Network Working Group
Internet-Draft
Intended status: Informational
Expires: 15 November 2026

T. Litzki
Litzki Systems LLC

14 May 2026

Sovereign Validation Protocol (SOVP)
draft-litzki-sovp-01

Abstract

This document specifies the Sovereign Validation Protocol (SOVP), a protocol for cryptographic verification of publisher-provided identity metadata bound to DNS-anchored public keys. SOVP enables consuming agents and gateways to verify the origin and integrity of machine-readable data published at a domain prior to ingestion, allowing early rejection of unauthenticated data before body parsing occurs. The protocol defines data structures, cryptographic procedures, operational modes, and associated DNS and HTTP mechanisms.

Note

This document describes work by Litzki Systems LLC.

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 15 November 2026.

Copyright Notice

Table of Contents

1. Introduction

The increasing deployment of autonomous agents and large language model (LLM) pipelines that consume web-published data creates a need for infrastructure-level verification of data origin and integrity. Existing mechanisms such as TLS operate at the transport layer and do not verify the integrity of application-layer data after delivery. Application-level validation typically occurs after data has been parsed and partially processed, consuming resources before unauthenticated content can be rejected.

SOVP addresses this by enabling a publisher to bind machine-readable identity metadata to an Ed25519 signature, with the corresponding public key anchored in DNS. A consuming agent or gateway (referred to as a Validating Agent or SOVP Gateway) can verify this binding without prior relationship or shared secret, and reject non-conformant data at the ingestion boundary before body parsing occurs.

Layer 0, as used in this document, refers to the ingestion boundary: the point at which a consuming system decides whether to accept or reject data from an external source, prior to any application-level processing.

This document specifies the protocol data structures, cryptographic procedures, and operational modes constituting SOVP. It is published as an Informational document describing the current protocol specification.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Non-Goals

SOVP enables verification of origin and integrity for publisher-provided identity metadata. Specifically, a successful verification (Psi_core = 1) indicates that the sovp-identity.json retrieved from a domain was signed by the holder of the private key corresponding to the DNS-published public key, and that the signed content has not been modified since signing. SOVP does not validate the semantic accuracy or truthfulness of the content. It does not replace downstream fact-checking or content analysis. SOVP provides origin binding, not content truth.

3. Validation Function: Psi_core

The core of the protocol is the validation function Psi_core. It produces a binary result indicating whether the identity metadata retrieved from a domain was signed by the holder of the corresponding DNS-published public key. The identity metadata (M) MUST be canonicalized using the JSON Canonicalization Scheme (JCS) as defined in [RFC8785] before signature verification, to ensure consistent byte representation across implementations.

The implementation uses Ed25519 in pure mode per [RFC8032]. The canonicalized identity metadata is passed directly to the Ed25519 sign and verify functions. No external pre-hash is applied.

Psi_core = Verify(K_pub, sigma, JCS(M))

Where:

* K_pub is the public key, retrieved via a DNS TXT record or the SOVP-Identity header.

* sigma is the digital signature provided within the integrity_proof.

* JCS(M) is the canonicalized identity metadata per [RFC8785], passed directly to the Ed25519 verify function.

* Verify is the Ed25519 verification function, returning 1 if the signature is valid for the given key and message, or 0 otherwise.

4. Technical Specification: The sovp-identity.json Structure

The implementation relies on a signed JSON object located at the well-known path of the host (/.well-known/sovp-identity.json, per [RFC8615]). This object serves as the primary data carrier for the publisher identity declaration.

The signature covers the fields outside the integrity_proof object. The integrity_proof object itself, including the signature field, is excluded from the signed scope. Implementations MUST canonicalize only the non-proof fields of M when computing or verifying JCS(M).

Proposed Schema for sovp-identity.json:

```
{
  "@context": "https://litzki-systems.com/protocol/v1.4",
  "@type": "SovereignIdentity",
  "entity": {
    "uid": "urn:sovp:litzki-systems-llc",
    "canonical_url": "https://litzki.systems",
    "verification_method": "Ed25519"
  },
  "integrity_proof": {
    "signature": "z58D...v9A",
    "created": "2026-03-14T17:00:00Z",
    "public_key_ref": "dns:txt:_sovp.litzki.systems",
    "nonce": "optional-unique-string-123"
  },
  "parameters": {
    "entropy_threshold": 0.12,
    "determinism_score": 0.98
  }
}
```

The parameters object and its attributes entropy_threshold and determinism_score are non-normative and optional. They represent publisher-supplied advisory values and are not cryptographically bound by the Ed25519 signature. Validating Agents MUST NOT use these values as a basis for trust decisions. They MAY be used for informational or monitoring purposes only.

5. Signature Mismatch and Verification Failure

A verification failure (Psi_core = 0) occurs when the sovp-identity.json retrieved from a domain does not produce a valid signature verification result. This may indicate that the identity declaration has been modified since signing, that the wrong key was used, or that the document is malformed. A Validating Agent or SOVP Gateway MUST treat Psi_core = 0 as a rejection condition and MUST NOT proceed with ingestion of data from the non-conformant source.

6. Protocol Execution Sequence

The protocol execution follows a non-interactive sequence to compute Psi_core. SOVP defines two primary operational modes:

Mode	Actor	Description
Mode A	Validating Agent (Client)	The agent performs verification locally before committing data to memory.
Mode B	SOVP Gateway (Server)	An infrastructure-level gateway validates requests on behalf of a protected cluster.

Table 1

Execution Sequence:

1. Public Key Exposure: The entity publishes its Ed25519 public key via a DNS TXT record (typically at _sovp.yourdomain.tld) or via the SOVP-Identity HTTP header.

2. Artifact Retrieval: The Validating Agent or Gateway retrieves the sovp-identity.json from the host’s well-known path (/.well-known/sovp-identity.json).

3. Integrity Verification: The actor canonicalizes the payload according to [RFC8785] and executes the Verify function using Ed25519 pure mode per [RFC8032].

4. Verification Result: Ingestion proceeds if Psi_core = 1. If Psi_core = 0, the Validating Agent or SOVP Gateway MUST reject the data and MUST NOT commit it to the processing pipeline.

6.1. Mode B Rejection Policy

If a source does not provide a valid sovp-identity.json, or if Psi_core = 0, the SOVP Gateway MUST reject the request. The RECOMMENDED HTTP status code for rejection is 403 (Forbidden). Implementations MAY use 422 (Unprocessable Content) where the rejection is specifically due to a malformed or invalid SOVP identity document. Rejection SHOULD occur before body parsing to avoid unnecessary resource consumption.

* Exception: Local allow-lists MAY be defined for legacy systems, though these bypass the Layer 0 integrity guarantee.

7. Security Considerations

This section addresses security considerations per [RFC3552].

7.1. Key Revocation and DNS TTL

To minimize the window of vulnerability during a key compromise, SOVP records in DNS SHOULD use a low Time-To-Live (TTL), with a recommended value of 300 seconds. Revocation is achieved by updating or removing the _sovp TXT record.

7.2. Replay Protection

All integrity_proof objects MUST contain a created timestamp. Validating Agents SHOULD reject signatures with a timestamp older than 600 seconds. For high-frequency pipelines, agents SHOULD implement nonce-based deduplication within the validity window to prevent replayed requests. Note: timestamp and nonce validation are not yet implemented in the reference implementation and are planned for a future release.

7.3. Origin Binding

SOVP binds the signed identity metadata to the Ed25519 key pair whose public component is published in DNS. A successful verification (Psi_core = 1) demonstrates that the holder of the private key signed the identity document. This enables attribution of the signed content to the DNS-identified publisher. SOVP does not provide legal non-repudiation and does not establish the real-world identity of the key holder beyond DNS control.

7.4. DNS Security (DNSSEC)

Since the trust anchor relies on DNS TXT records, protection against DNS spoofing is critical. The use of DNSSEC is RECOMMENDED for the zone hosting the _sovp record to ensure the authenticity of the public key.

8. Deployment Architecture: Ingestion Boundary Positioning

SOVP is designed to operate at the ingestion boundary (Layer 0), prior to application-level processing. In Mode B, the SOVP Gateway intercepts incoming data requests and performs verification before forwarding to backend systems. This positions verification upstream of resource-intensive processing, allowing unauthenticated data to be rejected early. DNS serves as the trust anchor for public key distribution, with DNSSEC recommended to protect against DNS-based attacks on key material.

9. IANA Considerations

9.1. Underscored and Globally Scoped DNS Node Names

Per [RFC8552], the following entry is to be added to the IANA registry:

* Label: _sovp
* Protocol: SOVP
* Reference: This document (Section 9.1)

9.2. Permanent Message Header Field Names

Per [RFC3864] and [RFC6648], the following header is to be registered. The SOVP-Identity header name does not use the deprecated "X-" prefix per [RFC6648]:

* Header field name: SOVP-Identity
* Applicable protocol: HTTP
* Status: Experimental
* Author/Change controller: Litzki Systems LLC
* Specification document: This document (Section 9.2)

10. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, 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, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8552] Crocker, S., "Scoped DNS Node Names", RFC 8552, March 2019, <https://www.rfc-editor.org/info/rfc8552>.

[RFC3864] Klyne, G., "Registration Procedures for Message Header Fields", RFC 3864, September 2004, <https://www.rfc-editor.org/info/rfc3864>.

[RFC8785] Rundgren, A., "JSON Canonicalization Scheme (JCS)", RFC 8785, June 2020, <https://www.rfc-editor.org/info/rfc8785>.

[RFC8615] Nottingham, M., "Well-Known Uniform Resource Identifiers (URIs)", RFC 8615, May 2019, <https://www.rfc-editor.org/info/rfc8615>.

[RFC8032] Josefsson, S. and I. Liusvaara, "Edwards-Curve Digital Signature Algorithm (EdDSA)", RFC 8032, January 2017, <https://www.rfc-editor.org/info/rfc8032>.

[RFC6648] Saint-Andre, P., Crocker, D., and M. Nottingham, "Deprecating the "X-" Prefix and Similar Constructs in Application Protocols", BCP 178, RFC 6648, June 2012, <https://www.rfc-editor.org/info/rfc6648>.

[RFC3552] Rescorla, E., "Guidelines for Writing RFC Text on Security Considerations", RFC 3552, July 2003, <https://www.rfc-editor.org/info/rfc3552>.

Author’s Address

Thorsten Litzki
Litzki Systems LLC
7901 4th St N, #32272
St. Petersburg, FL 33702
United States of America
Email: ietf@litzki-systems.com