Universal Manifest v0.5 — EXT-T2

Tier-2 Cryptographic-Binding Profile · Preview

v0.5.0. Published as tag v0.5.0 of UniversalManifest/universal-manifest, with the same wire files as release candidate 1 (v0.5.0-rc.1). It is a mechanism release, not a completed implementation or conformance qualification. Immutable versions in UniversalManifest/universal-manifest are normative; main is the working version. Earlier versions were reviewed in the OMA3 Technical Working Group, and publication asserts no OMA3 ratification.

8 November 2026

This document: /spec/v0.5/ext-t2/

Document set: Universal Manifest v0.5.0 · What changed in v0.5.0

Artifacts: JSON Schema · JSON-LD context

History: https://universalmanifest.net/spec/ (version index and changelog)

Author: Grig Bilham — inventor, architect, and author of the Universal Manifest.

Copyright © 2026 SumSet Tech LLC and the Universal Manifest contributors. This specification is available under the Apache License, Version 2.0. It is free and remains free. Earlier published revisions retain their original licence notices.

---

v0.5.0. Published as tag `v0.5.0` of `UniversalManifest/universal-manifest`, with the same wire files as release candidate 1 (`v0.5.0-rc.1`). Wire version `0.5`; namespace `https://universalmanifest.net/ns/v0.5`; reading routes `/spec/v0.5/`. It is a mechanism release, not a completed implementation or conformance qualification. Feature-specific PREVIEW and conditional requirements remain in force.

This is a **companion profile** to the Universal Manifest v0.5 Base specification. It specifies the **Tier-2** cryptographic cross-DID binding profiles, the BBS+ selective-disclosure track that shares the same cryptography, and the post-quantum signature migration path. Read the Base and EXT-T1 first.

Status: All content in this profile is PREVIEW — under active review, built on the editors' recommended defaults, and fully revisable before the v0.5 schema is locked.

EXT-T2: Tier-2 Cryptographic-Binding Profile PREVIEW

Companion to: Base specification, building on EXT-T1. **Registry categories:** `trust`, `signature`.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC 2119] [RFC 8174] when, and only when, they appear in ALL CAPITALS, as shown here.

T2.0. Scope and Relationship to the Base

The Base defines **Tier 2 — Cryptographic binding** (Base §6.4.2) as the assurance level at which cross-DID control is *cryptographically proven via zero-knowledge proof, without revealing private key material*, and reserves the `bindingProof` field on `identity.crossDidBinding` claims (Base §6.4.4) and the `postQuantumSignature` member (Base §6.7, Base §1.6.3). This profile specifies:

- **The Tier-2 ZKP proof profiles** (T2.1) — Profile 2A (BBS+ linked-secret) and Profile 2B (HD-derivation), carried in `bindingProof`.
- **The selective-disclosure dual-track model** (T2.2) — Track A (SD-JWT baseline, which stays in scope of EXT-T1) and Track B (BBS+ derived proofs), which is the privacy face of the same cryptography and is required to claim Tier-2 privacy-preserving binding (T2.2.2).
- **The post-quantum signature profile** (T2.3) — forward-looking quantum-resistant signing and its dual-signature migration path.

These are bundled here for tier coherence: a deployment certifying to Tier-2 high assurance generally needs the ZK proof, the BBS+ selective-disclosure track, and a forward-crypto plan together. Tier-2 evaluators MUST also implement EXT-T1, since Tier 2 is strictly additive over Tier 1.

T2.1. Tier 2 ZKP Proof Profile

This profile provides cryptographic proof that the same entity controls multiple DIDs without revealing private key material or linking secrets. v0.5 provides two proof profiles for Tier 2 cross-DID binding. Both use the `bindingProof` field on `identity.crossDidBinding` claims.

Scope note — same-scheme key material. Both proof profiles below assume the bound keys share a cryptographic scheme: Profile 2A proves equality of a secret committed under BBS+ credentials, and Profile 2B proves derivation of two keys from one seed under a single derivation

scheme. Neither covers **cross-suite** binding — an Ed25519 key (Signature Profile A) and a P-256 key (Signature Profile B, EXT-OPT §O15) held by one subject on different devices. That case is served by **reciprocal cross-signatures**, which work across suites: each key signs a statement naming the other, and each signature is verified under its own key's suite. Reciprocal control is already the `reciprocal-control` holder-binding mode (EXT-T1 §T1.1); it proves control of both keys without a zero-knowledge proof and without either key learning the other's scheme. A cross-suite binding therefore carries no `bindingProof` and is evaluated at Tier 1 as an attester-asserted or reciprocally-proven claim, not at Tier 2 — the tier reflects the proof mechanism, not the trustworthiness of the keys.

Cardinality. Both proof profiles are strictly pairwise: one `bindingProof` proves control over exactly one pair of DIDs. A claim that carries a `bindingProof` MUST therefore list **exactly 2** entries in `boundDids` (Base §6.4.4), and the proof binds to that pair in array order (Profile 2B's public inputs are position-sensitive — T2.1.3). Control of more than two DIDs is expressed as multiple `identity.crossDidBinding` claims, each carrying its own pairwise proof; since every such claim's `boundDids` includes the manifest `subject` per Base §6.4.4, an N-DID assertion is a star of pairwise claims around the subject. An evaluator encountering a `bindingProof` on a claim whose `boundDids` cardinality is not exactly 2 MUST treat the proof as failing verification (recorded `failed` and capped at Tier 0, per T2.1.3); it MUST NOT select a sub-pair to verify. *(Wire note: the v0.5 candidate schema applies `maxItems: 2` conditionally when `bindingProof` is present, alongside the existing `minItems: 2`. Claims without a `bindingProof`, including ceremony-only claims, keep the ≥ 2 cardinality. Structural cardinality does not verify a proof.)*

T2.1.1. Profile 2A: BBS+ Linked-Secret Proof

The holder possesses a secret `s` committed in the credentials of both DIDs. The ZK proof demonstrates equality of committed secrets across credentials without revealing `s`, following the AnonCreds link-secret model adapted for Universal Manifest. The two credentials are carried in (or resolvable from) the claim's `claimProof` entries (Base §6.4.3); a Profile-2A claim without resolvable credentials for both `boundDids` fails verification. Evaluators MUST verify the BBS+ proof of equality against the issuer's BLS12-381 G2 public keys for both credentials; the proof inherently demonstrates that the presenter possesses the holder-committed secret in both credentials.

A Profile 2A `bindingProof` MUST contain:

- `type` — MUST be `"ZkLinkedSecretProof"`.
- `cryptosuite` — MUST be `"bbs-2023"` per the W3C Data Integrity BBS Cryptosuites.
- `proofPurpose` — MUST be `"authentication"`.

- `proofValue` — Base64url-encoded BBS+ derived proof bytes.
- `publicInputs` — An object containing `commitmentA` and `commitmentB`, the BBS+ commitments from each credential.

BBS+ signatures ([BBS+]) are the RECOMMENDED cryptographic suite for privacy-preserving Tier 2 operations: BBS+ derived proofs do not repeat the issuer's signature across presentations, and selective disclosure is native to the signature scheme. Other carried values can still link presentations (Base §7.1).

T2.1.2. Profile 2B: HD Derivation Proof

Both DIDs are derived from a common master seed via hierarchical deterministic key derivation (BIP-32/SLIP-10). The ZK proof demonstrates that a master seed exists such that the two public keys are derivable at declared paths, without revealing the seed.

A Profile 2B `bindingProof` MUST contain:

- `type` — MUST be `"ZkHdDerivationProof"`.
- `proofSystem` — The ZK proof system used (e.g., `"groth16"`). Evaluators MUST support Groth16 [Groth16]; MAY support PLONK or Bulletproofs.
- `circuit` — URI identifying the HD derivation circuit (e.g., `"urn:uuid:circuit-hd-derivation-v1"`).
- `proofValue` — Base64url-encoded proof bytes.
- `publicInputs` — An object containing `publicKeyA`, `publicKeyB`, `derivationPathA`, and `derivationPathB`.

Groth16 proofs require a per-circuit trusted setup ceremony. The ceremony parameters MUST be publicly auditable. Evaluators MUST verify against the trusted pinned circuit/version/key and public-input binding required by EXT-T1 §T1.6.2, including exact key bytes, integrity digest, proof system, input schema/order, and setup/trust evidence. A circuit URI or prover-supplied key does not establish that binding.

T2.1.3. Verification Procedure

When an evaluator encounters an `identity.crossDidBinding` claim carrying a `bindingProof`, it MUST apply the following procedure (the declared `requiredTrustTier` governs what the verified result must satisfy, not whether the proof is verified):

1. Confirm the claim's `boundDids` cardinality is exactly 2 (T2.1), and identify the proof type from `bindingProof.type`.
2. For `ZkLinkedSecretProof` (Profile 2A): verify the BBS+ derived proof against the issuer's BLS12-381 G2 public keys referenced in the claim's credentials, and confirm that the proof demonstrates equality of the committed holder secret. Then **bind the proof to the claimed DIDs**: confirm that each credential against which the proof verifies resolves to a **distinct** member of `boundDids` as its credential subject — together, the two credential subjects MUST equal the `boundDids` pair. A proof whose credential subjects do not match `boundDids` demonstrates only that the presenter controls *some* pair of credentials, not the claimed binding.
3. For `ZkHdDerivationProof` (Profile 2B): select the already trusted, pinned circuit/version/key binding described in T2.1.2, require the declared proof system and circuit to match it, construct the public inputs in its specified schema and order, and verify the ZK proof. Do not fetch or trust a key solely because the untrusted proof supplies a circuit URI; an unknown or unavailable binding cannot grant a higher tier. Then **bind the public inputs to the claimed DIDs**: resolve each bound DID's DID document and confirm that `publicKeyA` is a verification method of `boundDids[0]` and `publicKeyB` is a verification method of `boundDids[1]`, respectively (listed in, or referenced from a verification relationship of, that DID's DID document). A bound DID whose DID document cannot be resolved fails this check. The ZK proof alone demonstrates only that the two *declared* keys derive from one master seed; the DID-document check is what ties those keys to the DIDs the claim asserts are bound — without it, a holder could present a valid proof over two keys it controls while `boundDids` names arbitrary DIDs.
4. Record the result in the receipt. A proof that both verifies and binds elevates the claim to Tier 2. Any failure or mismatch in steps 1–3 — a cardinality violation, a failing proof, a credential subject that is not a distinct member of `boundDids`, or a declared public key that is not a verification method of its respective bound DID — MUST cause the claim to be recorded as `failed` and treated at Tier 0, regardless of any accompanying attestations; evaluators MAY apply stricter local policy (for example, rejecting the manifest). Only an *absent* `bindingProof` falls back to attester-asserted Tier 1 evaluation.

(Worked example: Tier 2 cross-DID binding with linked-secret proof — see Cookbook.)

Preview — built on default: This makes **Groth16** the mandatory-to-implement proof system for Profile 2B (smallest proofs, fastest verification), with PLONK permitted as an optional alternative for deployments that prefer a universal setup. Input is requested on whether the trusted-setup cost of Groth16 justifies making PLONK the baseline instead. Flag to revise.

T2.2. Selective Disclosure: Dual-Track Model

Universal Manifest supports selective disclosure at the manifest level through holder-controlled projection (Base §3.1.3), encrypted facets (Base §2.3), and credential-level disclosure inside claims. For credential-level selective disclosure, v0.5 defines two interoperable tracks so deployments can choose between web-native simplicity and fewer repeated proof values:

- **Track A — SD-JWT (baseline, default).** Selective disclosure of JWT/JSON credential fields using SD-JWT [RFC9901], with key binding via KB-JWT (EXT-T1 §T1.1). This is the mandatory-to-implement baseline and the default for web and JWT environments, where library support is broad and integration is simplest. *(Track A's mechanics live in EXT-T1; it is named here for completeness of the dual-track model.)*
- **Track B — BBS+ derived proofs (privacy-critical).** Selective disclosure using BBS+ derived proofs [VC-DI-BBS], where the proof bytes of one presentation do not link to other presentations of the same credential. Other carried values can still link them (Base §7.1). Track B is RECOMMENDED for privacy-critical deployments and is **required to claim Tier 2 privacy-preserving binding** (T2.2.2).

Manifest-level selective presentation composes with both tracks via facet projection, claim filtering, and pointer elision: the holder controls which facets, claims, and pointers appear in a given manifest instance, and Track A or Track B controls disclosure *within* a presented credential. Spatial-computing deployments, where the same subject may be observed across many venues, SHOULD use Track B to remove the repeated issuer signature. Track B alone does not prevent cross-platform correlation. Deployments otherwise default to Track A.

T2.2.1. Privacy Considerations for Track B

The **privacy–binding tension** is fundamental: stronger binding enables correlation, while stronger privacy prevents binding verification. The specification provides mechanisms for both and requires evaluators to define their trust tier based on their threat model. Evaluators MUST NOT require cross-DID binding unless their use case demands Sybil resistance or trust transitivity; subjects SHOULD use the minimum binding tier that satisfies their evaluators' requirements. **BBS+ pseudonym scope:** if `pseudonymScope` is reused across verifiers, cross-verifier correlation becomes possible; issuers SHOULD generate unique scope URIs per verifier relationship, and evaluators MUST NOT use a shared pseudonym scope for multiple independent verifier relationships (EXT-T1 §T1.6.1).

Preview: The dual-track model is built on the editors' default (SD-JWT baseline; BBS+ for privacy-critical and Tier 2). Input is requested on whether Track B should be mandatory for any class of

spatial-computing deployment.

T2.2.2. Claiming Tier-2 Privacy-Preserving Binding PREVIEW

Tier-2 privacy-preserving binding — the property Track B is REQUIRED for (T2.0, T2.2) — is a claimable conformance label, not a new mechanism. Its candidate registered-profile identifier is `um:profile:trust:tier2-privacy-preserving-binding` (a `trust`-category profile per EXT-OPT §O6; the identifier is provisional until registration). Following the thin conformance-profile-label precedent of EXT-T1 §T1.3.3, the label names a bundle of existing requirements and defines no new ceremony, cryptography, or member. A deployment conforms to the label when, together:

1. cross-DID binding is verified at Tier 2 via a T2.1 proof profile, including the `boundDids` binding steps of T2.1.3;
2. credential-level selective disclosure uses Track B (BBS+ derived proofs, T2.2) for the credentials involved — Track A alone MUST NOT be the basis of this claim; and
3. the pseudonym-scope discipline of T2.2.1 / EXT-T1 §T1.6.1 is applied.

The label is claimed in the deployment's conformance claim (EXT-OPT §O11.1, item 2 — implemented modules and claimed PREVIEW features); it adds no manifest wire member.

T2.3. Post-Quantum Signatures

This profile defines a post-quantum signature profile providing quantum-resistant signing as a supplement to the baseline Ed25519 profile (Signature Profile A, Base §1.6).

T2.3.1. Candidate Algorithms

The following NIST-standardized post-quantum signature algorithms are candidates:

- **ML-DSA (CRYSTALS-Dilithium)** — NIST FIPS 204 [FIPS204]. Lattice-based, with a 3309-byte signature and 1952-byte public key for ML-DSA-65 (FIPS 204 Table 2), and fast verification. The leading candidate for general-purpose post-quantum signatures.
- **SLH-DSA (SPHINCS+)** — NIST FIPS 205 [FIPS205]. Hash-based, larger signatures (~8–30 KB), no lattice assumptions (conservative choice). Suitable as a fallback if lattice-based schemes are broken.

- **FN-DSA (FALCON)** — NIST (pending standardization). Lattice-based, compact signatures (~666 bytes for FALCON-512), more complex implementation. Suitable for constrained devices where signature size is critical.

For size accounting, the following are raw decoded lengths, before container encoding. ML-DSA values come from FIPS 204 Table 2; Ed25519 values come from RFC 8032 §7.

Algorithm	Raw public key (bytes)	Signature (bytes)	Unpadded base64url public key (characters)	Unpadded base64url signature (characters)
ML-DSA-65	1952	3309	2603	4412
ML-DSA-87	2592	4627	3456	6170
Ed25519	32	64	43	86

These raw-key lengths do not specify a DER/SPKI carrier. The table provides size accounting and does not register ML-DSA-87 or settle the post-quantum carrier and verification parameters.

T2.3.2. Profile Naming

The post-quantum profile follows the signature-profile naming convention of Base §1.6. The recommended baseline post-quantum profile is **JCS + ML-DSA-65**. “Signature Profile B” denotes JCS + ES256 under EXT-OPT O15; it does not denote this post-quantum profile. JCS + ML-DSA-65 shares Profile A's canonicalization: in a `postQuantumSignature` object following this profile, `canonicalization` MUST be `"JCS-RFC8785"` and `algorithm` MUST be `"ML-DSA-65"`; the `algorithm` + `canonicalization` pair identifies the profile, exactly as Base §1.6.5 requires for `signature`, and an evaluator that processes `postQuantumSignature` at all MUST apply the same unsupported-pair rejection discipline to it. This naming correction does not register a new suite, adopt a profile identifier, or settle the candidate carrier and verification parameters.

T2.3.3. Dual-Signature Migration Path

The transition from Ed25519 to post-quantum signatures uses a dual-signature period. During this period, manifests carry both an Ed25519 signature (in the existing `signature` field) and a post-quantum signature (in a new `postQuantumSignature` field).

- Evaluators that support the post-quantum profile MUST verify both signatures, and MUST reject the manifest if **either** verification fails. Dual verification is conjunctive: one valid

signature never compensates for the other's failure. (Accepting on one-of-two would let a forgery under a broken algorithm ride on the other signature's validity — defeating the purpose of carrying both during exactly the period this mechanism exists to protect.)

- Evaluators that do not yet support the post-quantum profile MUST verify only the Ed25519 signature and MUST ignore the `postQuantumSignature` field.
- The dual-signature period begins when the post-quantum profile is published and ends when the maintainers declare Ed25519-only manifests non-conformant at a major-version boundary.

The `postQuantumSignature` field follows the same schema as `signature` (Base §1.6) with the `algorithm` field set to the post-quantum algorithm identifier (e.g., `"ML-DSA-65"`). Both proofs are computed over the same signing input: the canonical bytes of Base §1.6.3 with the `signature`, `postQuantumSignature`, and (if present) `presentationProof` properties removed, so that neither signature's bytes feed the other's input.

(Worked example: dual-signature manifest — see Cookbook.)

Open questions: Whether the dual-signature mechanism should use a separate top-level field or an array of signature objects is open (separate field is simpler for backward compatibility; an array is more extensible). ML-DSA-65 signatures (3309 bytes) are significantly larger than Ed25519 (64 bytes); whether this size increase matters for constrained-device deployments is also open.

References

Cited in this profile (full descriptors in the published artifact):

- **[BBS+]** BBS+ Signature Scheme / BBS Signatures.
- **[VC-DI-BBS]** Verifiable Credentials Data Integrity BBS Cryptosuites (`bbs-2023`).
- **[Groth16]** Groth, *On the Size of Pairing-Based Non-interactive Arguments*.
- **[RFC9901]** SD-JWT (Selective Disclosure for JWTs).
- **[FIPS204]** NIST, *Module-Lattice-Based Digital Signature Standard*, FIPS 204, 13 August 2024, Table 2. **[FIPS205]** SLH-DSA (SPHINCS+). FN-DSA (FALCON), NIST.

✧ ✧ ✧

Companion to the Base specification. For the Tier-1 layer this builds on, see EXT-T1; for Tier-3 ceremonies, see EXT-T3.