Individual Submission
Internet-Draft
Intended status: Standards Track
Expires: 22 October 2026

L. Nyakiso
RuleMesh

20 April 2026

HTTP Compliance Authorization Protocol (HCAP)
draft-nyakiso-hcap-00

Abstract

This document specifies the HTTP Compliance Authorization Protocol (HCAP), an extension to the HTTP authentication framework that allows resource providers to require demonstrable evidence of caller compliance with declared policies before granting access to protected resources. Callers obtain signed compliance credentials from a compliance registry by presenting evidence of their controls. Credentials are conveyed at the HTTP layer and verified offline by the provider.

HCAP operates alongside existing authentication schemes (OAuth 2.0, mTLS) and does not replace identity authentication. It addresses the distinct question of "does the caller meet declared policy requirements" rather than "who is the caller".

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 22 October 2026.

Copyright Notice

Table of Contents

1. Introduction

1.1. Motivation

Compliance verification between API providers and API callers is today conducted almost entirely out-of-band: vendor questionnaires, bilateral Data Processing Agreements, emailed SOC 2 reports, annual audits, and security reviews. Verification happens at contract time, not at request time. The evidence is prose, not machine-readable. The result is that every API integration requiring regulatory or contractual compliance carries substantial per-pair overhead, and a provider has no runtime signal that a specific caller actually meets the policies they claimed to meet when the relationship was established.

HCAP moves compliance verification to the HTTP layer. A provider declares, per endpoint, what ruleset a caller MUST satisfy. A caller obtains a signed Compliance Credential from a Compliance Registry by presenting evidence of the required controls. The caller then presents the credential on subsequent requests. The provider verifies the credential’s signature using the Registry’s published trust anchors and checks that the credential’s claims satisfy the declared ruleset.

This allows compliance to be:

* Declarative: Providers publish their requirements in a machine-readable form rather than distributing prose policies.

* Portable: A caller’s credential is valid with any provider that trusts the same Registry and ruleset.

* Runtime-verifiable: Verification occurs on each request, not once at contract signing.

* Offline-verifiable: Trust anchor distribution allows signature verification without per-request calls to the Registry.

1.2. Non-Goals

This specification does not:

* Define what constitutes acceptable evidence for any specific ruleset.

* Mandate a specific evidence schema, compliance ontology, or legal framework.

* Govern liability, enforceability, or regulator recognition of Compliance Credentials.

* Replace identity authentication. A caller MUST still authenticate using an existing scheme.

1.3. Relationship to Existing Specifications

HCAP extends [RFC9110] by defining a new HTTP Authentication Scheme. It reuses [RFC7519] JSON Web Tokens for credential encoding, following [RFC8725] best current practice. Credential semantics align with the W3C Verifiable Credentials Data Model [VC-DATA-MODEL]. Trust anchors are published as JSON Web Key Sets [RFC7517]. Well-known URIs follow [RFC8615].

HCAP is complementary to OAuth 2.0 [RFC6749] and Rich Authorization Requests [RFC9396], which govern what a principal is authorized to do. HCAP governs what policy requirements the principal’s operating environment satisfies.

1.4. Protocol Scope and Registry Governance

This document specifies the wire protocol, data formats, and verification procedures for HCAP. It deliberately does not address the operational, procedural, or governance requirements for entities operating as Compliance Registries.

How a Registry establishes the truth of the facts it attests to is a governance problem of substantially different character from the protocol problem, and conflating the two would impair adoption of either. Registry operational requirements are expected to be addressed in separate documents, which may take the form of industry operational baselines (analogous to the CA/Browser Forum Baseline Requirements [CABF-BR]), regulator-recognized accreditation schemes (such as the eIDAS qualified trust service provider framework [EIDAS]), or domain-specific profiles.

Implementations that adopt this specification remain free to accept credentials from any Registry whose trust anchors they choose to trust, and to apply any additional operational criteria beyond those defined here.

2. Conventions and Terminology

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.

Caller: The HTTP client attempting to access a protected resource.

Provider: The HTTP origin server that protects the resource.

Registry: A trust service that issues Compliance Credentials on behalf of Callers. This specification does not privilege any particular Registry operator. Any entity whose signing keys are published as a trust anchor (Section 4) and whose credentials meet the verification requirements of Section 9 MAY operate as a Registry. Registry operational obligations are out of scope for this document (Section 1.4).

Ruleset: A named, versioned set of compliance requirements, identified by a URI and described by a Ruleset Manifest (Section 4.2).

Claim: An individual requirement within a Ruleset, identified by a short string (e.g., art32, dpa, encryption_at_rest).

Compliance Credential: A signed attestation by a Registry that a specific Caller satisfies a specific set of Claims against a specific Ruleset at a specific time.

Presentation: A Compliance Credential as it appears on an HTTP

request, transmitted via the Compliance-Presentation header (Section 7).

Trust Anchor: A Registry's public signing key, published as a JSON Web Key Set and retrievable via a jwks_uri.

Evidence Tier: A Registry-declared assurance level for the evidence underlying a Compliance Credential. Defined tiers are discussed in Section 8.2.

3. Protocol Overview

The HCAP flow has three participants: the Caller, the Registry, and the Provider. A typical interaction proceeds as follows:

1. The Caller sends a request to the Provider's protected resource.
2. The Provider responds with 401 Unauthorized and a WWW-Authenticate: Compliance challenge naming the required Ruleset and Claims.
3. The Caller obtains the Ruleset Manifest (typically cached from a prior interaction) and presents evidence of the required controls to the Registry.
4. The Registry issues a signed Compliance Credential to the Caller.
5. The Caller retries the original request with the credential in the Compliance-Presentation header.
6. The Provider verifies the credential's signature using cached trust anchors and checks that its Claims satisfy the declared Ruleset.
7. The Provider serves the protected resource.

Steps 3 and 4 are typically executed ahead of time and the resulting credential is reused until it expires. The Caller SHOULD NOT perform a Registry round-trip on every protected request.

4. Ruleset Discovery

4.1. Well-Known URI

A Provider implementing HCAP SHOULD publish a Ruleset Manifest at the well-known URI /.well-known/compliance, registered per [RFC8615]. A Provider MAY publish multiple manifests at distinct paths under this namespace, e.g., /.well-known/compliance/customers.

A Provider MAY reference a specific manifest from a challenge using the ruleset parameter (Section 5.3). When no explicit URI is given, the Caller resolves the manifest from the challenge's realm and the default well-known path.

4.2. Ruleset Manifest

A Ruleset Manifest is a JSON document with the media type application/compliance-manifest+json. The manifest MUST contain the following fields:

* ruleset_id: URI, the canonical identifier for this ruleset.
* version: Semantic version string.
* authority: URI or DID identifying the issuer of the ruleset definition (distinct from the Registry that signs credentials against it).
* claims: An array of Claim definitions, each with id, description, and OPTIONAL references to external normative texts (e.g., specific GDPR articles).
* trust_anchors: An array of jwks_uri values identifying Registries whose credentials the Provider will accept.
* endpoints: An array of endpoint rules (see Section 4.3).

The manifest MAY include accepted_equivalents, an array of other Ruleset URIs whose credentials the Provider will also accept for this Ruleset. Equivalence is always declared by the Provider; a Caller MUST NOT assume two Rulesets are equivalent absent such a declaration.

4.3. Endpoint Rules

Each entry in endpoints declares which requests require which Claims:

* path_pattern: String, using the pattern syntax of [RFC6570].
* methods: Array of HTTP methods this rule applies to.
* required_claims: Array of Claim IDs from this Ruleset.
* required_evidence_tier: OPTIONAL minimum Evidence Tier.

When multiple endpoint rules match a request, the Provider MUST require the union of their Claims.

4.4. Caching

Ruleset Manifests SHOULD be served with Cache-Control and ETag headers. Callers SHOULD cache manifests for the duration indicated by max-age. When no directive is present, Callers MAY cache for up to 24 hours.

5. The Compliance Challenge

5.1. Status Code

A Provider MUST return 401 Unauthorized when a request requires a Compliance Credential and no valid Compliance-Presentation is present.

A Provider MUST return 403 Forbidden when a valid Presentation is provided but it does not satisfy the required Ruleset.

5.2. The Compliance Authentication Scheme

HCAP defines a new HTTP Authentication Scheme named Compliance. A challenge has the form:

```
WWW-Authenticate: Compliance realm="api.example.com",
    ruleset="https://rules.example.com/gdpr-processor/v2",
    claims="art28 art32 dpa",
    trust_anchors="https://trust.example.net/.well-known/jwks.json",
    max_age=3600,
    error="compliance_required"
```

A Provider MAY include a Compliance challenge alongside other authentication challenges (e.g., Bearer). In this case the Caller MUST satisfy all required schemes.

5.3. Parameters

The Compliance scheme defines the following parameters:

* realm (REQUIRED): As defined in [RFC9110].

* ruleset (REQUIRED): URI identifying the required Ruleset.

* claims (OPTIONAL): Space-separated list of Claim IDs overriding the Ruleset manifest’s endpoint rules for this response.

* trust_anchors (OPTIONAL): Space-separated list of jwks_uri values identifying acceptable Registries. If omitted, the Provider accepts any Registry listed in the Ruleset Manifest.

* max_age (OPTIONAL): Maximum credential age in seconds that the Provider will accept, measured from the credential's iat.

* error (OPTIONAL): Error code from Section 10.

* error_description (OPTIONAL): Human-readable error detail. MUST NOT be relied upon by automated clients.

6. Credential Acquisition

The protocol between a Caller and a Registry is out of scope for this specification. Registries MAY define their own acquisition APIs. This section describes the minimum contract that any Registry protocol MUST satisfy.

6.1. Inputs

A Caller supplies, at minimum:

* A Caller identifier (typically the same sub used in the Caller's identity authentication against the Provider).

* The target Ruleset URI.

* Evidence sufficient to establish each required Claim. The form of evidence is determined by the Registry and MAY include self-attestations, signed statements from authorized personnel, third-party audit reports, cryptographic attestations of system state, or other artifacts.

6.2. Outputs

On successful evaluation, the Registry issues a Compliance Credential (Section 8). The Registry MUST NOT issue credentials for Claims whose evidence it has not evaluated.

6.3. Lifetime

A Registry SHOULD issue credentials with the shortest lifetime operationally practical. A lifetime of 1 hour is RECOMMENDED for typical deployments. Credentials MUST NOT have a lifetime exceeding 24 hours unless the Registry also publishes a revocation status endpoint (Section 9.4).

7. Credential Presentation

7.1. The Compliance-Presentation Header

A Caller presents a Compliance Credential by including the Compliance-Presentation header field on the request:

Compliance-Presentation: <token>

The <token> value is the JWT encoding of the credential as defined in Section 8.

The Compliance-Presentation header is independent of the Authorization header. A request MAY include both, and typically does: Authorization conveys identity; Compliance-Presentation conveys policy satisfaction.

7.2. Multiple Credentials

A Caller MAY present multiple credentials on a single request when the required Claims span multiple Rulesets or Registries. In this case the header value is a comma-separated list of tokens:

Compliance-Presentation: <token1>, <token2>

The Provider MUST consider the union of claims_satisfied across all valid credentials when evaluating the request.

7.3. Caching

Callers SHOULD cache credentials and reuse them across requests until expiry. Callers SHOULD NOT include credentials on requests to origins for which no challenge has been received.

8. Credential Format

8.1. Encoding

A Compliance Credential is encoded as a JSON Web Token per [RFC7519], signed using an algorithm from [RFC7518]. Registries MUST use an asymmetric signature algorithm; EdDSA and ES256 are RECOMMENDED.

The credential payload MAY be expressed as a Verifiable Credential [VC-DATA-MODEL] using the JWT binding, with a vc claim containing the VC document. Implementations that do not require W3C VC interoperability MAY omit the vc claim and use the flat JWT claim set defined below.

8.2. Claim Set

A Compliance Credential MUST contain the following claims:

* iss: URI identifying the Registry.

* sub: Caller identifier. Providers MUST verify this matches the subject of the Caller's identity authentication.

* aud: Ruleset URI, or an array including the Ruleset URI and OPTIONAL Provider identifier.

* iat: Issuance time (seconds since epoch).

* exp: Expiration time.

* jti: Unique credential identifier. Used for revocation tracking.

* ruleset: Ruleset URI this credential satisfies.

* claims_satisfied: Array of Claim IDs the Registry attests the Caller meets.

A Compliance Credential MAY contain:

* evidence_tier: String indicating evidence strength. Defined values:

  - self_attested: Caller-signed statement, no Registry verification of underlying facts.

  - attested_by_officer: Statement signed by a named authorized individual at the Caller (e.g., DPO, CISO).

  - third_party_audit: Underlying evidence is a dated attestation by an independent auditor.

  - cryptographic_proof: Underlying evidence is a cryptographic attestation of system state (e.g., TPM quote, TEE report).

* status: URI of a Status List [STATUS-LIST] entry for revocation.

* cnf: Confirmation claim [RFC7800] binding the credential to a key held by the Caller, for use with proof-of-possession.

8.3. Example

A decoded credential payload:

```
{
  "iss": "https://registry.example.net",
  "sub": "client_abc123",
  "aud": [
    "https://rules.example.com/gdpr-processor/v2",
    "https://api.example.com"
  ],
  "iat": 1713024000,
  "exp": 1713027600,
  "jti": "cred_7a3d91f0e2",
  "ruleset": "https://rules.example.com/gdpr-processor/v2",
  "claims_satisfied": ["art28", "art32", "dpa"],
  "evidence_tier": "third_party_audit",
  "status": "https://registry.example.net/status/12#5142"
}
```

9. Verification Procedure

On receiving a request with a Compliance-Presentation header, a Provider MUST perform the following checks, in order. If any check fails, the Provider MUST respond with 403 Forbidden and an appropriate error code (Section 10).

9.1. Signature

1. Parse the JWT.

2. Resolve the signing key from the Provider’s cached trust anchors for the iss claim, refreshing from jwks_uri if the key ID is unknown.

3. Verify the signature.

A Provider MUST NOT fetch a new jwks_uri from an iss not previously declared in its Ruleset Manifest.

9.2. Temporal Validity

1. Verify exp has not passed, allowing a clock skew of at most 60 seconds.

2. Verify iat is not in the future beyond the same skew.

3. Verify the credential age (now - iat) does not exceed any max_age declared in the challenge.

9.3. Binding

1. Verify sub matches the subject of the Caller's identity authentication. Providers MUST reject credentials whose sub does not match.

2. If cnf is present, verify proof-of-possession per [RFC9449] or equivalent.

3. Verify aud includes the Ruleset URI required by this endpoint.

9.4. Revocation

If status is present, the Provider SHOULD consult the status list per [STATUS-LIST]. Providers MAY cache status list results for up to the credential's remaining lifetime, but SHOULD refresh on cache-miss.

9.5. Ruleset Satisfaction

1. Determine the required Ruleset and Claims for the endpoint, per Section 4.

2. Compute the set of Claims satisfied by the union of all valid credentials on the request.

3. Verify this set is a superset of the required Claims.

4. If the endpoint specifies required_evidence_tier, verify that each required Claim is satisfied by at least one credential of sufficient tier.

9.6. Equivalence

If no credential directly satisfies the required Ruleset, the Provider MAY attempt to satisfy it via declared accepted_equivalents. Equivalence MUST NOT be inferred; the Provider's own manifest is authoritative.

10. Error Handling

Errors are reported via the error parameter of the WWW-Authenticate challenge, accompanying a 401 or 403 response. The following error codes are defined:

* compliance_required: No presentation on a protected request.

* invalid_credential: Signature invalid, malformed JWT, unknown algorithm.

* expired_credential: exp has passed, or credential age exceeds max_age.

* revoked_credential: Status list indicates revocation.

* insufficient_claims: Valid credential, but required Claims not met.

* insufficient_evidence_tier: Claim satisfied but tier too low.

* unsupported_ruleset: Credential cites a Ruleset the Provider does not recognize.

* trust_anchor_unknown: iss is not among declared Registries.

* subject_mismatch: sub does not match authenticated identity.

Additional error codes MAY be defined by future specifications or extensions, following the IANA registration procedure in Section 13.

11. Security Considerations

11.1. Credential Theft and Replay

A Compliance Credential is a bearer token in the absence of proof-of-possession. An attacker who obtains a credential can replay it until expiry. Mitigations:

* Short lifetimes (Section 6.3) limit replay windows.

* Subject binding (Section 9.3) ensures a stolen credential cannot be used by a different identity if identity authentication is sound.

* Providers MAY require cnf proof-of-possession for high-risk endpoints.

* Transport MUST be TLS 1.2 or later [RFC8446].

11.2. Registry Compromise

A compromised Registry can issue fraudulent credentials. Defenses:

* Providers SHOULD pin specific Registry keys or operate against short, published key-rotation schedules.

* Registries SHOULD publish signing-key transparency logs where feasible.

* A Provider MAY operate a local override list of Registries pending investigation.

11.3. Downgrade Attacks

* A Provider MUST NOT serve a protected resource on receipt of a request that lacks a valid Presentation when its own policy requires one.

* Providers MUST NOT accept Compliance-Presentation over plaintext HTTP.

* Intermediaries MUST NOT modify, cache, or forward Compliance-Presentation headers.

11.4. Signature Algorithm Confusion

Implementations MUST follow [RFC8725] and MUST NOT accept the none algorithm. Implementations MUST verify the algorithm used against an allow-list declared in the trust anchor JWKS.

11.5. Clock Skew

Verifiers MUST allow no more than 60 seconds of clock skew on temporal checks. Registries SHOULD use NTP-synchronized time.

12. Privacy Considerations

A Compliance Credential reveals that a Caller has satisfied specific Claims with a specific Registry. This may inadvertently disclose:

* The Caller’s choice of Registry, which may correlate with the Caller’s jurisdiction or industry.

* The Caller’s compliance posture at a point in time.

* Via evidence_tier, aspects of the Caller’s audit practices.

Registries SHOULD support selective disclosure, including issuing credentials that commit to a set of Claims without revealing which subset is invoked on any given request. BBS+ signatures and related schemes building on [VC-DATA-MODEL] are candidates for this mode.

Providers MUST NOT log Compliance-Presentation header values in cleartext. The credential’s jti MAY be logged for audit purposes.

13. IANA Considerations

This document requests the following registrations. All registrations are subject to IETF Review.

13.1. HTTP Authentication Scheme

Registration in the "Hypertext Transfer Protocol (HTTP) Authentication Scheme Registry":

* Scheme Name: Compliance
* Reference: This document.

13.2. HTTP Header Field

Registration in the "Hypertext Transfer Protocol (HTTP) Field Name Registry":

* Field Name: Compliance-Presentation
* Status: Standard
* Reference: This document (Section 7)
* Comments: Request-only.

13.3. Well-Known URI

Registration in the "Well-Known URIs" registry:

* URI Suffix: compliance
* Change Controller: IETF
* Reference: This document (Section 4)

13.4. Media Type

Registration in the "Media Types" registry:

* Type Name: application
* Subtype Name: compliance-manifest+json
* Reference: This document (Section 4.2)

13.5. HCAP Error Code Registry

Establishment of a new "HCAP Error Codes" registry with the values defined in Section 10. Registration policy: Specification Required.

13.6. HCAP Evidence Tier Registry

Establishment of a new "HCAP Evidence Tiers" registry with the values defined in Section 8.2. Registration policy: Specification Required.

14. References

14.1. 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/rfc/rfc2119>.

[RFC7517] Jones, M., "JSON Web Key (JWK)", RFC 7517, DOI 10.17487/RFC7517, May 2015, <https://www.rfc-editor.org/rfc/rfc7517>.

[RFC7518] Jones, M., "JSON Web Algorithms (JWA)", RFC 7518, DOI 10.17487/RFC7518, May 2015, <https://www.rfc-editor.org/rfc/rfc7518>.

[RFC7519] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", RFC 7519, DOI 10.17487/RFC7519, May 2015, <https://www.rfc-editor.org/rfc/rfc7519>.

[RFC7800] Jones, M., Bradley, J., and H. Tschofenig, "Proof-of-Possession Key Semantics for JSON Web Tokens (JWTs)", RFC 7800, DOI 10.17487/RFC7800, April 2016, <https://www.rfc-editor.org/rfc/rfc7800>.

[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/rfc/rfc8174>.

[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/rfc/rfc8446>.

[RFC8615] Nottingham, M., "Well-Known Uniform Resource Identifiers (URIs)", RFC 8615, DOI 10.17487/RFC8615, May 2019, <https://www.rfc-editor.org/rfc/rfc8615>.

[RFC8725] Sheffer, Y., Hardt, D., and M. Jones, "JSON Web Token Best Current Practices", BCP 225, RFC 8725, DOI 10.17487/RFC8725, February 2020, <https://www.rfc-editor.org/rfc/rfc8725>.

[RFC9110] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP Semantics", STD 97, RFC 9110, DOI 10.17487/RFC9110, June 2022, <https://www.rfc-editor.org/rfc/rfc9110>.

[VC-DATA-MODEL] Sporny, M., Longley, D., and G. Cohen, "Verifiable Credentials Data Model v2.0", 2025, <https://www.w3.org/TR/vc-data-model-2.0/>.

14.2. Informative References

[CABF-BR] CA/Browser Forum, "Baseline Requirements for the Issuance and Management of Publicly-Trusted Certificates", n.d., <https://cabforum.org/baseline-requirements-documents/>.

[EIDAS] European Union, "Regulation (EU) No 910/2014 on electronic identification and trust services", 2014, <https://eur-lex.europa.eu/eli/reg/2014/910/oj>.

[RFC6570] Gregorio, J., Fielding, R., Hadley, M., Nottingham, M., and D. Orchard, "URI Template", RFC 6570, DOI 10.17487/RFC6570, March 2012, <https://www.rfc-editor.org/rfc/rfc6570>.

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, DOI 10.17487/RFC6749, October 2012, <https://www.rfc-editor.org/rfc/rfc6749>.

[RFC9396] Lodderstedt, T., Richer, J., and B. Campbell, "OAuth 2.0 Rich Authorization Requests", RFC 9396, DOI 10.17487/RFC9396, May 2023, <https://www.rfc-editor.org/rfc/rfc9396>.

[RFC9449] Fett, D., Campbell, B., Bradley, J., Lodderstedt, T., Jones, M., and D. Waite, "OAuth 2.0 Demonstrating Proof of Possession (DPoP)", RFC 9449, DOI 10.17487/RFC9449, September 2023, <https://www.rfc-editor.org/rfc/rfc9449>.

[STATUS-LIST] Looker, T. and P. Bastian, "Token Status List", n.d., <https://datatracker.ietf.org/doc/draft-ietf-oauth-status-list/>.

Appendix A. Examples

A.1. Full Handshake

Initial request:

```
GET /customers/42 HTTP/1.1
Host: api.example.com
Authorization: Bearer eyJhbGc...
```

Challenge:

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Compliance realm="api.example.com",
    ruleset="https://rules.example.com/gdpr-processor/v2",
    claims="art28 art32 dpa",
    trust_anchors="https://trust.example.net/.well-known/jwks.json",
    max_age=3600,
    error="compliance_required"
Link: </.well-known/compliance>; rel="compliance-requirements"
```

Caller fetches manifest (cached after first retrieval):

```
GET /.well-known/compliance HTTP/1.1
Host: api.example.com
Accept: application/compliance-manifest+json
```

Caller presents evidence to Registry (out-of-band protocol) and receives a Compliance Credential.

Retry with Presentation:

```
GET /customers/42 HTTP/1.1
Host: api.example.com
Authorization: Bearer eyJhbGc...
Compliance-Presentation: eyJhbGc...
```

Success:

```
HTTP/1.1 200 OK
Content-Type: application/json

{ "id": 42, "name": "..." }
```

A.2. Ruleset Manifest

```
{
  "ruleset_id": "https://rules.example.com/gdpr-processor/v2",
  "version": "2.1.0",
  "authority": "did:web:rules.example.com",
  "claims": [
    {
      "id": "art28",
      "description": "Processor agreement in place",
      "references": [
        "https://eur-lex.europa.eu/eli/reg/2016/679#art28"
      ]
    },
    {
      "id": "art32",
      "description": "Technical and organizational measures",
      "references": [
        "https://eur-lex.europa.eu/eli/reg/2016/679#art32"
      ]
    },
    {
      "id": "dpa",
      "description": "Signed data processing agreement on file"
    }
  ],
  "trust_anchors": [
    "https://trust.example.net/.well-known/jwks.json"
  ],
  "endpoints": [
    {
      "path_pattern": "/customers/{id}",
      "methods": ["GET", "PATCH"],
      "required_claims": ["art28", "art32"],
      "required_evidence_tier": "attested_by_officer"
    },
    {
      "path_pattern": "/customers/{id}/pii",
      "methods": ["GET"],
      "required_claims": ["art28", "art32", "dpa"],
      "required_evidence_tier": "third_party_audit"
    }
  ],
  "accepted_equivalents": [
    "https://rules.example.com/ccpa-service-provider/v1"
  ]
}
```

Appendix B. Open Questions

The following questions are explicitly unresolved in this draft and invite community input:

1. Registry governance. The truthfulness of the facts a Registry attests to, the operational requirements Registries must meet, and the trust framework by which Providers decide which Registries to include in their trust anchors are out of scope for this specification (Section 1.4). A companion document or industry operational baseline is expected to address these concerns separately. Candidate models include the CA/Browser Forum Baseline Requirements [CABF-BR], the eIDAS qualified trust service provider framework [EIDAS], and domain-specific accreditation schemes.

2. Equivalence authority. Placing equivalence authority with the Provider is one option. An alternative model assigns equivalence assertions to a third-party authority (e.g., a regulator or a neutral standards body). Both models have merits; the current draft prefers provider autonomy.

3. Revocation granularity. The current design supports credential-level revocation via status lists. It does not support claim-level revocation within a credential. Whether this is needed in practice is open.

4. Selective disclosure. Whether to profile BBS+ signatures, SD-JWT, or both is unresolved.

5. Request-scoped claims. Some policies require binding between a credential and specifics of a request (e.g., "the caller may process data for EU subjects only"). The current draft does not express such binding. An extension mechanism via cnf-style request-bound confirmation claims is a candidate.

Acknowledgements

The author thanks early reviewers and design partners for their feedback on the protocol shape and on the separation between protocol and Registry-governance concerns. Specific acknowledgements to be added upon submission.

Author’s Address

Lawrance Nyakiso
RuleMesh
Stockholm
Sweden
Email: lawrance.ny@rulemesh.com