# SADAR Replication and Manifest Provenance

## SADAR Replication and Manifest Provenance

Replication and Manifest Provenance

SADAR Documentation — Federation

*Draft — May 2026*

## Purpose

This document is the orientation page for SADAR's replication model: how manifests retain verifiable provenance when replicated across federated registries, the structure of the home-registry binding artifact, the requirements for byte-identical replication, and the verification procedure consumers follow when retrieving content from any point in the federation. It also covers the push channel that registries use for replication propagation, revocation notification, and federation lifecycle messaging.

The model satisfies two simultaneous properties: manifests must be byte-identical wherever they appear (so the publisher's signature verifies regardless of which registry serves the content), and registries must be able to attest “this manifest was published through me” in a way that is independently verifiable wherever the manifest travels. The mechanism described here achieves both without introducing any registry-side modification of publisher content.

The normative basis lives across multiple sources. Scope §5.1.6 covers Registry Protocol (descriptor manifest validation and immutability). Scope §5.1.11 establishes registry topology and the Directory. Scope §5.1.12 establishes Registry Isolation. The replication model described here builds on those foundations, applying the same manifest, signing, and lifecycle disciplines to the cryptographic attribution of replicated content.

Audience: implementers building SADAR-conformant registries that participate in replication; architects reasoning about cross-registry trust properties; security teams evaluating the integrity of replicated content. The document is descriptive, with normative requirements summarized at the end and open items enumerated thereafter.

## What Replication Covers

Replication in SADAR is the operational mechanism by which content published through one registry becomes available through other registries with which the home registry has established a federation relationship. The replication model addresses three concerns:

- **Cryptographic attribution.** Every replicated manifest carries a cryptographic record of the registry through which it was published — the home registry. This record is verifiable at any consumer regardless of which peer registry served the content.
- **Byte-identical distribution.** Manifests look identical wherever they appear. The publisher's signature is invariant across replicas because the manifest is invariant. Peer registries do not modify, re-sign, or annotate the content they serve.
- **Lifecycle propagation.** Changes to replicated content (new versions, deprecation, revocation, federation status) propagate to peers through a mandatory push channel that uses the standard SADAR authentication primitives.

This page covers replication mechanics at orientation depth. Federation establishment (how registries enter into bilateral agreements in the first place) and federation policy (how registry admins govern what content flows through which federations) live in Federation Establishment and Policy. The governance regime that authorizes registries to participate in public federation lives in Governance and Conformance.

# Architectural Model

## Registries as Registered Entries

Within the SADAR model, a registry is itself a registered entry. Each registry MUST be associated with an `owning_entity`, MUST have its own publisher-signed manifest, MUST publish its verification key (typically through a JWKS endpoint), and MUST anchor its verification key in a declared trust anchor consistent with the trust-anchor neutrality model that applies to every entry. The registry's manifest follows the same publication and signing flow as any other manifest.

This recursion is intentional and follows the recursive architectural pattern established in Governance and Conformance. A registry is not a privileged third-party authority sitting outside the manifest-and-signature model; it is a participant within that model, subject to the same cryptographic disciplines as any other entry. The novelty of a registry is the role it plays (publication, discovery, federation) not the identity model that applies to it.

## The Home Registry

Every published agent, tool, business function, business process, or registry descriptor has exactly one **home registry** — the registry through which the publisher first published the manifest. The home registry is the cryptographic point of attribution for the manifest's publication provenance and is the registry that issues the home-registry binding (described below).

A publisher selects a home registry as part of the publication act. Once selected, the home registry is durable for that manifest version: subsequent replication to peer registries does not change the home registry; replicas serve the manifest with the original home-registry binding intact. A publisher wishing to publish through a different home registry republishes (creating a new manifest version, with a new home-registry binding); the new publication is treated as a separate artifact for replication purposes.

The home registry role for content flowing into public federation is restricted. A non-authorized registry (one not listed in the Directory of Authorized Registries) MAY issue home-registry bindings for its own internal content, and MAY consume content from authorized registries through bilateral federation, but MUST NOT serve as the home registry for content discoverable to public-federation peers. The institutional-trust verification at the consumer side enforces this structurally: home-registry bindings issued by non-authorized registries fail the institutional-trust check at step 5 of the verification chain (described below), so consumers reject the content even though the cryptographic chain itself verifies.

## Manifests Are Signed Only by Their Publishers

Manifests are signed exclusively by their publishers, never by registries. A publisher's manifest is byte-identical wherever it appears: at the home registry, at any peer registry that has replicated the content, or in any cache or out-of-band copy. The publisher's signature on the manifest verifies against the publisher's verification key, regardless of where the manifest is retrieved.

Registries do not modify, re-sign, overlay, or annotate publisher content. The only cryptographic act a registry performs in connection with another entry's manifest is the issuance of a home-registry binding (for the home registry only) — and that binding is structurally separate from the manifest itself. Peer registries serving replicated content do not sign anything on behalf of that content; their role is byte-faithful distribution, verifiable through hash comparison rather than additional signatures.

## Bilateral, Non-Transitive Federation

Federation in SADAR is bilateral and non-transitive. If Registry A federates with Registry B, and Registry B federates with Registry C, this does not extend Registry A's content to Registry C. Each replication relationship is a separate bilateral agreement between two specific registries; replication rights extend only across that one hop.

A registry that wishes to serve content from a non-adjacent registry MUST establish its own bilateral federation with that registry directly. A peer registry MUST NOT serve content whose home-registry binding is signed by a registry with which the peer has no current direct federation

agreement, even if the binding itself is cryptographically valid. Equivalently: a peer MUST NOT re-replicate content received from one federation to peers in a different federation. Each registry can only expose or forward entries for which it is the home, plus entries it has replicated from a directly-federated home registry under that bilateral agreement.

Non-transitivity has several rationales. It matches real-world organizational trust (commercial relationships do not compose transitively). It preserves the home registry's control over distribution (the home registry chose specific peers based on specific evaluation). It contains blast radius (a compromised peer cannot extend the impact of its compromise to additional registries). And it clarifies the semantic model (one home registry per manifest, one bilateral agreement per replication link, no implicit trust extensions).

# The Home-Registry Binding

## Structural Form

The home-registry binding is a **separate signed artifact** attached to the manifest, not a field within the manifest. It takes the form of a JSON Web Signature (JWS) detached signature envelope wrapping a binding object. The binding object references the manifest by cryptographic hash and is signed by the home registry's signing key.

Structurally, replicated content consists of two artifacts traveling together: the publisher-signed manifest, unchanged from publication; and the home-registry binding, which provides the cryptographic attribution to the home registry. The two are linked by hash reference; they are not concatenated, embedded, or merged.

### Binding Object Schema

The binding object is a JSON document with the following fields:

```
{   "manifest_hash": "<base64url(SHA-256(canonicalized manifest))>",
"manifest_hash_alg": "sha-256",   "home_registry_id": "<URN of the home registry>",
"home_registry_kid": "<key identifier of the signing key>",   "issued_at": "<RFC 3339
timestamp>",   "valid_until": "<RFC 3339 timestamp>",   "binding_version": "1" }
```

The `manifest_hash` is computed over the canonicalized form of the publisher-signed manifest as a single octet string, including the publisher's signature. This binds the home-registry attestation to the exact bytes of the manifest as published, making any subsequent modification of the manifest detectable.

The `home_registry_id` is the URN identifying the home registry as a registered entry. The `home_registry_kid` identifies the specific key used to sign the binding, allowing the consumer to select the correct verification key from the home registry's manifest.

The `valid_until` field carries the binding's TTL, supporting the universal lifecycle pattern. A binding past its `valid_until` is non-current and triggers re-resolution at the consumer in the same manner as any other expired SADAR artifact.

### Signature Format

The binding object is signed using a JWS detached signature in compact serialization. The signing algorithm is selected from the home registry's manifest-declared supported algorithms and the consumer's accepted-algorithm policy, with at minimum the bootstrap algorithm set (specified in the SADAR Conformance Specification) being mandatory for all conformant implementations.

ES256 (ECDSA P-256 with SHA-256) is the default algorithm consistent with Scope §5.1.8.1. EdDSA (Ed25519) is recommended for new deployments; RS256 is supported for legacy interoperability. The home registry's manifest declares which algorithm(s) it uses for binding signatures, and a consumer verifying the binding selects the appropriate algorithm based on the JWS `alg` header and the home registry's declared capabilities.

### Manifest Canonicalization

Because the binding's `manifest_hash` must be reproducible by any consumer verifying the binding, the manifest must be hashed in a canonical form. SADAR specifies JSON Canonicalization Scheme (JCS, RFC 8785) for this purpose. The publisher-signed manifest is reduced to its JCS canonical form, encoded as UTF-8 octets, and hashed with SHA-256.

This procedure produces a deterministic hash over the manifest content regardless of whitespace, key ordering, or transport-layer formatting differences. Consumers replicate the procedure to verify that the binding refers to the manifest they hold.

# Replication Mechanics

## What Is Replicated

When a peer registry replicates content from a home registry, the replicated set consists of three artifacts traveling together:

- The publisher-signed manifest, served byte-identical to the form in which it was published.
- The home-registry binding, served byte-identical to the form in which the home registry issued it.
- The home registry's own manifest, which contains the home registry's verification key and is required for binding verification.

All three artifacts must be available for downstream verification to succeed. A peer registry that replicates manifests without also serving the home registry's own manifest produces content whose home-registry attribution cannot be verified at the peer — which is itself a useful integrity-violation indicator. The structural requirement is a feature: it ensures the verification chain is closed wherever the content travels.

## Push and Pull Replication

SADAR registries support two replication models, both of which MAY be used in combination:

| Model | Behavior | Operational role |
|---|---|---|
| Push | The home registry actively notifies federated peers of new or changed content through their declared push endpoints. Push delivers operational responsiveness: peers learn of changes at the moment of change. | Mandatory for conformance. A registry that does not implement push MUST NOT be certified, and consequently MUST NOT be authorized for participation in public federation. |
| Pull | Peer registries periodically poll the home registry for changes since the last pull. Pull is a fallback mechanism for environments where push delivery is unreliable. | Permitted but not a substitute for push. Pull-only operation degrades revocation and deprecation propagation to TTL-bounded latency, which is unacceptable for security-critical messages. |

Push endpoints are themselves SADAR-authenticated channels. They accept mTLS-protected connections and authenticate incoming push messages using the same SADAR authentication primitives that govern every other SADAR channel: signature verification on push payloads against the issuing registry's manifest verification key, federation-status check at the receiver before processing, and rejection of messages from peers with which no current federation agreement exists. The push channel is not a special-purpose mechanism; it is the universal SADAR authentication model applied to this particular role.

## What Peer Registries Do Not Do

A peer registry serving replicated content does not:

- Sign the replicated manifest with its own key.
- Modify, annotate, or overlay the home-registry binding.
- Issue a new binding to replace the home-registry's binding.
- Represent itself as the publisher or as the home registry of replicated content.
- Re-replicate content to its own peers (federation is non-transitive).

A peer registry's role with respect to replicated content is to serve the bytes faithfully, with all original signatures intact, only to consumers and counterparties that the peer has independently authorized through its own federation policy. The peer's identity is verifiable through its own manifest, but the peer's identity is not bound to the replicated content; provenance attribution remains exclusively the home registry's.

### The Verification Chain at a Peer

A consumer retrieving an entry's manifest from a peer registry performs the following verification:

1. **Verify the publisher's signature on the manifest.** The consumer obtains the publisher's verification key from the publisher's manifest (typically through a JWKS endpoint declared therein) and verifies the manifest's publisher signature against it. The publisher's verification key is anchored to a trust anchor declared in the publisher's manifest.
2. **Compute the manifest hash.** The consumer canonicalizes the manifest (JCS) and computes the SHA-256 hash of the canonical UTF-8 octets.
3. **Verify the home-registry binding.** The consumer compares the computed hash to the `manifest_hash` field in the binding object, then verifies the binding's JWS signature against the home registry's verification key. The home registry's verification key is obtained from the home registry's manifest, identified by the `home_registry_id` and `home_registry_kid` fields. The `valid_until` field is checked to confirm the binding is current.
4. **Verify the home registry's manifest.** The consumer verifies the home registry's manifest signature against the registry operator's verification key, anchored according to the trust-anchor model declared in the home registry's manifest.
5. **Confirm institutional trust through current authorization.** The consumer confirms that the home registry is currently authorized — listed in the Directory of Authorized Registries (or in the relevant private RoR for content scoped to private federation). Certification alone is not sufficient; current authorization is required. A home registry that is certified but not authorized, or that has been deauthorized, fails this step. This confirms the institutional-trust path described in Governance and Conformance.

All verification steps MUST succeed for the consumer to accept the manifest as authentically published through the named home registry and as institutionally trustworthy. Failure of any step indicates either a tampered artifact, an invalid signature, an expired binding, a missing replicated component, or a federation policy violation.

## Revocation

Revocation in the replication context comes in two distinct cases. The first applies when the entry itself is no longer valid; the second applies when the entry is valid but a specific peer registry should not have replicated or served it. The two are governed differently and have different propagation semantics.

## Case A: Entry Revocation

Case A applies when the entry itself is no longer valid: a publisher's signing key has been compromised, an agent has been withdrawn from service, a manifest version has been superseded by a corrected version. The publisher (or, for delegated revocation, the home registry on the publisher's behalf) signals the revocation through the standard SADAR lifecycle mechanism: the entry is marked deprecated or revoked in the home registry's records, the change is propagated through push to federated peers, and consumers re-resolving the artifact pick up the change at next TTL expiry or sooner.

Case A revocation is a property of the entry itself. It applies regardless of where the entry's manifest is being served; any registry replicating the manifest receives the deprecation/revocation through push and updates its served content accordingly.

## Case B: Registry-Replica Revocation

Case B applies when the entry itself remains valid, but a specific registry should not have replicated or served it. This case arises in scenarios such as a peer registry that misrepresented its federation agreement, a federation agreement that has been terminated where the peer has not yet ceased to serve previously-replicated content, or a peer that has been deauthorized by the Authorizing Body but has not yet ceased operations.

The home registry is the accountable party in Case B. The home registry owns the relationship with the publisher (the entry remains valid; the publisher's authority is not in question), is likely the source of any error in establishing the federation, and is the partner in the federation in question. In the case of a dispute, the plaintiff can ask the Authorizing Body to deauthorize the registry from federations; the Authorizing Body has an accountability to investigate. Authorizing-Body-issued deauthorization propagates through the same standard lifecycle and push mechanism.

Case B revocation propagates through the push channel — push is one of the standard replication models, so each registry exposes a push endpoint as a conformance requirement. The home registry signs and pushes the revocation to all peers in its federation, naming the peer being deauthorized and (optionally) the specific content for which the deauthorization applies. Receiving registries update their internal records: the deauthorized peer is no longer

treated as a federated source for the named content (or, if the deauthorization is general, for any content from this home registry).

### Service-Side Provenance Filtering

Case B revocation is operational rather than purely cryptographic. The cryptographic artifacts a consumer receives from the deauthorized peer remain valid: the publisher's signature on the manifest is intact, the home-registry binding is intact, the home registry's manifest signature is intact. A consumer applying only artifact-level verification cannot detect that this peer should not have been the source of this content.

To address this, the SADAR Context Token (SCT) carries the registry ID from which the discovery was made. The receiving service can always decline incoming calls — that is how it proactively revokes access to a requester. The discovery-registry attribution in the SCT becomes another filter for that activity: a service MAY refuse calls discovered through a deauthorized registry independently of the artifact-level verification, providing enforcement that does not depend on the deauthorized peer's compliance.

## Properties of the Model

Several architectural properties follow from the structure described above.

| Property | What it means |
|---|---|
| Byte-identical replication | Manifests are byte-identical wherever they appear. A consumer fetching a manifest from a peer registry receives the same bytes a consumer fetching from the home registry would receive. The publisher's signature is invariant across replicas because the manifest is invariant. |
| No editorial authority for registries | A registry — home or peer — cannot alter what a publisher signed. It cannot strip another registry's home-registry binding to claim content as its own. It cannot pretend to have published content it did not. Editorial authority is bounded by what each party can sign, and what each party can sign is bounded by the keys they hold. |
| Bounded blast radius for compromised replicas | A compromised peer registry can refuse to serve content, serve stale content, or attempt to serve tampered content. Tampering is detectable through hash and signature verification. Impersonation is detectable because peers cannot issue valid bindings or publisher signatures for content they do not own. The most a compromised peer can do silently is censor. |
| Federation without centralized authority | Each registry is sovereign over its own identity (the operator's keypair), its own manifest, and the home-registry bindings it has issued. The Authorizing Body's role is governance, not centralization of operational authority. |

| Property | What it means |
|---|---|
| Replication includes the home registry's manifest | Verification of the home-registry binding requires the home registry's verification key, which lives in its manifest. Replication of any content structurally requires replication of the home registry's manifest alongside it. |
| Authorization-gated home-registry role | A non-authorized registry cannot serve as the home registry for public-federation content because the consumer-side institutional-trust check verifies current authorization (Directory listing). The architectural enforcement is automatic; no separate gating mechanism is required at the protocol level. |

## Boundaries — What's Not Here

This page covers replication mechanics and the home-registry binding at orientation depth. Several adjacent topics live in dedicated documents:

| Topic | Where it lives |
|---|---|
| Federation establishment and policy | How registries discover federation candidates, evaluate compatibility, establish bilateral agreements, and administer per-registry policy through the registry admin console. See Federation Establishment and Policy. |
| Governance and certification | The Steward, the Authorizing Body, the conformance/certification/authorization ladder, the two-path trust model, and the universal lifecycle. See Governance and Conformance. |
| Registry architecture | The registry itself — what it does, what it doesn't do, the six registry types, the three foundational principles. See Registry Overview. |
| RoR architecture | The federation-layer component that resolves cross-registry discovery, the canonical Directory of Authorized Registries, and registry descriptors. See Registry of Registries. |
| Manifest schema | What manifests contain. This document treats the manifest as the artifact whose hash the binding references and whose publisher signature is the primary authentication of authorship. See 8. NFR Schema and Registry Manifests. |
| SCT and authorization context propagation | The structure, claims, and chain semantics of the SADAR Context Token, which carries the discovery-registry attribution used for service-side provenance filtering. See SADAR Context Token and 9. SCT Operations. |
| Conformance criteria | The bootstrap algorithm set, the mandatory feature boundary, and the certification evaluation criteria. See SADAR Conformance Specification. |

## Normative Requirements Summary

The following requirements use the terminology of **RFC 2119** / RFC 8174 as it will appear in the normative specification revision.

## Publisher Requirements

6. **R-PUB-1.** Publishers MUST sign their manifests with their own durable signing key.
7. **R-PUB-2.** Publishers MUST select exactly one home registry per published manifest at the time of publication.
8. **R-PUB-3.** Publishers MUST NOT sign content as another publisher.

## Home Registry Requirements

9. **R-HR-1.** A home registry MUST issue a home-registry binding for each manifest published through it.
10. **R-HR-2.** The home-registry binding MUST be a separate signed artifact, structurally distinct from the manifest itself.
11. **R-HR-3.** The home-registry binding MUST reference the manifest by cryptographic hash computed over the manifest's JCS-canonicalized form using SHA-256 (or another mandatory bootstrap hash algorithm).
12. **R-HR-4.** The home-registry binding MUST carry a `valid_until` field declaring its TTL.
13. **R-HR-5.** The home registry MUST publish its own manifest containing the verification key(s) used to sign home-registry bindings, identified by `kid`.
14. **R-HR-6.** The home registry MUST NOT modify, re-sign, or overlay publisher content.
15. **R-HR-7.** The home registry MUST be associated with an `owning_entity`, MUST have its own manifest, and MUST anchor its verification key in a declared trust anchor.
16. **R-HR-8.** Only an authorized registry (one currently listed in the Directory of Authorized Registries) MAY serve as the home registry for content flowing into public federation. A non-authorized registry MAY issue home-registry bindings for its own internal content, but those bindings fail the institutional-trust verification at consumers retrieving them as public-federation content.
17. **R-HR-9.** The home registry MUST issue Case B (registry-replica) revocations through the push channel when peer registries are found to be serving content without authorization.

## Peer Registry (Replicating) Requirements

18. **R-PR-1.** A peer registry serving replicated content MUST serve the publisher-signed manifest byte-identically to the form in which it was published.

19. **R-PR-2.** A peer registry serving replicated content MUST serve the home-registry binding byte-identically to the form in which the home registry issued it.
20. **R-PR-3.** A peer registry serving replicated content MUST also serve the home registry's manifest, or provide a verifiable reference by which consumers can retrieve it.
21. **R-PR-4.** A peer registry MUST NOT sign, modify, annotate, overlay, or replace the home-registry binding for replicated content.
22. **R-PR-5.** A peer registry MUST NOT represent itself as the publisher or the home registry of replicated content.
23. **R-PR-6.** A peer registry MUST NOT re-replicate content to its own peers; federation is non-transitive. A peer can only expose or forward entries for which it is the home registry, plus entries replicated from a registry with which it has a current direct federation agreement.
24. **R-PR-7.** A peer registry MUST NOT serve content whose home-registry binding is signed by a registry with which the peer has no current direct federation agreement.
25. **R-PR-8.** A peer registry MUST expose a push endpoint as required for conformance.
26. **R-PR-9.** A non-authorized registry MAY consume content from authorized registries through bilateral federation, subject to the authorized registry's federation policy. A non-authorized registry MUST NOT serve replicated content as if it were a home registry for public-federation purposes.

## Push Channel Requirements

27. **R-PUSH-1.** Conformant registries MUST expose a push endpoint declared in their manifest.
28. **R-PUSH-2.** Push channels MUST use mutual TLS at the transport layer.
29. **R-PUSH-3.** Push messages MUST be signed by the issuing registry, with signatures verifiable against the issuing registry's manifest verification key.
30. **R-PUSH-4.** Push receivers MUST verify message signatures, sender federation status, and message type appropriateness before processing.
31. **R-PUSH-5.** Push receivers MUST reject messages from senders with which no current federation agreement exists.

## Consumer (Verifier) Requirements

32. **R-CV-1.** A consumer verifying replicated content MUST verify the publisher's signature on the manifest using the publisher's verification key.
33. **R-CV-2.** A consumer MUST verify the home-registry binding by computing the JCS-canonicalized SHA-256 hash of the manifest, comparing it to the binding's

`manifest_hash`, checking the binding's `valid_until`, and verifying the binding's JWS signature against the home registry's verification key.

34. **R-CV-3.** A consumer MUST verify the home registry's manifest signature against the registry operator's verification key.
35. **R-CV-4.** A consumer MUST confirm institutional trust by verifying that the home registry is currently authorized (listed in the Directory of Authorized Registries for public-federation content). Certification alone is not sufficient; current authorization is required.
36. **R-CV-5.** A consumer SHOULD reject manifests for which any verification step fails.
37. **R-CV-6.** A consumer MAY cache verified artifacts in accordance with their TTL; cache expiry results in re-resolution and re-verification.

### SCT Discovery Provenance Requirements

38. **R-SCT-1.** SCTs MUST include a discovery-registry attribution identifying the registry through which the requester discovered the service.
39. **R-SCT-2.** The discovery-registry attribution MUST be included in the SCT's signature scope.
40. **R-SCT-3.** Receiving services MAY apply policy on the discovery-registry attribution, including declining calls discovered through deauthorized registries.

## Open Items

This document has no open items. The architectural model is established; specific schema details, algorithm parameters, and operational policies that compose with this model are governed by their respective companion specifications (the SADAR Conformance Specification for the bootstrap algorithm set; 8. NFR Schema for manifest schema details; the SADAR Context Token specification for SCT structure including the discovery-registry attribution).

## Where to Learn More

Governance and Conformance — the Steward, the Authorizing Body, the conformance/certification/authorization ladder, the two-path trust model, and the universal lifecycle that this document depends on.

Federation Establishment and Policy — how registries enter into the bilateral federation agreements that gate replication, including the registry-admin policy layer.

Registry Overview — the registry's operational scope, the six registry types, and the three operational modes including the federation-eligibility distinction.

Registry of Registries — the federation-layer component that resolves cross-registry discovery and the canonical Directory of Authorized Registries.

SADAR Context Token — the chain-of-custody token that carries the discovery-registry attribution used for service-side provenance filtering.

8. NFR Schema — the canonical source for manifest structure, NFR vocabulary, the bilateral match algorithm, and registry-side validation requirements.

9. SCT Operations — the abstract operations any conformant SCT implementation must support, including chain-integrity verification.

2. Scope §5.1.6 — Registry Protocol: registration, validation, immutability, error responses.

2. Scope §5.1.11 — Registry Topology, Federation, and Directory.

2. Scope §5.1.12 — Registry Isolation: the normative constraints establishing the registry as a discovery-time component only.

SADAR Conformance Specification — the bootstrap algorithm set, mandatory features, and certification evaluation criteria.