volume semantic publication defines the extension mechanism and authoring requirements while enabling no concrete optional profile or extension. Machine-readable registries are regenerated only after all normative volumes are approved.

Registry	Required base state
profiles	No enabled record.
profile_combinations	No enabled record.
extensions	No enabled optional extension record.
profiled-bundle target	Enabled target with deterministic Stage 3 PROFILE_SELECTION_REQUIRED behavior when no profile is selected.

No synthetic assurance. No "integrity profile" or other redundant profile is created merely to repeat checks already required by the bundle target or to manufacture a positive profiled-bundle vector.

C. Profile authoring checklist

- The profile adds a real public assurance condition rather than a product workflow or duplicate base check.
- Every artifact, extension, check, trust-input form, result contract, claim, limitation, and refusal has one authoritative semantic source.
- Every selected record resolves completely at Stage 3; Stage 10 contains no selection or merge logic.
- The profile does not weaken strict parsing, external trust, prepared-directory scope, artifact_type classification, fail-closed behavior, or the target boundary.
- Required artifacts are discovered only through artifact_type and declare permitted content roles and cardinality.
- Added trust is supplied independently, has an exact canonical identity, and is retained by a closed public result contract.
- The permitted claim is narrower than or equal to the property directly verified; mandatory limitations remain explicit.
- No privacy-screening claim proves universal absence of sensitive information.
- Unknown or unselected extension files receive generic integrity only and never contribute assurance.
- Carrier, multi-signature, post-quantum, or hybrid behavior uses distinct registered identifiers and complete public semantics.
- The profile remains offline-verifiable and does not require a live service or unidentified participant.
- No proprietary decision logic, private evidence, thresholds, prompts, models, orchestration, or architecture is required for base interoperability.
- The semantic record contains no vector-set, fixture, expected-result, report, implementation, or release identity.
- Volume IV evidence covers positive behavior where applicable, every refusal, boundaries, multi-fault precedence, claims, limitations, and opaque-extension handling.
- Every placeholder example is labeled SCHEMATIC - NOT MACHINE-VALID.

End of specification. End of LARRI Profiles and Extensions Specification 0.1.0.