

Network Working Group  
Internet-Draft  
Intended status: Informational  
Expires: 17 September 2026

B.A. Fisher  
DPA R&D  
16 March 2026

The Universal Zero-Port Interconnect Framework (UZPIF): An Identity-  
Centric Architecture for Post-Port Networking  
draft-dpa-uzpif-framework-01

## Abstract

The Universal Zero-Port Interconnect Framework (UZPIF) describes a post-port networking model in which communication is established via outbound, identity-bound sessions to Rendezvous Nodes (RNs). By removing publicly reachable listening ports at endpoints, UZPIF changes exposure assumptions and can reduce unsolicited ingress and Internet-wide scanning pressure, but it does not by itself guarantee privacy, decentralisation, or availability.

This document outlines architectural motivation, a high-level security model, operational and economic considerations, a Pantheon trust and policy plane baseline, and an incremental migration approach. It is part of an experimental, research-oriented Independent Stream suite and defines the current normative baseline for trust objects, validation rules, and security semantics within the suite framework. Hard interoperability is expected for shared object semantics and validation rules. Full wire-level, clustering, and proof-family interoperability is not claimed everywhere yet; the remaining details are intentionally profile-defined or deferred in companion drafts or profiles. UZPIF is intended to be read alongside companion work describing the Universal Zero-Port Transport Protocol (UZP; [UZP]) and TLS-DPA ([TLS-DPA]).

## Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 17 September 2026.

## Copyright Notice

Copyright (c) 2026 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

## Table of Contents

1. Scope and Status . . . . .	3
2. Executive Summary . . . . .	4
3. Introduction . . . . .	5
3.1. Motivation and Deployment Context . . . . .	6
3.2. Relationship to VPN-Based Approaches . . . . .	6
4. Terminology . . . . .	6
5. Common Signed Artefact Envelope . . . . .	7
5.1. Canonical Serialisation and Signature Coverage . . . . .	9
5.2. Envelope Members and Extensions . . . . .	10
5.3. Suite-wide object_type Registry . . . . .	12
5.4. Signature Algorithm Identifiers . . . . .	14
5.5. Signature Entries, Ordering, and Detached Form . . . . .	14
5.6. Validation Rules . . . . .	15
6. Core Architecture Overview (Informative) . . . . .	17
6.1. High-Level Flow: EP to RN to EP (Informative) . . . . .	17
6.2. Pantheon Grant Verification (Informative) . . . . .	17
6.3. Flow Stitching by the RN (Informative) . . . . .	18
6.4. Multi-RN Stitching for High-Assurance Tenants (Informative) . . . . .	18
7. Legacy Hardware Integration via HIL . . . . .	19
7.1. HIL Requirements . . . . .	19
7.2. Local Trust Boundary of HIL-Mediated Systems . . . . .	20
7.3. HIL Software and Supply-Chain Integrity . . . . .	21
7.4. Legacy Compatibility Classes . . . . .	21
8. Decentralisation and Authority Model . . . . .	22
8.1. Bootstrap Model Clarification . . . . .	22
8.2. Bootstrap, Recovery, and Re-Enrolment . . . . .	22
8.2.1. Bootstrap Trust-Transition Artefacts . . . . .	23
8.2.2. Baseline Trust-Transition Flow . . . . .	26
8.2.3. Initial Trust Establishment . . . . .	27
8.3. Key Custody and Signing-Role Separation . . . . .	28
8.4. Client Sovereignty Model . . . . .	29
8.5. RN Controller Properties . . . . .	29

8.5.1.	RN Non-Sovereignty and Compromise Containment . . . . .	30
8.5.2.	Multi-RN Consistency and Failover Semantics . . . . .	31
8.6.	RN Transparency Log Format . . . . .	35
8.7.	Proof of RN Misbehaviour . . . . .	36
9.	Explicit Non-Goals . . . . .	36
10.	Pantheon: Identity and Governance Model . . . . .	37
10.1.	Identity Model . . . . .	37
10.2.	Certificate Format . . . . .	38
10.3.	Grant Structure . . . . .	38
10.4.	Authority Identifiers and Metadata . . . . .	39
10.5.	Issuer Discovery . . . . .	40
10.6.	Cross-Authority Validation . . . . .	40
10.7.	Conflict and Split-Brain Handling . . . . .	40
10.8.	Attestation Model . . . . .	41
10.9.	Caching Rules . . . . .	41
11.	Benefits and Trade-offs . . . . .	42
12.	Threat Model . . . . .	43
13.	Economics . . . . .	44
14.	Migration Plan for Organisations . . . . .	44
15.	Security Considerations . . . . .	45
15.1.	Transition of Network Security Roles . . . . .	45
15.2.	Compliance, Observability, and Intercept Boundaries . . . . .	46
15.3.	Identity-Aware Observability Shift . . . . .	46
15.4.	Time as a Trust Dependency . . . . .	47
15.5.	Replay, Caching, and Stale Authority . . . . .	48
15.6.	Downgrade and Compatibility-Mode Abuse . . . . .	48
15.7.	Break-Glass and Emergency Override Discipline . . . . .	49
15.8.	Metadata Leakage and Traffic Analysis . . . . .	49
15.9.	Rendezvous Concentration and Availability Pressure . . . . .	50
16.	IANA Considerations . . . . .	51
17.	Normative References . . . . .	52
18.	Informative References . . . . .	52
	Author's Address . . . . .	53

## 1. Scope and Status

This Internet-Draft is part of an experimental, research-oriented suite prepared for the Independent Stream. It is published to enable structured technical review, interoperability discussion, and disciplined specification development around the Universal Zero-Port Interconnect Framework (UZPIF).

Within that suite, this document defines the current normative baseline for trust objects, validation rules, and security semantics governing the suite's shared identities, Grants, RN roles, and trust relationships. Hard interoperability is expected for shared object semantics and validation rules.

The material is a research artefact. It does not claim technical completeness, production readiness, or endorsement by the IETF or any other standards body, and it is not presented as a standards-track specification.

Full wire-level, clustering, and proof-family interoperability is not claimed everywhere yet. Companion wire encodings, clustering details, proof families, and deployment profiles remain intentionally profile-defined or deferred, so this draft should not be read as claiming a fully closed transport, proof, or availability model.

It is designed for experimentation and profile-driven deployments within its target environment. Privacy, decentralisation, and RN availability remain deployment- and profile-dependent properties.

During conversion from internal research documents into IETF XML, care has been taken to:

- \* preserve a clear distinction between normative and informative content;
- \* use requirement language (e.g., "MUST", "SHOULD", "MAY") only where protocol behaviour is intentionally specified;
- \* avoid any implication of registry finalisation, mandatory implementation, or standard-track status; and
- \* maintain intellectual-property neutrality, with no implied patent grants or licensing commitments beyond the IETF Trust copyright licence applicable to Internet-Draft text.

Ongoing research, implementation, performance validation, and real-world pilot work remain outside the scope of this Internet-Draft text and may be pursued separately.

## 2. Executive Summary

The Internet still commonly exposes services via publicly reachable transport ports, a legacy design choice that enables scanning and unsolicited connection attempts at global scale. Operationally, this contributes to exposure for denial-of-service attacks, credential attacks, and lateral movement within networks.

UZPIF (the framework) and UZP ([UZP]) (its transport protocol) remove the concept of exposed ports at endpoints. Both endpoints initiate outbound, identity-anchored sessions to a Rendezvous Node (RN), which only stitches traffic when identity, context, and declared purpose align under policy issued by Pantheon, the identity and policy plane.

The intent is a model where discoverability and session establishment are policy-mediated rather than assumed by default, and where application traffic can be end-to-end authenticated and encrypted. UZP ([UZP]) is designed to support performance properties such as:

- \* block-level reliability,
- \* selective retransmission, and
- \* deterministic pacing.

Legacy applications and non-UZPIF-capable hardware are intended to continue to operate via a Hardware Integration Layer (HIL) that acts as the explicitly modelled compatibility edge between port-based protocols and identity-centric sessions.

A UZPIF-aligned design should be evaluated not only for its steady-state cryptographic model, but also for whether its bootstrap, recovery, downgrade, mediation, operational override, and observability paths reintroduce singular trust dependencies that the architecture is intended to avoid.

UZPIF builds on transport, security, and identity work embodied in QUIC [RFC9000], TLS 1.3 [RFC8446], and the Host Identity Protocol [RFC7401], while aligning with modern zero-trust guidance (e.g., NIST SP 800-207 [NIST-SP800-207]) and post-quantum cryptography standardisation efforts (e.g., the NIST PQC project [NIST-PQC]).

### 3. Introduction

This document provides an architectural overview of UZPIF and the deployment conditions addressed by an identity-first, rendezvous-based model that avoids publicly reachable listeners.

UZPIF should therefore be read as part of an experimental, research-oriented Independent Stream suite and as the current normative baseline for trust objects, validation rules, and security semantics within the framework. Hard interoperability is expected for shared object semantics and validation rules. Full wire-level, clustering, and proof-family interoperability is not claimed everywhere yet; the remaining details are intentionally profile-defined or deferred. Removing listeners alone does not by itself solve privacy, decentralisation, or RN availability.

### 3.1. Motivation and Deployment Context

- \* Investment in perimeter defences (e.g., DDoS mitigation and application firewalls) can yield diminishing returns as attackers automate scanning and exploit discovery at Internet scale.
- \* Zero Trust Network Access (ZTNA) and SASE deployments indicate demand for identity-first networking, yet many approaches still expose TCP/UDP ingress and rely on perimeter constructs. [NIST-SP800-207]
- \* Post-quantum cryptography efforts provide a path to identity-first transport without prohibitive performance regression as key encapsulation and signature schemes mature. [NIST-PQC]

### 3.2. Relationship to VPN-Based Approaches

Conventional VPNs and overlay networks typically retain the assumption that services listen on IP:port tuples, even if those ports are only reachable within a private address space or through a gateway. QUIC [RFC9000], TLS 1.3 [RFC8446], HIP [RFC7401], and systems such as Tor [Tor] demonstrate that identity, encryption, and rendezvous can be decoupled from raw addressing semantics, but they generally retain listener-based service reachability.

- \* *\*No listeners\** at endpoints.
- \* *\*Identity-as-address\** via identities (e.g., canonical and ephemeral identities) rather than IP:port.
- \* *\*Companion transport profiles\** define session and transport behaviour rather than inheriting a VPN tunnel abstraction.
- \* *\*Pantheon policy plane\** encoding purpose, context, and validity into every session.

## 4. Terminology

*\*Requirements Language:* 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 [RFC2119] and [RFC8174] when, and only when, they appear in all capitals.

This Internet-Draft is primarily architectural; requirement language is used sparingly and only where behaviour is intentionally specified.

- EP Endpoint. A host or service that participates in UZPIF by initiating outbound sessions.
- RN Rendezvous Node. A mediator that accepts outbound sessions and stitches permitted flows.
- RN Controller Control-plane software used to operate an RN deployment, including policy decisions and transparency log publication.
- Pantheon An identity, attestation, and policy plane whose authorities bind identity, policy, and trust metadata to keys or selectors accepted under local policy, may validate or certify those bindings, and may issue credentials, Grants, and delegations over them; when federated, they rely on authority metadata, issuer discovery, cross-authority validation, and conflict handling.
- HIL Hardware Integration Layer. A constrained edge mediation component that exposes or terminates legacy port-based protocols on behalf of non-UZPIF-capable hardware, while participating in UZPIF as a policy-bound identity-aware gateway.
- CID Canonical Identity. A long-term cryptographic identity used to identify an EP (or a delegated sub-identity).
- EID Ephemeral Identity. A short-lived identity used for sessions, derived or issued under policy.
- ZPIT Zero-Port Interconnect Tunnel. An end-to-end encrypted tunnel stitched via one or more RNs.
- Proof of RN Misbehaviour A cryptographically verifiable evidence package that demonstrates RN protocol deviation without requiring a central adjudicator.

## 5. Common Signed Artefact Envelope

This section defines the normative signed-object backbone for the UZPIF suite, including Pantheon Grants, Pantheon Certificates (PCerts), bootstrap and recovery artefacts, revocation signals and threshold evidence as profiled by TLS-DPA ([TLS-DPA]), Proof-of-Reachability evidence as defined by UZP ([UZP]), RN set advertisements, RN transparency entries, Discovery Grants, Signed Checkpoints, Indexer Receipts, and Revocation Acknowledgement artefacts as profiled by outbound indexing ([OUTBOUND-INDEXING]), and related auditable objects. For interoperable exchange across the suite, such objects MUST use this common signed artefact envelope.

The envelope is wire-neutral: it does not define transport framing or carriage format. It does define a single logical object model, canonical serialisation, signature coverage rule, extension rule, and validation baseline so that conforming implementations do not sign the same logical object differently.

Each signed artefact consists of three top-level members: "envelope", "body", and "signatures". The top-level member set is closed; unknown top-level members MUST cause rejection. The "envelope" member carries the suite-wide semantics defined here. The "body" member carries the object-specific semantics defined by the relevant draft. The "signatures" member carries one or more embedded signatures over the canonical signature input defined below. Object-specific drafts MAY extend the body, but they MUST NOT redefine the envelope semantics, canonical serialisation, exact signature coverage, extension handling, object identifier derivation, algorithm identifier handling, epoch-versus-sequence precedence, detached-signature policy, or ordering rules defined in this section.

This section is normatively central for suite-interoperable signed objects. Companion drafts define only the body profile, object-specific eligibility checks, and any explicitly deferred proof-family behaviour for a registered object type. They MUST NOT redefine canonical serialisation, exact signature coverage, object identifier derivation, unknown-field handling, signature ordering, algorithm identifier comparison, epoch-versus-sequence precedence, or detached-signature policy. If a companion draft conflicts with this section, this section controls for suite-interoperable exchange.



```
{
  "envelope": {
    "version": 1,
    "object_type": "grant",
    "object_id": "urn:uzpif:obj:v1:sha256:6f9c8a4c...",
    "issuer_authority_id": "authority:example",
    "subject_id": "cid:subject",
    "audience_id": "cid:audience",
    "scope": "tenant=alpha;service=payments;action=connect",
    "policy_id": "pantheon-connect",
    "policy_version": "2026-03-13",
    "issued_at": "2026-03-13T12:00:00Z",
    "not_before": "2026-03-13T12:00:00Z",
    "not_after": "2026-03-13T12:05:00Z",
    "epoch": 44,
    "sequence": 181,
    "nonce": "4flc8a9d...",
    "key_id": "sig-main-2026q1",
    "prev_hash": "sha256:...",
    "log_ref": "log://rn.example/181"
  },
  "body": {
    "requested_peer": "cid:peer",
    "action": "connect",
    "qos_class": "interactive"
  },
  "signatures": [
    {
      "key_id": "sig-main-2026q1",
      "alg": "ed25519",
      "value": "base64..."
    }
  ]
}
```

Figure 1: Informative example of the common signed artefact envelope.

This figure is illustrative only. The normative envelope semantics, canonical serialisation rules, and validation requirements are defined by the text in this section.

### 5.1. Canonical Serialisation and Signature Coverage

Producers and verifiers MUST use the JSON Canonicalization Scheme (JCS) defined in [RFC8785] over UTF-8 JSON text.

Object production and verification use two exact canonical preimages. First, the producer derives "object\_id" from the JCS serialisation of {"body": body, "envelope": envelope\_without\_object\_id}, where envelope\_without\_object\_id is the final envelope with only the "object\_id" member omitted. Second, after inserting the derived "object\_id", the canonical signature input is exactly the UTF-8 octet string of the JCS serialisation of {"body": body, "envelope": envelope}. The top-level "signatures" member, any detached-signature container, and any transport wrapper are outside both canonical preimages.

The "object\_id" value MUST be formatted as "urn:uzpif:obj:v1:sha256:<lowercase-hex>", where the digest is the SHA-256 hash of the object-id preimage described above. Verifiers MUST recompute and compare "object\_id" before evaluating signatures. Adding, removing, or reordering signatures does not change "object\_id". Any change to envelope or body content, including extensions, validity bounds, policy references, or sequence metadata, MUST produce a different "object\_id".

Each accepted signature MUST cover the exact canonical signature input bytes and nothing else. Suite-interoperable signatures MUST NOT be computed over pretty-printed JSON, partially selected members, alternate canonicalisation schemes, XML renderings, CBOR transcodings, log wrappers, transport records, or presentation-layer containers.

If an object-specific schema admits optional defaults, producers MUST materialise the exact values to be signed before canonicalisation. Verifiers MUST recompute "object\_id" and the signature input from the received values, not from locally inferred defaults or omitted aliases.

Non-finite numbers, duplicate member names, semantically equivalent but non-canonical encodings, or values that cannot be represented under [RFC8785] MUST cause rejection.

## 5.2. Envelope Members and Extensions

Every common signed artefact MUST include "version", "object\_type", "object\_id", "issuer\_authority\_id", "subject\_id", "scope", "issued\_at", "not\_before", "not\_after", "key\_id", and at least one entry in "signatures". The "body" member MUST be a JSON object whose direct member set is closed by the declared "object\_type": direct body members not defined for that object type, "extensions", or "critical\_extensions" MUST cause rejection. "audience\_id", "policy\_id", "policy\_version", "epoch", "sequence", "nonce", "prev\_hash", and "log\_ref" MUST be present when relevant to the

object's semantics, replay model, or transparency linkage.

Authenticity alone is insufficient for reliance on a common signed artefact. A valid signature proves origin and integrity, but a relying party **MUST** also evaluate freshness, scope, supersession, audience binding where relevant, and policy eligibility before treating the artefact as current authority.

**version** Identifies the envelope version so that parsers and relying parties can apply the correct validation rules. Suite-interoperable objects defined by this revision **MUST** set "version" to 1.

**object\_type** Classifies the artefact using the exact case-sensitive token registered in Section 5.3. For suite-interoperable exchange, aliases or profile-specific synonyms are invalid.

**object\_id** Provides the stable identifier for the logical signed object. Producers **MUST** derive it according to Section 5.1. The same envelope and body content **MUST** always produce the same **object\_id**, regardless of signature count. Producers **MUST NOT** reuse an **object\_id** for different envelope or body content.

**issuer\_authority\_id** Identifies the authority context under which the object is issued. It **MUST** map to signed authority metadata or other locally trusted issuer metadata.

**subject\_id** Identifies the principal, resource set, or relationship primarily affected by the artefact.

**audience\_id** Identifies the intended relying party or recipient when the artefact is not intended for universal consumption.

**scope** Defines the operational scope of the artefact, such as authorised actions, resource sets, tenant boundaries, RN context, or revocation domain.

**policy\_id / policy\_version** Name the policy basis and version under which the artefact was issued when policy tracking matters.

**issued\_at / not\_before / not\_after** Define issuance time and validity bounds. Artefacts outside their validity window **MUST** be rejected.

**epoch / sequence** Provide ordering and conflict-detection context.

If both are present, "epoch" takes precedence for semantic supersession and "sequence" orders objects within the same epoch or append-only stream. A higher "epoch" supersedes any lower "epoch" regardless of "sequence". If "epoch" is equal, a higher "sequence" is newer. A lower-epoch object MUST NOT supersede a higher-epoch object merely because its "sequence" is larger.

**nonce** Carries replay-unique entropy for replay-sensitive artefacts such as Grants, PoR evidence, or session-bound authorisations. Re-usable objects that are not replay-sensitive MAY omit it.

**key\_id** Identifies the primary issuer key used for key discovery. It MUST match the first embedded signature entry after signature-array sorting and MUST match at least one embedded signature entry.

**prev\_hash / log\_ref** Link the artefact into an append-only sequence or an external transparency record when auditability or append-only verification is required.

**body members** The direct members of "body" are defined by the registered object-type profile. Unknown direct body members MUST cause rejection unless they are carried under "body.extensions".

**body.extensions / body.critical\_extensions** Object-specific extension values MUST appear only under "body.extensions", which, if present, MUST be a JSON object. "body.critical\_extensions", if present, MUST be an array of unique extension keys sorted in ascending lexicographic order, and each listed key MUST exist in "body.extensions". Extension keys MUST be unique namespaced ASCII strings. Unknown extension values MAY be ignored only if they are not listed in "body.critical\_extensions". Unknown critical extensions MUST cause rejection. Unknown members in "envelope", in an embedded signature entry, or as direct body members MUST cause rejection.

### 5.3. Suite-wide object\_type Registry

The "object\_type" member is a case-sensitive lowercase ASCII token. For suite-interoperable exchange, it MUST use one of the exact values registered in Table 1. Companion drafts define only the body profile and additional validation for their registered object types; they MUST NOT rename, alias, or reinterpret a registered value. New suite-wide object types require an explicit update to this registry.

Detached signatures are not permitted for any registered object type in this revision. A future suite revision MAY change that only by updating both this registry and Section 5.5.

object_type	Body profile	Detached signatures
grant	Pantheon Grant body profile (Section 10.3)	No
pcert	Pantheon Certificate body profile (Section 10.2)	No
bootstrap-seed-bundle	Bootstrap trust seed bundle (Section 8.2.1.1)	No
bootstrap-admission	Bootstrap admission body profile (Section 8.2.1.2)	No
recovery-reenrolment-bundle	Recovery / Re-Enrolment body profile (Section 8.2.1.3)	No
rn-set-statement	RN set advertisement body profile (Section 8.5.2.1)	No
rn-transparency-entry	RN transparency log entry body profile (Section 8.6)	No
misbehaviour-proof	RN misbehaviour evidence body profile (Section 8.7)	No
revocation	TLS-DPA Revocation Signal body profile ([TLS-DPA])	No
threshold-consensus-evidence	TLS-DPA threshold-consensus body profile ([TLS-DPA])	No
por-evidence	UZP Proof-of-Reachability body profile ([UZP])	No
discovery-grant	Outbound indexing Discovery Grant body profile ([OUTBOUND-INDEXING])	No
index-transparency-entry	Outbound indexing transparency-entry body profile ([OUTBOUND-INDEXING])	No

signed-checkpoint	Outbound indexing checkpoint body profile ([OUTBOUND-INDEXING])	No
indexer-receipt	Outbound indexing receipt body profile ([OUTBOUND-INDEXING])	No
revocation-acknowledgement	Outbound indexing revocation-acknowledgement body profile ([OUTBOUND-INDEXING])	No

Table 1: Registered suite-wide object types

#### 5.4. Signature Algorithm Identifiers

The "alg" member is a case-sensitive ASCII token that identifies the exact signature verification procedure applied to the signature bytes in "value". Comparison is byte-for-byte. Aliases, case folding, numeric registry codes, OIDs, or implicit translation between naming schemes MUST NOT be used for suite-interoperable verification.

Trusted issuer metadata MUST bind each signing key identifier to one or more exact "alg" tokens. A verifier MUST accept a signature only when the signature entry's "alg" exactly matches a token bound to that key and local policy permits that algorithm for the declared object\_type and time interval.

Composite or hybrid constructions, if used, MUST have their own exact "alg" token and verification procedure. They MUST NOT be represented by multiple "alg" values within one signature entry or by guessing from key type alone.

#### 5.5. Signature Entries, Ordering, and Detached Form

The "signatures" array MUST contain at least one embedded signature. Each embedded signature entry MUST contain exactly "key\_id", "alg", and "value". The embedded-signature member set is closed.

The array order is significant for serialised interchange and MUST be sorted in ascending lexicographic order by "key\_id", then "alg", then "value". Duplicate entries with the same "key\_id" and "alg", or duplicate complete entries, MUST cause rejection. A received array that is not already in this order MUST cause rejection. The envelope "key\_id" MUST identify the first embedded signature entry after sorting.

The "alg" value is mandatory and case-sensitive, and it is processed according to Section 5.4. A signature entry whose "alg" is absent, unknown, unsupported, mismatched to the key metadata, or forbidden by local policy MUST be treated as invalid. If the remaining valid signatures do not satisfy the required signature set, the object MUST be rejected.

Embedded signatures are the only interoperable signature form for the registered object types in Table 1. For those object types, detached signatures MUST NOT be used and MUST NOT count toward policy satisfaction. If a future suite revision explicitly permits detached signatures for a specific object type, each detached signature MUST cover the same canonical signature input as the embedded form and MUST carry "object\_id", "key\_id", "alg", and "signed\_object\_hash", where "signed\_object\_hash" is "sha256:<lowercase-hex>" over the canonical signature input. If both embedded and detached signatures are present for such a future object type, every signature MUST verify against the same canonical signature input and object\_id; any mismatch MUST cause rejection.

## 5.6. Validation Rules

Relying parties validating a common signed artefact MUST verify at least the following:

- \* the top-level member set, envelope member set, embedded-signature member set, and object-type-specific direct body member set are well-formed and contain no unknown members except under "body.extensions";
- \* the envelope version is supported for the declared object\_type, and registered suite-wide object types defined by this revision use "version" equal to 1;
- \* the declared object\_type is an exact registered suite-wide token when interoperable suite exchange is claimed;
- \* object\_id recomputation succeeds;

- \* issuer\_authority\_id is locally trusted, validly delegated, or otherwise accepted under the applicable federation policy;
- \* the signing key identified by key\_id is valid for the object\_type and time interval, and the envelope key\_id matches the first embedded signature entry after sorting;
- \* signature ordering and duplicate checks succeed, and every accepted signature verifies over the exact canonical signature input;
- \* each "alg" value exactly matches trusted key metadata and a locally permitted signature verification procedure for the declared object\_type;
- \* the required signature set satisfies local policy, including any threshold or quorum rule;
- \* the current time falls within the declared validity interval;
- \* audience\_id matches the relying party when the object is audience-bound;
- \* nonce uniqueness and epoch or sequence freshness checks succeed when replay resistance or state ordering matter;
- \* the artefact has not been superseded, rendered stale under local policy, or made policy-ineligible for the relevant scope and decision;
- \* prev\_hash or log\_ref, when present, are consistent with the relevant append-only or transparency evidence; and
- \* body.critical\_extensions, when present, is unique, sorted, references only keys present in body.extensions, and contains no unsupported critical extension names.

Objects sharing the same issuer\_authority\_id, object\_type, subject\_id, scope, and applicable epoch or sequence state, but with incompatible canonical signature inputs, MUST be treated as conflicting artefacts and processed according to Section 10.7.



## 6. Core Architecture Overview (Informative)

This section is informative. It provides high-level role and flow sketches only. Authoritative session, handshake, stitching, failover, and proof semantics are defined in the companion UZP ([UZP]) and TLS-DPA ([TLS-DPA]) drafts. Authoritative shared object semantics are defined by the common signed artefact envelope and the object-specific sections of this document and the relevant companion drafts.

If any figure or shorthand message label in this section conflicts with the companion transport or handshake drafts, the companion drafts govern.

### 6.1. High-Level Flow: EP to RN to EP (Informative)

In UZPIF, both peers initiate outbound sessions towards an RN. After policy evaluation and authorisation, the RN stitches the two sessions into a tunnel (ZPIT) that carries end-to-end protected application data.

```

EP A (Initiator)           RN           EP B (Responder)
|----- outbound ----->|           |
|                           |<----- outbound|
|<==== end-to-end encrypted ZPIT (stitched via RN) ====>|

```

Figure 2: High-level communication pattern (informative)

This figure shows both endpoints initiating outbound sessions to the RN, which stitches them into a single ZPIT.

The labels in this figure are illustrative only; this document does not define authoritative wire messages or handshake ordering for the transport steps shown.

### 6.2. Pantheon Grant Verification (Informative)

Prior to stitching, an EP is expected to obtain a signed authorisation ("Grant") from Pantheon, as defined in Section 10.3. Grants bind identity to purpose and validity constraints, enabling RNs to make consistent policy decisions. The authoritative Grant format and interoperability requirements are defined in Section 10.3 and Section 5.

```

EP           Pantheon           RN
|-- Grant Request -->|           |
|<- Signed Grant ----|           |
|----- Grant + CID/EID ----->|

```

Figure 3: Grant request and issuance flow (informative)

This figure illustrates the basic grant request and issuance exchange between an EP and Pantheon.

The message labels are illustrative only. Authoritative Grant object semantics are defined in Section 10.3, and authoritative transport or handshake sequencing is defined in companion drafts.

### 6.3. Flow Stitching by the RN (Informative)

The RN establishes a stitched tunnel only if both peers present acceptable identities and authorisations. The exact transport, stitching, failover, and protected-traffic semantics are defined in the companion UZP and TLS-DPA drafts.

```

EP A          RN          EP B
|  -- Join Request -->  |
|  <-- Stitch OK ----->  |
|                          |
|                          |  <-- Join Request --
|                          |  -- Stitch OK ----->
|                          |
|  <===== end-to-end encrypted ZPIT (SessionID-bound) =====>  |

```

Figure 4: Join and stitch establishment (informative)

This figure shows the RN joining two authorised sessions into a stitched tunnel without learning plaintext.

The message names and sequencing in this figure are illustrative only. Authoritative stitching, failover, and session semantics are defined in the companion UZP and TLS-DPA drafts.

### 6.4. Multi-RN Stitching for High-Assurance Tenants (Informative)

UZPIF can be extended to multi-hop stitching, for example where a tenant requires multiple independently operated RNs and attestation chains. End-to-end protection is expected to remain between endpoints.

EP A -> RN1 -> RN2 -> EP B

EP A <===== end-to-end AEAD protected traffic =====> EP B

Figure 5: Multi-hop stitching with end-to-end authenticated encryption (informative)

This figure depicts a multi-hop RN chain while end-to-end AEAD protection remains between endpoints.

This subsection is a deployment sketch only and does not define authoritative multi-RN transport semantics.

## 7. Legacy Hardware Integration via HIL

Where attached equipment cannot natively participate in UZPIF, a Hardware Integration Layer (HIL) MAY mediate between legacy port-based protocols and UZPIF sessions. Real-world examples include industrial devices, medical and laboratory equipment, older appliances, embedded controllers, legacy PBXs, telephony systems, and other equipment whose firmware cannot be remodelled around identity-first session initiation or removal of hard-coded inbound listeners.

A HIL is a compatibility and containment boundary, not a redefinition of legacy protocols as identity-native. A HIL is not an exception to UZPIF. It is the explicitly modelled compatibility edge through which such equipment interacts with an identity-first deployment. A HIL does not make a legacy protocol identity-safe; it constrains and evidences the points at which a legacy protocol is permitted to interact with an identity-first system.

HILs MUST apply explicit policy, minimise exposed legacy surface, and produce auditable evidence for mediated actions. A deployment using a HIL MUST treat the attached device as outside the native UZPIF trust model unless that device is separately enrolled with its own identity and policy context. The HIL therefore terminates the trust boundary cleanly rather than treating legacy traffic as natively identity-bound.

### 7.1. HIL Requirements

A HIL used to bridge non-UZPIF-capable equipment:

- \* MUST be explicitly identified as a translation boundary;
- \* MUST terminate trust domains cleanly rather than representing legacy traffic as natively identity-bound;
- \* MUST enforce a narrow allow-listed protocol surface for the specific attached device or device class;
- \* MUST expose only the minimum necessary legacy port behaviour toward the attached equipment;

- \* MUST NOT elevate unauthenticated legacy assertions into Pantheon, Grant, or attestation truth without validation;
- \* SHOULD support one-way or brokered operation where possible rather than transparent raw pass-through; and
- \* MUST be auditable as a security-critical adapter.

When a HIL mediates an action into UZPIF, it SHOULD bind that action to auditable evidence including:

- \* the HIL identity;
- \* the attached device identity or slot or port identity;
- \* the translated protocol;
- \* the relevant time interval;
- \* the applicable policy or Grant; and
- \* an evidence-log reference or equivalent audit record.

## 7.2. Local Trust Boundary of HIL-Mediated Systems

A HIL secures the boundary between a UZPIF environment and a legacy device interface; it does not guarantee the integrity or confidentiality of the local physical segment behind that boundary. Deployments MUST treat the HIL-attached legacy domain as an independently exposed trust zone unless additional protections are present.

The HIL's protection applies to the mediated protocol boundary, not automatically to the local wire, serial line, bus, switch port, backplane, maintenance interface, or other attached legacy segment behind it. Compromise of that local segment may still permit false device input, spoofed local traffic, local firmware replacement, bus replay, serial-line manipulation, switch-port insertion, maintenance-port misuse, false sensor injection, or manipulation of translated events before they cross into the UZPIF-mediated side.

HIL containment therefore reduces and evidences legacy exposure, but it does not retroactively make the attached hardware domain cryptographically trustworthy. Where stronger assurance is required, deployments SHOULD consider physical controls, tamper detection, local attestation, secure serial or bus wrappers where available, and device-origin integrity checks where feasible.

### 7.3. HIL Software and Supply-Chain Integrity

A compromised HIL is especially dangerous because it sits at the boundary between legacy protocol behaviour and native identity-bound operation. A malicious or subverted HIL may spoof device status, fabricate translated events, widen the permitted legacy surface, create covert ingress, or launder insecure traffic into a deployment that appears to be operating under stronger trust assumptions.

HIL software manifests, configurations, policy bundles, and updates SHOULD be signed and auditable. No HIL software, configuration, or policy change SHALL become authoritative unless its manifest, version, signer set, scope, and policy eligibility validate successfully under the deployment's normal trust model.

HIL update, provisioning, and recovery channels MUST NOT become hidden sovereign authority paths. Local administrative overrides, emergency maintenance paths, or vendor tooling MUST NOT silently bypass the same validation model claimed for ordinary operation. This is aligned with the ICSD ([ICSD]) principle that privileged update or override paths must be modelled so that invalid states are rejected rather than normalised.

### 7.4. Legacy Compatibility Classes

This document distinguishes three compatibility classes for legacy integration so that deployments can state clearly whether a device is fully mediated, tightly brokered, or merely forwarded.

**Class A - Full Mediation** The HIL terminates the legacy protocol and exposes a modelled, policy-bound interface into UZPIF. This is the RECOMMENDED integration model.

**Class B - Brokered Pass-Through** The HIL permits tightly scoped transport bridging for a specific session under explicit Grant, time, scope, and evidence rules. This MAY be used where full mediation is not feasible, but it provides weaker containment than Class A.

**Class C - Direct Forwarding Compatibility Mode** The HIL forwards arbitrary port traffic. This mode is outside the recommended security model, SHOULD NOT be used except as a bounded migration exception, and MUST NOT be represented as equivalent to native UZPIF participation or to Class A or Class B mediation.

## 8. Decentralisation and Authority Model

This section defines constitutional constraints for UZPIF deployments and federation behaviour. These constraints are normative.

RN controllers are permissionless to deploy. No central authorising body is required for RN controller deployment.

UZPIF federation MUST be bilateral or multilateral among consenting parties and MUST NOT require a mandatory hierarchical control model.

UZPIF MUST NOT depend on any single registry, single directory, or single governance entity for protocol operation. UZPIF MUST NOT require a global mandatory trust root.

### 8.1. Bootstrap Model Clarification

Bootstrap MAY use multiple Bootstrap Seed Bundles, mirrored distribution paths, offline media, local ceremony, and DNS or similar mechanisms as transports for seed distribution. The transport used to deliver seed material does not itself create trust; trust begins only after the device validates a Bootstrap Seed Bundle under local bootstrap policy.

Bootstrap mechanisms MUST be replaceable across deployments and over time. Bootstrap mechanisms MUST NOT depend on a single distribution source. No single seed source SHALL be mandatory.

### 8.2. Bootstrap, Recovery, and Re-Enrolment

Bootstrap, recovery, factory reset, re-enrolment, and emergency rekeying are privileged trust transitions. A deployment that appears decentralised during steady-state operation but depends on a singular bootstrap or recovery path has introduced a hidden trust anchor and does not satisfy the intended UZPIF sovereignty model.

New-node bootstrap, client recovery after state loss, RN rejoin after corruption, initial HIL enrolment, and emergency rekeying after compromise MUST NOT require a singular globally mandatory authority or delivery path. These workflows MUST NOT depend solely on one website, one operator, one Recovery / Re-Enrolment Bundle, one vendor endpoint, or one golden administrative key as the only path to valid trust state.

Recovery workflows MUST fail closed on missing, unsigned, expired, conflicting, or policy-ineligible recovery material. Recovered trust state MUST remain independently verifiable through the signed artefacts, authority metadata, and transparency or quorum evidence defined below before it becomes authoritative.

This document defines a minimal baseline model for those privileged transitions. Baseline-conformant bootstrap or recovery support consists of validated Bootstrap Seed Bundle input, explicit first-contact admission through a Bootstrap Admission Object, explicit continuity-sensitive return through a Recovery / Re-Enrolment Bundle when prior state existed, and issuance of only bounded initial or recovered identity and Grant state under policy.

Factory reset and re-enrolment MUST be treated as privileged trust transitions rather than convenience workflows. A reset or re-enrolled endpoint, RN, or HIL MUST re-establish trust as a new or explicitly recovered principal under local policy, and previous trust state MUST NOT be silently reinstated absent verifiable recovery artefacts.

Profiles MAY use multiple Bootstrap Seed Bundles, mirrored distribution paths, offline recovery packages, out-of-band provisioning, or quorum-approved rekey procedures. However, no single bootstrap or recovery source SHALL be mandatory for global protocol validity. This aligns with the invariant-closed design discipline described in ICSD ([ICSD]), where privileged recovery transitions are modelled so that invalid or unauthorised recovery states are rejected rather than normalised.

#### 8.2.1. Bootstrap Trust-Transition Artefacts

For interoperable exchange, bootstrap and recovery trust-transition artefacts MUST use the common signed artefact envelope defined in Section 5. This document defines three baseline object types: "bootstrap-seed-bundle", "bootstrap-admission", and "recovery-reenrolment-bundle".

##### 8.2.1.1. Bootstrap Seed Bundle

A Bootstrap Seed Bundle carries the initial trust material a device uses before it has normal steady-state authority. It is the artefact that introduces the first accepted authority metadata and the first accepted RN set or RN discovery hints into the device's trust evaluation.

A Bootstrap Seed Bundle MUST carry at least:

- \* the initial trusted authority metadata, or stable digests and retrieval references for that metadata;
- \* the accepted RN set, RN discovery hints, or both;
- \* a validity interval suitable for bounded bootstrap use;
- \* a bootstrap policy identifier or equivalent environment or trust-domain identifier;
- \* an epoch or sequence value sufficient for supersession and rollback detection; and
- \* a signature set sufficient for the local bootstrap policy.

A Bootstrap Seed Bundle authorises initial trust discovery only. It does not by itself admit a device, grant long-term authority, or silently create steady-state trust.

At baseline, before first authority contact, the device MUST verify the Bootstrap Seed Bundle's signature set, bootstrap policy identifier, validity interval, authority metadata, and epoch or sequence freshness. Manufacturer certificates, vendor-hosted seed material, or installer media MAY contribute to that evaluation only if explicitly accepted by local bootstrap policy; they are not sufficient by themselves to admit the device or create steady-state trust.

#### 8.2.1.2. Bootstrap Admission Object

A Bootstrap Admission Object is the bounded initial enrolment artefact used to admit a new device or principal into first authority contact. It is a narrow bootstrap credential, not a substitute for steady-state identity, Grant, or administrative authority.

A Bootstrap Admission Object MUST carry at least:

- \* a binding to the device or principal being admitted, such as an attestation digest, presented public key, hardware identifier, or ceremony token;
- \* the intended bootstrap audience, such as the authority context, enrolment service, or admitted RN class;
- \* the permitted first-contact actions, such as initial identity issuance, initial Grant acquisition, or bootstrap-session establishment;



- \* a narrow scope, expiry, and nonce, counter, or equivalent one-time-use bound;
- \* the bootstrap policy identifier under which admission is evaluated; and
- \* the approval signatures or approval evidence required by the local bootstrap policy.

A Bootstrap Admission Object MUST be treated as bounded and bootstrap-only. It MUST NOT be equivalent to unlimited long-term authority, a standing administrator credential, or an implicit right to bypass normal Grant, revocation, or transparency processing.

At baseline, first authority contact MUST fail closed unless the Bootstrap Admission Object matches the presented device or principal binding, intended bootstrap audience, permitted first-contact action set, and one-time-use or bounded-validity constraints. Successful admission MAY yield only bounded initial identity state, bounded initial Grant state, or both, and MUST NOT create broader authority than the object and policy explicitly permit.

When onboarding is mediated by a HIL, the admission decision MUST still be made under bootstrap policy and MUST bind the admitted device or principal and, where relevant, the mediating HIL context. HIL presence, reachability, or vendor tooling is not a substitute for Day Zero trust admission.

#### 8.2.1.3. Recovery / Re-Enrolment Bundle

A Recovery / Re-Enrolment Bundle is the artefact set used after state loss, corruption, key rotation, compromise response, or factory reset. It defines how a device or principal re-establishes trust after prior state existed and therefore carries stronger continuity and incident-handling semantics than initial bootstrap admission.

A Recovery / Re-Enrolment Bundle MUST carry at least:

- \* the recovered subject or replacement subject binding;
- \* the recovery reason or incident class, such as state loss, corruption, compromise, or planned rotation;
- \* references to the last known trusted state, such as prior object identifiers, issuer state, or highest accepted epoch or sequence values when available;

- \* the recovery actions permitted, including whether identity continuity, replacement identity issuance, rekey, or Grant re-establishment is allowed;
- \* a bounded validity interval and intended recovery audience;
- \* the required approval signatures, quorum evidence, or recovery evidence references; and
- \* refreshed authority metadata, RN hints, or both when recovery requires trust-anchor or routing refresh.

Recovery requires a distinct evaluation path from bootstrap. A Recovery / Re-Enrolment Bundle MUST be evaluated under explicit recovery policy and MUST require stronger approval, continuity evidence, or incident classification than ordinary bootstrap admission when prior state existed or compromise is suspected.

At baseline, a Recovery / Re-Enrolment Bundle MUST be evaluated against explicit prior-state linkage, incident classification, intended recovery audience, and the required approval or quorum evidence before any recovered identity or Grant state is accepted. Replaying bootstrap seed or bootstrap-admission artefacts without those tighter recovery checks MUST NOT be treated as valid recovery.

#### 8.2.2. Baseline Trust-Transition Flow

The baseline bootstrap and recovery ceremony is as follows:

1. The device begins outside the closed trust domain and MUST NOT assume Pantheon-recognised identity, Grant state, accepted steady-state RN authority, or inherited session trust.
2. The device receives and validates one or more Bootstrap Seed Bundles through one or more policy-accepted distribution paths and verifies signatures, validity, scope, and supersession state before use.
3. Under bootstrap policy, the device performs first authority contact using the accepted seed material and a Bootstrap Admission Object, or a Recovery / Re-Enrolment Bundle when recovering existing trust state.
4. The contacted authority validates bootstrap or recovery artefacts, applies local bootstrap or recovery policy, and issues bounded initial or recovered identity and Grant state only within the bounds authorised for that transition.

5. The device transitions into normal steady-state trust handling, after which ordinary UZPIF identity, Grant, revocation, transparency, scope, and policy validation rules apply.

If the device is recovering after prior state loss, corruption, or compromise, it **MUST NOT** rely on a Bootstrap Admission Object alone when stronger recovery controls are required by policy. Recovery admission **MUST** be explicit rather than hidden inside an apparently ordinary bootstrap ceremony.

A HIL, manufacturer channel, or deployment installer **MAY** stage artefacts or carry first-contact traffic, but none of them is itself the baseline bootstrap mechanism. The baseline mechanism is the validated trust-transition artefact chain plus the policy-controlled authority action described above.

### 8.2.3. Initial Trust Establishment

The zero-port trust model begins only after initial enrolment. A device cannot originate within the closed trust domain; it must first be admitted to it through an explicit bootstrap process. Bootstrap trust is therefore a first-class architectural concern and **MUST NOT** be treated as an implicit property of zero-port operation.

A brand-new device, endpoint, RN, or HIL is not yet within the closed trust domain and may lack a Pantheon-recognised identity, Grant, trusted RN set, valid policy bundle, known authority set, secure time basis, prior session relationship, or attested operational state. Pre-enrolment identity establishment is therefore an explicit bootstrap transition rather than a mature steady-state protocol property.

Initial enrolment depends on bootstrap artefacts and a bootstrap trust ceremony accepted under deployment policy. At baseline, that means a validated Bootstrap Seed Bundle plus a Bootstrap Admission Object, or a Recovery / Re-Enrolment Bundle when prior trust state is being restored or replaced. Deployments **MUST** define which bootstrap sources and Bootstrap Seed Bundles they accept rather than assuming any universal root of truth. Acceptable bootstrap inputs **MAY** include manufacturer attestation, operator-installed seed trust, pre-provisioned deployment credentials, local physical ceremony, physically mediated enrolment, custodial onboarding authority, verified HIL-assisted enrolment, or enterprise MDM-style enrolment under signed policy. Bootstrap authority **MUST** be explicit, bounded, and auditable.

Manufacturer identity or vendor material MAY be one acceptable bootstrap input, but it does not by itself solve enrolment, is not equivalent to a Bootstrap Admission Object or Recovery / Re-Enrolment Bundle, and MUST NOT silently become sovereign steady-state authority. Admission into Pantheon or another local authority context still requires explicit acceptance under local policy via a Bootstrap Admission Object or stronger recovery artefact.

A HIL MAY assist onboarding, staging, or mediated admission of non-native devices, but HIL presence does not by itself solve the bootstrap problem. HIL-assisted enrolment still requires explicit acceptance of the HIL, its policy context, a validated Bootstrap Seed Bundle, and either a Bootstrap Admission Object or Recovery / Re-Enrolment Bundle under local deployment policy.

Bootstrap trust does not, by itself, confer unlimited steady-state authority. After enrolment, the normal UZPIF model for identity, Grant issuance, revocation, transparency, scope, and policy validation MUST apply. Recovery after state loss, corruption, or compromise MUST transition through the Recovery / Re-Enrolment Bundle model rather than silently restoring prior authority as if no privileged transition had occurred.

### 8.3. Key Custody and Signing-Role Separation

Cryptographic separation of keys is not, by itself, sufficient if the same actor, machine, or operational role can exercise all practically decisive signing and administrative powers. A deployment that concentrates issuance, policy, revocation, software-release, routing, and trust-bundle control into one custody domain has recreated a hidden sovereign control point even if each function uses a distinct key.

Deployments SHOULD separate, where practical, at least some of the following roles:

- \* identity issuance;
- \* Grant or policy signing;
- \* revocation quorum participation;
- \* software release or patch manifest signing;
- \* RN operational administration; and
- \* transparency witnessing or equivalent append-only verification roles.

This document does not mandate one governance structure. It does define the architectural principle that concentrated custody of all such roles SHOULD be avoided when stronger separation is practical, especially for deployments claiming high assurance, multi-party accountability, or resistance to unilateral control.

Where small or local deployments combine roles, that concentration SHOULD be explicit, auditable, and bounded by stronger local controls such as threshold approval, independent logging, external witnessing, or stricter operational review. Combined-role operation MUST NOT be treated as evidence that organisational separation is unnecessary at broader deployment scale.

#### 8.4. Client Sovereignty Model

In UZPIF, clients are the ultimate trust arbiters at the edge.

Clients MUST:

- \* validate RN transparency logs;
- \* verify Proof-of-Reachability (PoR) responses as defined by UZP ([UZP]); and
- \* independently determine trust decisions using local policy and cryptographic evidence.

#### 8.5. RN Controller Properties

An RN Controller is the control-plane implementation used to operate one or more RNs. The following requirements apply to RN Controller designs.

RN Controllers MUST:

- \* be self-hostable without external approval;
- \* publish signed transparency logs;
- \* support cryptographic verifiability of routing behaviour; and
- \* allow client-side trust evaluation.

RN Controllers MUST NOT:

- \* enforce global revocation unilaterally;
- \* depend on a single upstream authority; and

- \* possess unilateral kill-switch capability.

#### 8.5.1. RN Non-Sovereignty and Compromise Containment

Rendezvous Nodes are operationally significant, availability-sensitive infrastructure and MAY be high-value attack targets. However, UZPIF-compliant designs MUST ensure that RN compromise, local administrative access, or physical possession do not by themselves permit issuance of valid identity, Grant, revocation, or control-plane truth. RN-relevant state transitions MUST be cryptographically validated, MUST fail closed on integrity mismatch, and MUST remain independently auditable.

An RN MUST be treated as a traffic-stitching and rendezvous facilitator, a transparency-logged operator, and a cryptographically constrained participant. It MUST NOT become a unilateral authority for identity, Grant issuance, revocation, software truth, or any other control-plane truth that would silently redefine system validity.

RN behaviour MUST be externally verifiable through signed state, append-only transparency records, or other independently checkable evidence. Compromise of one RN MUST NOT permit undetectable falsification of end-to-end identity or authorisation state. Physical control of RN infrastructure MUST NOT, by itself, be sufficient to create valid protocol artefacts, alter authoritative control-plane state, or impersonate endpoints without cryptographically detectable divergence.

RN control-plane changes MUST follow deterministic validation rules. No RN SHALL apply unsigned or policy-ineligible control-plane state. No RN-observed traffic event SHALL be sufficient to mint valid endpoint identity or Grant state. No local administrative path SHALL silently bypass the same verification rules required of normal control-plane transitions.

RN software manifests, configurations, policy bundles, and updates SHOULD be signed by independent release authorities, and deployments SHOULD verify reproducible or attestable build identity where feasible. No RN software, configuration, or policy update SHALL become authoritative unless its manifest, version, signer set, scope, and policy eligibility validate successfully. Update or provisioning channels MUST NOT become hidden sovereign authority paths. Clients and relying parties MUST NOT infer trust from RN software version claims alone.

Deployments SHOULD provide operator plurality, multi-RN alternatives, or client escape paths where practical so that no single RN or RN Controller is the sole required trust anchor for continued protocol validity. Companion work on Invariant-Closed System Design (ICSD) ([ICSD]) is informative for structuring RN controller state so that rollback, drift, partial-write, unsigned-override, or otherwise invalid states are rejected rather than normalised.

#### 8.5.2. Multi-RN Consistency and Failover Semantics

Deployments MAY use multiple RNs for resilience, locality, operator diversity, or high-assurance stitching. In such deployments, RNs remain coordination points rather than sovereign issuers of identity, Grant, revocation, or other control-plane truth. Divergent RN views MUST NOT, by themselves, create valid authorisation or identity truth.

Deployments that require resilience SHOULD provision, discover, or otherwise maintain multiple candidate RNs acceptable under local policy. RN identity, role eligibility, and participation in stitching MUST be independently verifiable from Pantheon artefacts, RN Set Statements, transparency evidence, and local policy rather than inferred merely from referral by another RN. UZPIF does not require a single globally authoritative RN, and compliant deployments SHOULD avoid designs in which loss of one RN renders an otherwise healthy trust domain permanently unreachable.

##### 8.5.2.1. RN Set Advertisement and Discovery

For interoperable RN discovery, a deployment SHOULD publish one or more RN Set Statements using the common signed artefact envelope with "object\_type" set to "rn-set-statement". A Bootstrap Seed Bundle or Recovery / Re-Enrolment Bundle MAY carry one or more RN Set Statements directly, or MAY carry references and digests that allow retrieval and validation of the current RN Set Statement under the same trust policy.

Endpoints claiming resilient rendezvous MUST validate an RN Set Statement under the common envelope rules before using it for RN selection. At baseline, that means verifying the signature set, trust-domain scope, validity interval, and epoch or sequence state, and preferring the highest eligible non-conflicting statement for the relevant scope. Where policy and deployment topology permit, endpoints SHOULD maintain multiple currently eligible RNs rather than a single active candidate.

An RN Set Statement MUST carry at least:

- \* the issuing authority context or trust domain for which the RN set is valid;
- \* the eligible RN identifiers and sufficient contact or discovery hints for each RN;
- \* the RN roles, capabilities, or scope restrictions relevant to stitching, relay, or recovery use;
- \* the validity interval for the RN set;
- \* an epoch or sequence value sufficient for supersession and rollback detection;
- \* optional locality, priority, diversity, or operator-separation hints; and
- \* the signature set required by the deployment policy.

RN referrals learned from an RN MAY be used as hints, but they MUST NOT by themselves establish RN eligibility. Before an endpoint uses an alternate RN for authorisation-relevant operation, the RN MUST be accepted by a valid RN Set Statement, a Bootstrap Seed Bundle, a Recovery / Re-Enrolment Bundle, or another equally trusted signed bootstrap or recovery artefact accepted under local policy.

An RN Set Statement is the baseline plurality artefact: it advertises which RNs are eligible for a given scope and role, but it does not by itself authorise a session or override current Grant, revocation, transparency, or other control-plane truth.

If multiple RN Set Statements conflict for the same trust domain, scope, epoch, or sequence, endpoints MUST process them using the normal conflict rules for signed objects and MUST fail closed for authorisation-expanding actions until a non-conflicting eligible RN set is established.

#### 8.5.2.2. Alternate RN Eligibility and Failover Baseline

An endpoint MAY treat an alternate RN as eligible only if all of the following are true:

- \* the RN appears in a currently valid RN Set Statement or equally trusted bootstrap or recovery artefact;
- \* the RN role and capability set are compatible with the requested stitching or recovery action;



- \* the RN is not locally distrusted, revoked, or outside the currently accepted trust-domain scope; and
- \* the endpoint has enough fresh authority and continuity state to satisfy the chosen failover mode.

Prior successful use of an RN does not by itself keep that RN eligible. Eligibility **MUST** be re-evaluated at failover time against current RN Set Statements, current policy scope, current distrust or revocation state, and the exact stitching or recovery role the endpoint is attempting to use.

Baseline failover triggers include RN loss, overload, local unreachability, policy-eligible alternate-RN preference, bounded continuity expiry, transparency inconsistency, partition suspicion, or signed or otherwise locally accepted evidence of RN misbehaviour. Triggering failover does not by itself alter identity, Grant validity, or authority truth.

When a failover trigger fires, the endpoint enters failover handling, freezes new authorisation-expanding actions, and retains only portable state pending alternate-RN selection and revalidation. It **MUST NOT** promote the failed or disconnected RN into a truth anchor merely because it was the last RN in use.

The baseline portability rule is: identity, Grants, revocation state, Pantheon metadata, RN Set Statements, transparency checkpoints, and other externally verifiable authorisation artefacts are portable across RNs; RN-local queue placement, opaque stitch handles, pacing state, relay caches, and other non-verifiable relay state are not.

#### 8.5.2.3. Re-Stitching and Bounded Continuity State

Fresh authorisation presentation is the default baseline for re-stitching at a new RN. An alternate RN **MAY** accept bounded continuity state instead of full fresh authorisation only for non-expanding continuity when local policy permits it.

Bounded continuity state **MUST** be externally verifiable or endpoint-verifiable and **MUST** be bound to the authenticated subject, peer, accepted Grant object identifiers or digests, relevant authority epoch or sequence state, and the accepted RN set. It **MUST** carry its own expiry and **MUST NOT** be treated as standing authority.

Failover **MAY** reuse bounded continuity state only when the resulting session scope is unchanged or narrower, the underlying Grant or equivalent authorisation remains valid, replay and freshness checks still succeed, and no unresolved RN disagreement or partition

condition would make the reused state ambiguous. Otherwise, the endpoint MUST obtain fresh authorisation or perform a fresh re-stitch under current policy.

Before an alternate RN returns a session to normal stitched operation, the endpoint and alternate RN MUST revalidate at least the chosen RN's current eligibility, the applicable RN Set Statement or equally trusted bootstrap or recovery artefact, current Grant and revocation state for the requested scope, any required transparency or misbehaviour evidence, and the binding, freshness, and expiry of any continuity state that will be resumed.

Session continuity MUST NOT be confused with trust continuity. Even when transport or application continuity is preserved, the alternate RN MUST revalidate the externally verifiable artefacts needed for the requested action and MUST NOT inherit opaque authorisation-relevant state solely because another RN previously held it.

#### 8.5.2.4. RN Disagreement and Partition Handling

Loss, overload, unreachability, or partition of an RN is an availability event rather than an authorisation event. Failure of one RN MUST NOT by itself invalidate endpoint identity, Grants, revocation state, or other externally verifiable control-plane truth.

If two RNs or RN clusters present conflicting control views for the same session, subject, scope, epoch, or routing decision, endpoints and relying services MUST treat that disagreement as an availability or continuity issue rather than as proof that either view is newly authoritative.

Under RN disagreement or suspected partition, endpoints MUST suspend new authorisation-expanding actions until the relevant artefacts are revalidated against externally verifiable authority. Endpoints MAY continue narrowly scoped, already-established non-expanding operation only within an explicitly bounded continuity window allowed by local policy. When the disagreement cannot be resolved within that window, implementations SHOULD prefer fresh re-stitching or clean session termination over silent continuity shortcuts.

If only one RN remains reachable during disagreement or partition, endpoints MUST still treat external artefacts rather than that RN's local control view as authorisation truth. A lone surviving RN is a remaining coordination path, not a newly elevated trust anchor.

#### 8.5.2.5. RN Failover Continuity Constraints

Failover MAY preserve continuity only if end-to-end trust assumptions are preserved. An RN failover path MUST NOT silently weaken identity binding, freshness checks, replay resistance, transparency requirements, or downgrade posture merely to keep traffic flowing. When continuity and freshness are in tension, deployments SHOULD prefer fresh and independently verifiable authority for authorisation-expanding decisions, while allowing only bounded, policy-defined continuity behaviour for non-expanding recovery or session resumption.

Session continuity across RN loss is not guaranteed. A second RN or RN cluster MAY resume rendezvous only from externally verifiable artefacts and bounded continuity state accepted under local policy; it MUST NOT inherit opaque trust-relevant state solely because another RN previously held it. When no safe resumable state exists, implementations SHOULD prefer fresh re-stitching over continuity shortcuts that weaken trust assumptions.

This document does not require peer-to-peer fallback, HIL substitution, or reversion to inbound-port exposure as a universal recovery mechanism. Profiles MAY define degraded or mediated recovery modes, but such modes MUST be explicit, policy-bound, auditable, and MUST NOT silently reintroduce weaker trust assumptions, hidden authority paths, or legacy open-listener exposure merely to preserve reachability.

#### 8.6. RN Transparency Log Format

Each RN MUST publish a transparency log as an append-only sequence of signed entries. Logs MUST be cryptographically verifiable, timestamped, and publicly retrievable. Transparency entries used for interoperable verification MUST use the common signed artefact envelope defined in Section 5 with "object\_type" set to "rn-transparency-entry".

Each RN transparency log MUST contain signed records for:

- \* uptime attestations;
- \* session summaries at aggregated granularity; and
- \* error reports.

Signed policy decisions are OPTIONAL but RECOMMENDED and SHOULD be published when policy constraints allow.

Each log entry MUST include at least a monotonically increasing sequence number, a timestamp, an entry type, a payload digest, and an RN signature. In the common envelope, these properties are expressed via "object\_type", "sequence", "issued\_at", "prev\_hash" or "log\_ref", the object-specific body, and the embedded signature set. Log interfaces MUST allow third parties to retrieve entries and independently verify append-only integrity.

Evidence published for accountability SHOULD be sufficient for independent verification without unnecessarily exposing protected content, business-sensitive metadata, or broader activity graphs.

#### 8.7. Proof of RN Misbehaviour

A Proof of RN Misbehaviour is a cryptographic evidence object that can be validated without a central judge. Misbehaviour proofs used for interoperable exchange MUST use the common signed artefact envelope defined in Section 5 with "object\_type" set to "misbehaviour-proof".

Proofs MAY be constructed from one or more of the following evidence classes:

- \* signed evidence of misrouting;
- \* evidence that an RN failed to relay a required nonce during reachability procedures defined by UZP ([UZP]);
- \* log inconsistency proofs (for example, two different signed entries for the same sequence number) derived from the transparency material in Section 8.6; and
- \* signature replay detection evidence.

Clients MUST be able to validate misbehaviour proofs locally using published keys and log material. When proof validation succeeds, clients MAY blacklist the RN and SHOULD support policy controls for local blacklist retention and expiry.

UZPIF MUST NOT require a token-slashing mechanism or any central adjudication entity for RN accountability. Exclusion decisions are evidence-based and locally enforceable.

#### 9. Explicit Non-Goals

To avoid governance ambiguity, the UZPIF stack does not define or require the following:

- \* enforcement of a global content policy;
- \* mandatory jurisdictional compliance mechanisms;
- \* centralised identity arbitration; and
- \* global kill-switch functionality.

## 10. Pantheon: Identity and Governance Model

Pantheon is the identity, attestation, and policy plane for UZPIF. It MAY be deployed as a single authority or as a multi-authority federation. As specified in Section 8, Pantheon federation is not a mandatory hierarchy and does not imply a single global authority.

A deployment MUST NOT claim interoperable Pantheon federation unless relying parties can obtain authority metadata, discover issuers, apply cross-authority validation, and process conflicts or split-brain conditions using the rules in this section or a stricter profile. Without those capabilities, Pantheon remains a local authority deployment rather than an interoperable federation.

Pantheon objects exchanged across authorities MUST use the common signed artefact envelope defined in Section 5.

### 10.1. Identity Model

Pantheon authorities bind identity, policy, and trust metadata to keys or selectors accepted under policy.

Deployments MAY use locally generated or externally attested keys if policy allows; Pantheon authorities validate or certify those bindings rather than being required to generate the underlying private keys.

In the suite baseline, Pantheon authorities issue credentials, Grants, and delegations over accepted bindings rather than being required to generate or custody the underlying private keys.

Pantheon commonly binds or delegates the following identity-bearing material:

- \* long-term public signing keys bound to CIDs;
- \* ephemeral session keys or ephemeral public keys bound to EIDs under policy; and

- \* delegated sub-identities, selectors, or delegated credentials for services or microprocesses.

This identity approach is conceptually aligned with HIP's separation of endpoint identities from locators [RFC7401], but elevated to a policy plane.

## 10.2. Certificate Format

Pantheon Certificates (PCerts) used for interoperable federation SHOULD include at least the following elements:

- \* an issuer authority identifier;
- \* a CID and public signing key (and optionally a post-quantum key encapsulation key);
- \* purpose tags (e.g., service, role, tenant);
- \* validity bounds (time or epoch); and
- \* optional attestation claims (e.g., hardware trust or enclave measurement).

A PCert binds identity, scope, and trust metadata to an asserted public key. It does not by itself state how the corresponding private key was generated or who holds custody of it unless explicit attestation claims in the object-specific body say so.

Profiles MAY extend PCerts with additional fields, but interoperable federation requires enough issuer, subject, and validity information for cross-authority validation under Section 10.6.

For interoperable exchange, a PCert MUST use the common signed artefact envelope with "object\_type" set to "pcert". The certificate-specific fields listed above populate the object-specific body and MUST NOT redefine envelope semantics.

## 10.3. Grant Structure

A Grant is described as a signed assertion binding:

- \* the CID/EID of the requester;
- \* the requested peer identity;
- \* purpose and action;

- \* a time window and replay nonce; and
- \* an authorised quality-of-service (QoS) class.

For interoperable exchange across the suite, a Grant MUST be represented as a signed artefact with "object\_type" set to "grant", using the envelope defined in Section 5.

This section is the canonical suite-level Grant definition for the architecture described in UZPIF. Companion drafts such as UZP, TLS-DPA, and outbound indexing may add object-specific fields or application-specific constraints in the object-specific body, but they MUST preserve this baseline Grant semantics and the shared artefact envelope.

#### 10.4. Authority Identifiers and Metadata

Each Pantheon authority participating in a federation MUST be identified by a stable authority\_id. An authority\_id MUST remain stable across ordinary signing-key rollover and MUST be unique within the relying party's trust context.

Each authority MUST publish signed authority metadata sufficient for relying parties and peer authorities to validate issued objects. At a minimum, the metadata MUST identify:

- \* the authority\_id;
- \* one or more current signing keys, and optionally future or retiring keys for rollover;
- \* service endpoints used for issuing or retrieving Pantheon objects;
- \* a metadata validity interval;
- \* the supported object types (for example PCerts, Grants, delegations, revocations, or attestation artefacts); and
- \* an optional transparency-log endpoint.

Metadata MAY contain additional deployment-specific policy claims, but Pantheon federation MUST NOT depend on a single global metadata registry or a mandatory schema beyond the minimum semantics above.

### 10.5. Issuer Discovery

Issuer discovery is deliberately non-centralised. A relying party MAY discover an authority and its signed metadata through one or more of the following mechanisms:

- \* local trust bundles;
- \* signed referrals;
- \* cached authority metadata; and
- \* out-of-band provisioning.

No global registry, mandatory directory, or single discovery service is required for Pantheon federation. Deployments MAY combine discovery mechanisms and MAY continue to rely on cached metadata when fresh discovery is temporarily unavailable, subject to local expiry policy. An authority without discoverable or locally provisioned metadata MUST NOT be treated as an interoperable federation participant.

### 10.6. Cross-Authority Validation

A client, RN, or other relying service evaluating a PCert, Grant, revocation statement, delegation, or other signed Pantheon object MUST accept that object only if its issuer is:

- \* locally trusted;
- \* delegated from a locally trusted authority, with a delegation chain valid for the relevant object type and scope; or
- \* accepted under a profile-defined federation rule.

If none of these conditions holds, the object MUST be rejected. Validation MUST include signature verification, issuer metadata validity, object-type authorisation, scope, and time validity.

Acceptance of one object type from an authority does not automatically authorise acceptance of other object types unless the applicable delegation or federation profile explicitly permits it.

### 10.7. Conflict and Split-Brain Handling

Pantheon federation MUST distinguish between issuer misbehaviour and federation divergence. These cases are not equivalent and MUST NOT be processed using a single merge rule.



If the same authority issues conflicting signed objects for the same epoch and scope, relying parties MUST treat the condition as evidence of authority misbehaviour. The conflicting objects SHOULD be retained as auditable evidence and MAY trigger local distrust, quarantine, or incident-response policy.

If different authorities publish conflicting signed views for the same subject, scope, or revocation state, relying parties MUST treat the condition as federation divergence. Such conflicts MUST NOT be auto-merged or silently normalised. Resolution MUST be performed by local policy, an explicit federation profile, or quorum rules agreed by the deployment.

Until resolved, implementations SHOULD fail closed for authorisation-expanding decisions and SHOULD surface the divergence to operators or relying services with enough issuer metadata to support audit and review.

#### 10.8. Attestation Model

RNs and endpoints may publish attestations such as:

- \* hardware and software measurements;
- \* configuration hashes; and
- \* policy compliance data.

Attestation artefacts SHOULD be published in transparency logs retrievable by relying parties. Storage and serving MAY be provided by Pantheon services, RN Controllers, or both, mirroring transparency and verifiability goals in broader zero-trust guidance.  
[NIST-SP800-207]

#### 10.9. Caching Rules

The following caching guidance is illustrative deployment guidance rather than a protocol-level interoperability requirement. Deployment profiles MAY tighten or relax these values according to their revocation, freshness, and transparency policies.

Authenticity alone is insufficient for continued reliance on cached authority. Before cached metadata, PCerts, Grants, or attestation material are used for security-relevant decisions, implementations SHOULD evaluate freshness, scope, sequence or epoch where applicable, and current policy eligibility under local state-state rules. Authorisation-expanding decisions SHOULD require revalidation once a locally permitted stale window has been exceeded or when superseding state may exist.

Endpoints may cache:

- \* authority metadata for the shorter of its stated validity interval or a locally configured cache cap;
- \* PCerts for a locally configured interval appropriate to the deployment's revocation and freshness model (for example, up to 24 hours in low-churn environments);
- \* Grants for the shorter of their validity window or session lifetime; and
- \* attestation proofs for the duration of RN handshake paths.

## 11. Benefits and Trade-offs

UZPIF and UZP ([UZP]) intentionally reuse established transport and cryptographic primitives, but change where and how they are bound to identity, policy, and reachability. In particular, QUIC [RFC9000] and TLS 1.3 [RFC8446] demonstrate encrypted transports with modern handshake properties, while UZPIF shifts the reachability model away from listening endpoints.

Dimension	UZPIF/UZP ([UZP])	Traditional TCP/TLS	QUIC/TLS
Exposure	No open ports (endpoints are not publicly listening)	Open ports and/or proxies	Open ports at network edge
Identity	Mandatory cryptographic identity	Typically TLS-level identity only	TLS-level identity
Transport semantics	Defined by companion transport profile	TCP byte-stream transport	QUIC stream transport
RN trust	Drop/delay only (no end-to-end plaintext visibility expected)	N/A	N/A
Latency control	Deterministic pacing (design goal)	Congestion control variants (e.g., Reno/CUBIC)	Congestion control variants (e.g., BBR/CUBIC)
Legacy applications	Supported via HIL (intended)	Native	Often requires gateways or adaptation
Post-quantum readiness	Designed for cryptographic agility	Inconsistent deployment	Emerging

Table 2: Comparison of transport architectures

## 12. Threat Model

This section sketches attacker classes and example controls. It is not a complete security analysis and will evolve with implementation experience.

\*Attacker classes\* include:

- \* Internet-wide scanners;
- \* botnets seeking command-and-control beacons;
- \* malicious RNs (assumed capable of drop/delay/reorder);
- \* insiders with credentials; and
- \* traffic analysts performing correlation.

Existing rendezvous and overlay systems (e.g., Tor [Tor]) and NAT traversal mechanisms based on STUN [RFC5389] and TURN [RFC5766] demonstrate the power of indirection, but they still assume exposed or discoverable listeners somewhere in the path. UZPIF's design intent is to remove those listeners from the endpoint security model.

Example controls discussed for UZPIF include:

- \* end-to-end authenticated encryption (AEAD);
- \* RN and endpoint attestation;
- \* puzzles and identity-bound rate limits;
- \* multi-RN stitching for higher assurance; and
- \* post-quantum readiness and cryptographic agility (see [NIST-PQC]).

### 13. Economics

- \* Capital expenditure reduction: reduced reliance on perimeter appliances and complex DMZ designs.
- \* Operational expenditure reduction: fewer ACL/NAT rule changes and less inbound exposure management.
- \* Risk reduction: reduced externally visible attack surface.
- \* Potential service models: governance and RN validation as managed components.

### 14. Migration Plan for Organisations

UZPIF is intended for incremental deployment alongside existing TCP/TLS and QUIC-based stacks [RFC8446] and [RFC9000].

1. \*Deploy an RN:\* Introduce an outbound-only rendezvous node.

2. \*Deploy the HIL:\* Install the Hardware Integration Layer at endpoints or device edges where legacy applications or hardware require mediated compatibility.
3. \*Dual-stack operation:\* Run UZP ([UZP]) alongside existing TCP/TLS.
4. \*Cutover:\* Migrate services gradually to zero-port operation.

## 15. Security Considerations

UZPIF's central security claim is that avoiding publicly reachable listeners at endpoints reduces exposure to scanning and unsolicited ingress. However, the framework introduces reliance on identity, authorisation, and policy evaluation components (e.g., Pantheon and RNs) whose compromise or misconfiguration could impact availability and authorisation correctness.

The threat model in Section 12 discusses attacker classes and candidate controls. Future revisions of this document (and the companion UZP ([UZP]) and TLS-DPA ([TLS-DPA]) documents) are expected to provide a more systematic analysis, including key management, revocation, attestation trust, and traffic analysis resistance.

### 15.1. Transition of Network Security Roles

UZPIF shifts primary access control from network-coordinate filtering toward cryptographic identity, Grants, and invariant-bound policy enforcement. Network address and open-port exposure therefore cease to be the primary admission model in a native UZPIF deployment.

Traditional firewalls, DPI systems, and other middleboxes MAY remain useful for containment, telemetry, egress control, QoS enforcement, volumetric denial-of-service handling, forensic monitoring, HIL containment, regulatory inspection domains, and non-UZPIF legacy enclaves. However, they MUST NOT be treated as authoritative truth sources for identity or policy validity.

UZPIF does not assume that packet visibility implies authority. Where middleboxes remain deployed, they operate as bounded policy, containment, or observability components rather than as trust anchors that define whether an identity or Grant is valid.

### 15.2. Compliance, Observability, and Intercept Boundaries

Some deployments will require explicit policy enforcement, enterprise inspection, regulated access, or other observability controls. This document does not define a universal lawful-intercept or hidden inspection authority for UZPIF.

Where compliance, monitoring, mediation, or inspection components are deployed, they MUST remain explicit, bounded, and cryptographically distinguishable from protocol truth. The existence of such a component MUST NOT silently redefine endpoint identity, Pantheon authority, Grant validity, revocation state, or other authorisation-relevant truth in the UZPIF suite.

A compliance or inspection function MAY enforce local policy, provide observability, or participate in a regulated deployment boundary, but it MUST be treated as a deployment-specific control surface rather than as implicit sovereign protocol authority. Operator or institutional pressure to insert monitoring MUST NOT be satisfied by silently converting middleboxes, RNs, HILs, or auxiliary services into hidden trust anchors.

### 15.3. Identity-Aware Observability Shift

Network-address and port-oriented monitoring may lose significant explanatory power in UZPIF environments. Legacy network-centric visibility therefore becomes incomplete rather than useless: defenders may see less value in source or destination address patterns, open-port exposure, unsolicited inbound attempts, protocol signatures, or packet-shape heuristics than in conventional port-based environments.

This does not imply that UZPIF blinds defenders. It does mean that security analytics SHOULD rely increasingly on identity-aware telemetry, authority artefacts, Grant usage patterns, rendezvous behaviour, and policy-verification events rather than on legacy assumptions about open-port exposure. Otherwise, Shadow IT activity, stealthy misuse of valid outbound behaviour, unauthorised stitching requests, abuse of legitimate-looking identity objects, or compromised internal workloads acting within apparently allowed tunnel patterns may become harder to explain or detect.

Privacy-related properties in UZPIF MUST be separated. Reduced service discoverability means that publicly reachable listeners and routine unsolicited scanning lose some value. Encrypted content protection means that authorised endpoints can protect payloads and application content from intermediaries. Traffic-pattern privacy concerns whether observers can still infer communication existence,

counterpart relationships, cadence, timing, or operational significance from metadata. Improvement in one of these properties MUST NOT be represented as automatically delivering the other two.

Effective deployment profiles SHOULD expose auditable telemetry such as Grant issuance and usage logs, RN selection patterns, failed stitching attempts, revocation-consistency signals, identity or authority anomalies, HIL mediation events, downgrade or compatibility-mode activation, and unusual policy-bound session graphs. Such telemetry SHOULD remain explicit, policy-bound, and reviewable rather than being inferred indirectly from legacy packet visibility alone.

#### 15.4. Time as a Trust Dependency

Time is a security-relevant input in the UZPIF suite. Grant expiry, revocation freshness, log checkpoints, replay defence, policy activation windows, and software or patch manifest validity all depend on time evaluation. Implementations MUST therefore treat local time and time-derived freshness decisions as part of the trusted security boundary rather than as a purely operational convenience.

Policy decisions that depend on time SHOULD tolerate bounded clock drift as defined by deployment policy or a stricter profile. However, no single unauthenticated time source SHALL become a mandatory truth anchor for continued protocol validity. Deployments SHOULD use independently checkable, redundant, or otherwise policy-vetted time inputs where time correctness materially affects authorisation or freshness decisions.

When time is uncertain, inconsistent, or outside locally permitted drift bounds, implementations SHOULD degrade safely rather than silently accepting stale or not-yet-valid authority. Safe degradation MAY include rejecting time-sensitive artefacts, requiring revalidation, narrowing replay windows, surfacing operator alarms, or falling back to more conservative local policy.

This document does not define a suite-wide time synchronisation protocol. It does define the security posture that incorrect or manipulated time MUST NOT silently widen authority, revive expired Grants, delay emergency revocation effect, or normalise ambiguous checkpoint ordering without explicit local policy.

### 15.5. Replay, Caching, and Stale Authority

Authenticity alone is insufficient for security-relevant artefacts in the UZPIF suite. Grants, revocation signals and threshold evidence as profiled by TLS-DPA ([TLS-DPA]), PoR artefacts as defined by UZP ([UZP]), Proofs of RN Misbehaviour (Section 8.7), transparency checkpoints and receipts as profiled by outbound indexing ([OUTBOUND-INDEXING]), and similar signed objects MUST also be evaluated for freshness, scope, sequence or epoch, audience where relevant, and current policy eligibility before they influence authorisation or trust decisions.

An old but well-signed artefact MUST NOT be accepted merely because its signature validates. Implementations MUST determine whether the artefact has expired, has been superseded by a newer eligible artefact, falls outside the current policy scope, or remains within any locally permitted stale window for the decision being made.

Profiles SHOULD define bounded stale windows for continued operation during outage or partition conditions. Within such a window, implementations MAY continue to rely on cached authority for narrowly scoped, non-expanding behaviour if local policy permits it. Outside that window, or when freshness cannot be established, systems SHOULD require revalidation before continuing security-relevant operation and SHOULD fail closed for authorisation-expanding decisions.

### 15.6. Downgrade and Compatibility-Mode Abuse

UZPIF deployments may support native operation, HIL-mediated operation, brokered compatibility modes, legacy transport fallback, optional transparency behaviour, or advisory revocation thresholds. These modes create downgrade risk if an attacker can force the system onto a weaker posture while preserving the appearance of normal operation.

Fallback and degraded modes MUST be explicit, observable, and policy-bound. An implementation MUST NOT silently downgrade from native identity-first operation to mediated, brokered, legacy-compatible, transparency-reduced, or revocation-weakened behaviour without an allowed policy rule and locally observable state indicating that the downgrade occurred.



Peers, operators, or relying services SHOULD be able to determine whether a session or decision is operating in native, mediated, brokered, or otherwise degraded mode. Degraded operation MUST NOT silently inherit the same trust posture, authority scope, or audit assumptions as native operation. Profiles SHOULD narrow permissions, reduce session lifetime, strengthen revalidation requirements, or require additional operator visibility when compatibility or degraded modes are active.

Suppressed transparency checks, unavailable revocation evidence, weaker identity binding rules, or temporary compatibility exceptions MUST be treated as explicit trust reductions rather than harmless operational details. If the required stronger mode cannot be established, implementations SHOULD fail closed for authorisation-expanding behaviour unless local policy explicitly permits the weaker mode for a bounded scope and duration.

#### 15.7. Break-Glass and Emergency Override Discipline

Real-world deployments may require emergency action during outage, recovery, migration, or incident containment. However, a break-glass path that silently disables normal trust checks is itself a high-risk authority path.

Any emergency override MUST be explicit, narrowly scoped, time-bounded, auditable, and incapable of silently becoming normal steady-state policy. A break-glass decision MUST NOT be normalised as an invisible standing exception to identity validation, Grant evaluation, revocation processing, transparency expectations, or other authorisation-relevant checks in the UZPIF suite.

Deployment profiles SHOULD record the triggering reason, affected scope, approving authority, activation time, expiry condition, and review outcome for such overrides. Emergency operation SHOULD surface clear local state to operators and relying parties when the stronger normal posture has been reduced.

#### 15.8. Metadata Leakage and Traffic Analysis

UZPIF can materially reduce direct service and endpoint discoverability without guaranteeing metadata indistinguishability. Removing publicly reachable listening ports does not, by itself, provide strong privacy or anonymity. Unscannable is not the same as unseen. Rendezvous, transparency, and indexing infrastructure MAY still expose valuable metadata such as which parties communicate, which RN or RN cluster they use, how often they communicate, when sessions occur, how long they persist, retry patterns, topology clusters, discovery behaviour, revocation bursts, patch or rollout

waves, and whether bursts correlate with alarms, updates, or emergency operations.

Even where payload plaintext remains protected, this metadata can support correlation, surveillance, tenant mapping, operational fingerprinting, or coercive analysis. A Rendezvous Node or indexing service that is not a truth anchor may still become a surveillance anchor if deployments over-retain, over-expose, or casually centralise metadata visibility.

Reduced discoverability, encrypted content protection, and traffic-pattern privacy are separate design axes. UZPIF can reduce unsolicited inbound exposure and support protected payload transport while still leaving observable metadata patterns available to infrastructure operators or network observers. Observers such as carriers, backbone operators, ISPs, or state-level monitors may still infer communication existence, cadence, rendezvous relationships, and operational timing from outbound rendezvous patterns unless additional privacy-preserving measures are deployed. Claims about strong observation resistance or traffic-pattern privacy **MUST** therefore be bounded and profile-dependent.

Operators **SHOULD** minimise metadata retention, restrict unnecessary visibility, and avoid collecting or exposing finer-grained metadata than is needed for security, accountability, or operational continuity. Deployment profiles **SHOULD** define retention bounds, access controls, aggregation rules, and disclosure controls for security-relevant logs, receipts, checkpoints, and session records.

Privacy-sensitive profiles **MAY** use identifier rotation, cover traffic, padding, batching, aggregation, timing obfuscation, multi-RN distribution, traffic shaping, delayed publication, or metadata-minimising RN policy to reduce correlation risk where those measures are compatible with the deployment's accountability and performance goals. Removal of inbound ports alone **MUST NOT** be treated as sufficient support for broader privacy claims.

#### 15.9. Rendezvous Concentration and Availability Pressure

UZPIF and its companion mechanisms introduce stateful verification paths, including handshake processing, PoR validation, Grant verification, transparency checking, authority discovery or resolution, repeated stitching attempts, and HIL-mediated translation work. These paths may improve exposure relative to openly listening services while still creating denial-of-service or state-exhaustion risk if attackers can force expensive pre-authorisation work at scale.

Removing inbound exposure from endpoints does not eliminate denial-of-service risk; it shifts strategic pressure toward rendezvous and control-plane infrastructure. UZPIF therefore redistributes attack surface away from broad endpoint exposure and toward smaller sets of highly defended coordination infrastructure. UZPIF-compliant deployments MUST therefore treat RN availability as a first-class design concern rather than as an operational afterthought.

RNs are likely focal points for volumetric, state-exhaustion, and coordination-disruption attacks. Endpoint hardening alone does not solve network-wide availability. Deployments SHOULD assume deliberate RN-targeted overload attempts and SHOULD use plurality, geographic dispersion, bounded-state processing, rate controls, admission pacing, queue isolation, and failure isolation where practical. The availability of RN infrastructure also depends partly on underlying transport and routing ecosystems that are outside the protocol's direct control, so protocol and deployment design SHOULD minimise the consequences of temporary RN overload, routing impairment, or RN loss.

Expensive cryptographic verification, policy evaluation, federation lookups, or translation work SHOULD occur as late as safely possible in the processing path. Stateless screening, bounded-prestate admission checks, puzzles, quotas, or other low-cost filters SHOULD be preferred where practical before allocating scarce state or invoking high-cost validation.

RNs, HILs, and related control-plane services SHOULD rate-limit, queue-bound, or otherwise constrain untrusted requests before high-cost cryptographic or policy evaluation where feasible. Implementations SHOULD separate basic survivability controls from richer verification paths so that overload of one validation function does not automatically collapse unrelated session handling or local containment.

Basic survivability SHOULD NOT depend on a single always-available validation bottleneck such as one authority-resolution path, one transparency endpoint, or one high-cost policy service. Deployment profiles SHOULD define bounded-failure behaviour, degraded but observable operation, and local overload policy so that exhaustion pressure does not silently widen authority or force unsafe fallback.

## 16. IANA Considerations

This document has no IANA actions.

UZPIF is an architectural framework and does not define protocol parameters requiring registries.

## 17. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8785] Rundgren, A., Jordan, B., and S. Erdtman, "JSON Canonicalization Scheme (JCS)", RFC 8785, DOI 10.17487/RFC8785, June 2020, <<https://www.rfc-editor.org/info/rfc8785>>.

## 18. Informative References

- [ICSD] Fisher, B. A., "Invariant-Closed System Design (ICSD)", Work in Progress, Internet-Draft, draft-dpa-icsd, <<https://datatracker.ietf.org/doc/html/draft-dpa-icsd>>.
- [NIST-PQC] National Institute of Standards and Technology, "NIST Post-Quantum Cryptography Standardization Project", 2022, <<https://csrc.nist.gov/Projects/post-quantum-cryptography>>.
- [NIST-SP800-207] Rose, S., Borchert, O., Mitchell, S., and S. Connelly, "Zero Trust Architecture", NIST SP 800-207, 2019, <<https://doi.org/10.6028/NIST.SP.800-207>>.
- [OUTBOUND-INDEXING] Fisher, B. A., "Outbound Indexing over UZPIF: Consent-First Discovery and Indexing", Work in Progress, Internet-Draft, draft-dpa-uzpif-outbound-indexing, <<https://datatracker.ietf.org/doc/html/draft-dpa-uzpif-outbound-indexing>>.
- [RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, DOI 10.17487/RFC5389, October 2008, <<https://www.rfc-editor.org/info/rfc5389>>.

- [RFC5766] Mahy, R., Matthews, P., and J. Rosenberg, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)", RFC 5766, DOI 10.17487/RFC5766, April 2010, <<https://www.rfc-editor.org/info/rfc5766>>.
- [RFC7401] Moskowitz, R., Ed., Heer, T., Jokela, P., and T. Henderson, "Host Identity Protocol Version 2 (HIPv2)", RFC 7401, DOI 10.17487/RFC7401, April 2015, <<https://www.rfc-editor.org/info/rfc7401>>.
- [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.
- [RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <<https://www.rfc-editor.org/info/rfc9000>>.
- [TLS-DPA] Fisher, B. A., "TLS-DPA: An Identity-Bound Security Protocol for Traditional, Overlay, and Zero-Port Transports", Work in Progress, Internet-Draft, draft-dpa-tls-dpa, <<https://datatracker.ietf.org/doc/html/draft-dpa-tls-dpa>>.
- [Tor] Dingledine, R., Mathewson, N., and P. Syverson, "Tor: The Second-Generation Onion Router", 2004.
- [UZP] Fisher, B. A., "UZP: Universal Zero-Port Transport Protocol", Work in Progress, Internet-Draft, draft-dpa-uzp-transport, <<https://datatracker.ietf.org/doc/html/draft-dpa-uzp-transport>>.

## Author's Address

Benjamin Anthony Fisher  
DPA R&D Ltd (<https://www.dpa-cloud.co.uk>)  
Email: [b.fisher@dpa-cloud.co.uk](mailto:b.fisher@dpa-cloud.co.uk)  
URI: <https://orcid.org/0009-0004-4412-2269>