

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

R. Anders  
RegisteredBrands.AI  
15 April 2026

Merchant Identity Assertions for Autonomous Commerce  
draft-anders-merchant-identity-assertions-00

## Abstract

Existing work helps a relying party determine whether an automated client is authorized to initiate a transaction. This document addresses how that client can obtain verifiable merchant identity information about the payee before completing the transaction.

Specifically, this document defines a Merchant Identity Assertion (MIA): a JSON document that attests to the verifiable identity of a merchant entity and carries an embedded cryptographic proof. It defines the assertion object structure, required and optional fields, the proof object, key discovery via well-known URIs, validity semantics, and an optional signed Evaluation Result Token for pre-transaction verification.

This document is intended to complement, not replace, existing agent identity and payment authorization protocols.

## 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 October 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. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

## Table of Contents

1. Introduction . . . . .	3
2. Terminology . . . . .	4
3. Problem Statement . . . . .	5
3.1. Merchant Impersonation . . . . .	5
3.2. Absent Audit Trail . . . . .	5
3.3. Policy Enforcement Gap . . . . .	5
3.4. Interoperability Absence . . . . .	5
4. Design Goals . . . . .	6
5. Merchant Identity Assertion . . . . .	6
5.1. Claim Fields (Required) . . . . .	6
5.2. Claim Fields (Optional) . . . . .	7
5.3. Proof Object . . . . .	8
5.4. Wire Format . . . . .	8
5.5. Example . . . . .	9
6. Discovery . . . . .	9
6.1. Well-Known URI . . . . .	9
6.2. Third-Party Issuer Authorization . . . . .	10
7. Signing and Verification . . . . .	11
7.1. Signing Algorithm . . . . .	11
7.2. Key Directory . . . . .	11
7.3. Signing Procedure . . . . .	11
7.4. Verification Procedure . . . . .	12
8. Freshness and Revocation . . . . .	13
9. Evaluation Result Token . . . . .	13
9.1. Claims . . . . .	14
9.2. Signing . . . . .	14
9.3. Example . . . . .	15
10. Privacy Considerations . . . . .	15
10.1. Minimal Disclosure . . . . .	15
10.2. Evidence URI Privacy . . . . .	15
10.3. Relying Party Query Privacy . . . . .	15
10.4. Consumer Principal Privacy . . . . .	16
11. Security Considerations . . . . .	16
11.1. Replay Attacks . . . . .	16
11.2. Key Compromise . . . . .	16
11.3. Domain Transfer . . . . .	16
11.4. Malicious Issuers . . . . .	16

11.5.	Stale Assertions . . . . .	17
11.6.	Downgrade Attacks . . . . .	17
11.7.	Third-Party Authorization Poisoning . . . . .	17
11.8.	JWT Security . . . . .	17
12.	IANA Considerations . . . . .	17
12.1.	Well-Known URI: merchant-identity.json . . . . .	17
12.2.	Well-Known URI: mia-delegation.json . . . . .	18
12.3.	Media Type: application/merchant-identity+json . . . . .	18
13.	References . . . . .	19
13.1.	Normative References . . . . .	19
13.2.	Informative References . . . . .	20
	Acknowledgements . . . . .	21
	Author's Address . . . . .	21

## 1. Introduction

The rapid growth of autonomous AI agents executing financial transactions on behalf of human principals has created a new class of trust requirements that existing protocols do not fully address.

Current work in agent identity -- including IETF Web Bot Auth [WEBBOTAUTH] and payment authorization frameworks such as Visa Trusted Agent Protocol, Mastercard Agent Pay, and the KYAPay Profile [KYAPAY] -- provides mechanisms for a receiving system to verify that an automated client is legitimate and authorized to transact. HTTP Message Signatures [RFC9421] provide the cryptographic foundation on which several of these frameworks build.

These mechanisms answer: "Is this agent who it claims to be, and is it authorized to pay?"

A complementary question remains unanswered by existing standards: "Is the merchant this agent intends to pay who it claims to be?"

In human-mediated commerce, consumers evaluate merchant identity through visual cues, brand recognition, and accumulated reputation. Autonomous agents lack these faculties. Without a machine-readable, cryptographically verifiable merchant identity mechanism, agents must either trust merchant identity claims implicitly -- creating fraud exposure -- or implement proprietary verification mechanisms that are not interoperable.

This document defines the Merchant Identity Assertion (MIA): a minimal, interoperable signed JSON document that enables any autonomous agent to verify merchant identity before executing a transaction.

The MIA is designed to:

- \* Be self-hostable by any merchant at a well-known URI
- \* Be verifiable by any agent without central registry dependency
- \* Complement existing agent identity protocols without replacing or competing with them
- \* Support independent implementations by any party

This document does not define trust scoring algorithms, merchant ranking systems, payment authorization protocols, or agent identity mechanisms. Those concerns are addressed by other specifications.

## 2. 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.

**Merchant:** A legal entity offering goods or services in exchange for payment, identified by a domain name.

**Merchant Identity Assertion (MIA):** A signed JSON document attesting to the verifiable identity of a merchant entity, as defined in Section 5.

**Claims Object:** The JSON object containing the identity fields of an MIA, excluding the proof. Used as the signing input.

**Proof Object:** The JSON object embedded in an MIA that carries the cryptographic signature and key reference, as defined in Section 5.3.

**Issuer:** An entity that signs and publishes a Merchant Identity Assertion. The issuer MAY be the merchant itself or a third party acting on the merchant's behalf with explicit authorization per Section 6.2.

**Relying Party:** An autonomous agent or system that consumes a Merchant Identity Assertion to make a pre-transaction verification decision.

**Subject Domain:** The fully qualified domain name (FQDN) to which the Merchant Identity Assertion applies.

**Evaluation Result Token (ERT):** A signed JWT [RFC7519] issued by a

verification service summarizing the result of a pre-transaction merchant identity check, as defined in Section 9.

Key Directory: A JSON Web Key Set [RFC7517] published at a well-known location used to discover public keys for signature verification.

### 3. Problem Statement

Autonomous agents executing financial transactions face a fundamental asymmetry: agent identity protocols enable merchants to verify agents, but no complementary mechanism enables agents to verify merchants.

This asymmetry creates the following risks:

#### 3.1. Merchant Impersonation

Without verifiable merchant identity, an agent may be directed to pay a fraudulent entity impersonating a legitimate merchant. Domain spoofing, homograph attacks, and DNS hijacking all represent realistic attack vectors in autonomous transaction flows.

#### 3.2. Absent Audit Trail

Regulated industries including healthcare, government procurement, and financial services require demonstrable proof that vendor identity was verified before payment. No standardized cryptographic mechanism currently exists to produce this proof in autonomous transaction flows.

#### 3.3. Policy Enforcement Gap

Human principals delegating transaction authority to autonomous agents may wish to restrict transactions to verified merchant entities. Without a machine-readable merchant identity standard, such policies cannot be enforced consistently across agent implementations.

#### 3.4. Interoperability Absence

Proprietary merchant verification mechanisms implemented by individual agent platforms are not interoperable. A merchant verified by one platform's mechanism provides no assurance to a different platform. A common standard eliminates this fragmentation.

#### 4. Design Goals

- G1. Verifiability: Any relying party MUST be able to verify an MIA using only the assertion document and publicly available key material, without dependency on a central authority.
- G2. Self-Hostability: Any merchant MUST be able to publish an MIA without registration with a central registry. Discovery via well-known URI [RFC8615] enables decentralized publication.
- G3. Interoperability: The MIA format MUST permit independent implementation by any party. No proprietary extensions SHALL be required for basic verification.
- G4. Minimal Disclosure: The MIA MUST contain only the fields necessary to establish merchant identity. Trust scoring, ranking, and behavioral data are explicitly out of scope.
- G5. Composability: The MIA MUST complement existing agent identity and payment authorization protocols without requiring modification to those protocols.
- G6. Forward Compatibility: The MIA format MUST support extension through optional fields without breaking existing implementations.

#### 5. Merchant Identity Assertion

An MIA is a JSON [RFC8259] document served with media type `application/merchant-identity+json` (Section 12.3). It consists of a Claims Object (Section 5.1 and Section 5.2) and an embedded Proof Object (Section 5.3).

The Claims Object is the canonical input to the signing procedure. The Proof Object records the resulting signature and key reference. Both are serialized together as a single JSON document on the wire (Section 5.4).

##### 5.1. Claim Fields (Required)

`version` (integer): The MIA specification version. This document defines version 1. Implementations MUST reject assertions with unrecognized version values.

`subject` (string): The subject domain. MUST be a fully qualified domain name (FQDN) expressed in lowercase. MUST match the domain from which the assertion is served or the domain for which the issuer holds explicit authorization per Section 6.2.

`legalName (string)`: The registered legal name of the merchant entity as it appears in the jurisdiction of incorporation or registration.

`entityType (string)`: The legal entity type. MUST be one of: "corporation", "llc", "partnership", "sole\_proprietor", "cooperative", "nonprofit", "government", or "other".

`jurisdiction (string)`: The ISO 3166-1 alpha-2 country code of the primary jurisdiction of registration (e.g., "US", "GB", "DE"). Where a subdivision-level identifier is available and relevant, implementors MAY use an ISO 3166-2 code (e.g., "US-AZ") in the extensions field.

`issuedAt (string)`: The UTC datetime at which this assertion was issued, in RFC 3339 format (e.g., "2026-04-15T00:00:00Z").

`expiresAt (string)`: The UTC datetime after which this assertion MUST NOT be accepted, in RFC 3339 format. Issuers SHOULD NOT set validity periods exceeding 365 days.

`issuer (object)`: An object identifying the signing entity. MUST contain:

`name (string)`: Human-readable issuer name.

`domain (string)`: Issuer FQDN.

`keyDirectory (string)`: HTTPS URI of the issuer's Key Directory (Section 7.2).

## 5.2. Claim Fields (Optional)

`registrationId (string)`: A jurisdiction-specific business registration identifier (e.g., EIN, Companies House number, GLEIF LEI [LEI]).

`evidenceUris (array of strings)`: HTTPS URIs referencing third-party authoritative records that corroborate the identity claims in this assertion (e.g., a public government business registry record, a GLEIF LEI record). Each URI MUST be publicly accessible over HTTPS without authentication.

`extensions (object)`: A JSON object for implementation-specific fields. Relying parties MUST ignore unrecognized keys. Extension keys MUST use reverse domain notation (e.g., "com.example.customField").

### 5.3. Proof Object

The proof field MUST be present in a signed MIA and MUST be a JSON object containing:

`type (string)`: MUST be "MerchantIdentityProof-EdDSA-v1" for assertions signed per this document. This value identifies the proof format defined in this section. Future documents MAY define additional proof type values.

`created (string)`: The UTC datetime at which the proof was created, in RFC 3339 format. MUST match the `issuedAt` claim field.

`verificationMethod (string)`: An HTTPS URI of the form: `{issuer.keyDirectory}#{kid}` where `{kid}` is the key identifier within the Key Directory used to produce this signature. This field anchors the signature to a specific, named key and provides a deterministic path for key retrieval without requiring the relying party to trust the payload-embedded `keyDirectory` value in isolation (see Section 11, trust bootstrap).

`proofValue (string)`: The base64url-encoded Ed25519 [RFC8032] signature over the canonical Claims Object, as defined in Section 7.3.

The `keyDirectory` URI appears in the assertion payload, which could raise a trust bootstrap concern if a relying party were to trust the payload as the sole source of key discovery information. To avoid this, the relying party MUST derive the authoritative key directory location from `proof.verificationMethod` and MUST NOT rely on `issuer.keyDirectory` as the sole key discovery path. Relying parties SHOULD additionally verify that the domain in `verificationMethod` matches `issuer.domain` as a consistency check.

### 5.4. Wire Format

An MIA is serialized as a single JSON document. The Claims Object fields (Section 5.1 and Section 5.2) and the proof field (Section 5.3) appear at the top level of this JSON document.

The document MUST be served over HTTPS with:

`Content-Type`: `application/merchant-identity+json`

Implementations MUST NOT serve or accept MIA documents over plain HTTP.



### 5.5. Example

The following is a non-normative example of a complete signed MIA document:

```
{
  "version": 1,
  "subject": "supplier.example.com",
  "legalName": "Example Supply Corporation",
  "entityType": "corporation",
  "jurisdiction": "US",
  "issuedAt": "2026-04-15T00:00:00Z",
  "expiresAt": "2027-04-15T00:00:00Z",
  "issuer": {
    "name": "Example Trust Registry",
    "domain": "trust.example.org",
    "keyDirectory": "https://trust.example.org/.well-known/jwks.json"
  },
  "registrationId": "12-3456789",
  "evidenceUris": [
    "https://www.gleif.org/lei/5493001KJTIIGC8Y1R12"
  ],
  "proof": {
    "type": "MerchantIdentityProof-EdDSA-v1",
    "created": "2026-04-15T00:00:00Z",
    "verificationMethod":
      "https://trust.example.org/.well-known/jwks.json#key-01",
    "proofValue": "z3FXQjecMBCm...base64url-signature"
  }
}
```

## 6. Discovery

### 6.1. Well-Known URI

A merchant MAY self-issue an MIA and publish it at:

`https://{subject-domain}/.well-known/merchant-identity.json`

This resource MUST be served over HTTPS with:

Content-Type: application/merchant-identity+json

A relying party wishing to obtain an MIA for a given merchant domain SHOULD first attempt retrieval from this well-known URI.

For a self-issued MIA, the issuer.domain field MUST equal the subject field, and the issuer.keyDirectory MUST resolve to a Key Directory at the subject domain.

## 6.2. Third-Party Issuer Authorization

Merchants MAY authorize a third-party issuer to publish an MIA on their behalf. Before relying on a third-party MIA, the relying party MUST verify that the issuer is authorized by the subject domain using one of the following mechanisms:

### a) DNS TXT Authorization Record:

The subject domain MUST publish a DNS TXT record at the name `_mia-auth.{subject-domain}`:

```
_mia-auth.supplier.example.com. IN TXT "v=mial; issuer=trust.example.org"
```

The issuer value MUST exactly match the issuer.domain field in the MIA.

### b) HTTP Delegation Document:

The subject domain MUST publish a Merchant Identity Delegation Document (MIDD) at:

```
https://{subject-domain}/.well-known/mia-delegation.json
```

An MIDD is a distinct document type from an MIA. It MUST be a JSON object signed using the same proof structure defined in Section 5.3 and MUST contain:

version (integer): MUST be 1.

type (string): MUST be "MerchantIdentityDelegation".

subject (string): The merchant domain granting delegation.

authorizedIssuer (string): The FQDN of the authorized third-party issuer.

issuedAt (string): RFC 3339 datetime.

expiresAt (string): RFC 3339 datetime.

proof (object): Per Section 5.3, signed by a key published at the subject domain's Key Directory.

The MIDD MUST be served over HTTPS with Content-Type application/json. This document does not define a dedicated media type for MIDD documents. Relying parties MUST verify the MIDD proof using a key hosted at the subject domain before accepting the named issuer as authorized.

Relying parties MUST treat an MIA as invalid if neither authorization mechanism is satisfied.

## 7. Signing and Verification

### 7.1. Signing Algorithm

All MIA signatures MUST use the Edwards-Curve Digital Signature Algorithm (EdDSA) with the Ed25519 curve [RFC8032]. No other algorithms are defined by this document.

### 7.2. Key Directory

Issuers MUST publish a JSON Web Key Set [RFC7517] at the HTTPS URI in the issuer.keyDirectory field. This document is the Key Directory.

Key Directory entries used for MIA signing MUST include:

kty: "OKP"

crv: "Ed25519"

use: "sig"

kid: A stable, unique, URL-safe identifier.

x: The base64url-encoded Ed25519 public key.

The Key Directory MUST be served over HTTPS with appropriate HTTP caching headers. Relying parties SHOULD cache key material per those headers to reduce verification latency and network load.

### 7.3. Signing Procedure

To produce an MIA signature, the issuer MUST:

1. Construct the Claims Object containing all required fields (Section 5.1) and any optional fields (Section 5.2), excluding the proof field.

2. Produce the canonical signing input by serializing the Claims Object as a JSON string with keys in lexicographic order and no insignificant whitespace [RFC8785].
3. Sign the UTF-8 encoding of the canonical string using the Ed25519 private key corresponding to the chosen kid.
4. Encode the resulting 64-byte signature as a base64url string (proofValue).
5. Construct the proof object (Section 5.3) with the type, created, verificationMethod, and proofValue fields.
6. Merge the Claims Object and proof object into a single JSON document (Section 5.4).

#### 7.4. Verification Procedure

A relying party MUST perform the following steps:

1. Retrieve the MIA document via HTTPS.
2. Verify the Content-Type is application/merchant-identity+json.
3. Parse the JSON document. Extract the proof field. If absent, reject the assertion.
4. Verify proof.type is "MerchantIdentityProof-EdDSA-v1". If not, reject. Implementations MUST NOT accept assertions with unknown proof type values.
5. Extract the key directory URI and kid from proof.verificationMethod by splitting on the last "#" character. The key directory URI is the portion before "#"; the kid is the portion after. The relying party MUST use this URI as the authoritative key directory location. It MUST also verify that the domain of this URI matches the issuer.domain field in the assertion as a consistency check. If they do not match, the assertion MUST be rejected.
6. Retrieve the Key Directory over HTTPS from the extracted URI.
7. Locate the JWK entry with the matching kid. If absent, reject.
8. Reconstruct the canonical signing input from the assertion document with the proof field removed, following the procedure in Section 7.3, step 2.

9. Verify the Ed25519 signature in `proof.proofValue` against the canonical signing input using the located public key.
10. Verify that the current time is strictly between `issuedAt` and `expiresAt` (exclusive).
11. Verify that the subject field matches the domain for which verification is being performed (case-insensitive FQDN comparison).
12. If `issuer.domain` differs from subject, verify third-party authorization per Section 6.2.
13. If all steps succeed, the assertion is valid.

A relying party MUST NOT act on an assertion that fails any verification step.

## 8. Freshness and Revocation

Relying parties MUST NOT accept assertions for which the current time is at or after the `expiresAt` value.

Issuers SHOULD limit assertion validity periods. Validity periods of 90 days or less are RECOMMENDED for issuers with automated re-issuance pipelines.

Issuers MAY support early revocation by including a `revocationUri` extension field. If present, this field MUST be an HTTPS URI that resolves to a JSON object with a boolean "revoked" field. Relying parties MAY check this URI before relying on an assertion.

Issuers MUST re-issue assertions promptly when material identity information changes (e.g., legal name change, jurisdiction change, loss of business registration).

Relying parties in high-stakes or regulated contexts SHOULD retrieve a fresh assertion rather than relying on a cached copy when the cached copy is older than 24 hours, regardless of the `expiresAt` value.

## 9. Evaluation Result Token

A verification service MAY issue an Evaluation Result Token (ERT) after successfully verifying an MIA. The ERT is a signed JWT [RFC7519] that serves as a portable audit artifact confirming that verification was performed.

The ERT is an optional layer. Its issuance does not alter the validity of the underlying MIA.

### 9.1. Claims

iss (string, REQUIRED): The domain of the verification service.

sub (string, REQUIRED): The subject domain that was verified.

iat (NumericDate, REQUIRED): Time of ERT issuance.

exp (NumericDate, REQUIRED): Expiry time. MUST NOT exceed iat + 300 seconds to limit replay exposure.

jti (string, REQUIRED): A cryptographically random, globally unique token identifier for replay detection.

mia\_verified (boolean, REQUIRED): True if a valid, unexpired MIA was located and successfully verified for the subject domain per Section 7.4. False otherwise.

mia\_subject (string, REQUIRED when mia\_verified is true): The subject field from the verified MIA.

mia\_issued\_at (string, REQUIRED when mia\_verified is true): The issuedAt field from the verified MIA.

mia\_issuer\_domain (string, REQUIRED when mia\_verified is true): The issuer.domain field from the verified MIA.

No consumer or agent identity fields SHALL be included in an ERT without explicit principal consent.

### 9.2. Signing

The ERT MUST be signed as a JWS [RFC7515] using EdDSA with Ed25519 [RFC8032]. The JWS Protected Header MUST include:

alg: "EdDSA"

kid: Key identifier in the verification service's Key Directory.

Implementations MUST follow JWT Best Current Practices [RFC8725], including algorithm validation and rejection of the none algorithm.

### 9.3. Example

The following is a non-normative example of a decoded ERT payload:

```
{
  "iss": "trust.example.org",
  "sub": "supplier.example.com",
  "iat": 1744934400,
  "exp": 1744934700,
  "jti": "f7a2c9e1b4d8",
  "mia_verified": true,
  "mia_subject": "supplier.example.com",
  "mia_issued_at": "2026-04-15T00:00:00Z",
  "mia_issuer_domain": "trust.example.org"
}
```

A relying party MAY include the ERT in payment metadata or transaction records to provide downstream parties with a verifiable audit artifact that pre-transaction merchant identity verification occurred.

## 10. Privacy Considerations

### 10.1. Minimal Disclosure

MIA documents MUST NOT include personal data about natural persons except where the subject is a sole proprietor, such disclosure is legally required in the jurisdiction, and the natural person has provided explicit consent.

### 10.2. Evidence URI Privacy

The evidenceUris field, if included, reveals which third-party registries were used to corroborate the merchant's identity. Issuers SHOULD assess the privacy implications of each URI before inclusion.

### 10.3. Relying Party Query Privacy

Real-time retrieval of MIAs from well-known URIs or Key Directories reveals which merchants a relying party intends to transact with. Implementations SHOULD use caching and pre-fetching strategies to reduce this correlation risk.

#### 10.4. Consumer Principal Privacy

ERTs as defined in Section 9 MUST NOT include consumer or agent principal identifiers. Verification services MUST NOT log consumer identifiers in association with merchant verification events without explicit consumer consent.

### 11. Security Considerations

#### 11.1. Replay Attacks

The ERT jti claim provides a unique token identifier. Relying parties MUST maintain a jti cache for at least the duration of the token's validity window ( $\text{exp} - \text{iat}$ ) and MUST reject any ERT whose jti has been seen previously. The RECOMMENDED maximum ERT validity of 300 seconds bounds the size of this cache.

#### 11.2. Key Compromise

If an issuer's signing key is compromised, all MIA documents signed with that key MUST be treated as invalid. Issuers MUST maintain key rotation procedures and MUST remove compromised keys from their Key Directory promptly. Relying parties that cache Key Directory material MUST re-fetch on signature verification failure.

#### 11.3. Domain Transfer

An MIA is bound to a domain name. Transfer of that domain to a new owner does not automatically invalidate outstanding assertions. Issuers MUST implement procedures to detect domain transfer and revoke or re-evaluate affected assertions without delay.

#### 11.4. Malicious Issuers

A third-party issuer operating in bad faith may publish false assertions. Relying parties SHOULD NOT treat all issuers as equally trustworthy. Relying parties MAY maintain allowlists of trusted issuer domains and SHOULD treat assertions from unknown issuers with heightened scrutiny. The third-party authorization requirement in Section 6.2 ensures that an issuer cannot publish an MIA for a domain that has not explicitly authorized it, but does not guarantee the issuer itself is trustworthy.



### 11.5. Stale Assertions

Merchant legal standing may change after assertion issuance (e.g., insolvency, license revocation, sanctions designation). Relying parties in regulated industries **MUST NOT** treat an MIA as sole evidence of current compliance and **SHOULD** supplement MIA verification with jurisdiction-appropriate checks for high-value or high-risk transactions.

### 11.6. Downgrade Attacks

An attacker with network access could substitute an older but still-valid MIA for a more recently issued one. Relying parties **SHOULD** record the `issuedAt` timestamp of the assertion used in a transaction and **SHOULD** reject assertions whose `issuedAt` predates their recency threshold for the transaction context.

### 11.7. Third-Party Authorization Poisoning

DNS TXT records (Section 6.2, mechanism a) are susceptible to cache poisoning if DNSSEC is not deployed. Implementations **MUST** use DNSSEC-validated resolution where available when performing DNS-based third-party authorization checks. HTTP delegation documents (mechanism b) **MUST** be retrieved over HTTPS with certificate validation; relying parties **MUST** reject delegation documents served without a valid TLS certificate.

### 11.8. JWT Security

Implementations of the ERT (Section 9) **MUST** comply with JWT Best Current Practices [RFC8725], including:

- \* Explicitly validating the `alg` header value.
- \* Rejecting tokens with `alg: "none"`.
- \* Validating `iss`, `sub`, `iat`, `exp`, and `jti` before acting on any token.
- \* Not relying on unverified header parameters for security decisions.

## 12. IANA Considerations

### 12.1. Well-Known URI: `merchant-identity.json`

This document requests registration in the "Well-Known URIs" registry [RFC8615]:

URI suffix: merchant-identity.json

Change controller: IETF

Specification document(s): This document, Section 6.1

Status: permanent

Related information: None

#### 12.2. Well-Known URI: mia-delegation.json

This document requests registration in the "Well-Known URIs" registry [RFC8615]:

URI suffix: mia-delegation.json

Change controller: IETF

Specification document(s): This document, Section 6.2

Status: permanent

Related information: Used to publish a Merchant Identity Delegation Document (MIDD) authorizing a named third-party issuer to publish MIAs on behalf of the subject domain.

#### 12.3. Media Type: application/merchant-identity+json

This document requests registration of a new media type per [RFC6838]:

Type name: application

Subtype name: merchant-identity+json

Required parameters: none

Optional parameters: version: The MIA version number (integer).  
Defaults to 1 if absent.

Encoding considerations: Binary (UTF-8 encoded JSON per [RFC8259]).

Security considerations: See Section 11 of this document.

Interoperability considerations: See Section 4 of this document.

Published specification: This document.

Applications that use this media type: Autonomous agents performing pre-transaction merchant identity verification; verification services issuing Evaluation Result Tokens; merchant identity registries.

Fragment identifier considerations: None.

Additional information: Deprecated alias names for this type: None.  
Magic number(s): None. File extension(s): .mia.json. Macintosh file type code(s): None.

Person & email address to contact for further information:  
founder@registeredbrands.ai

Intended usage: COMMON

Restrictions on usage: None.

Author: Robb Anders

Change controller: IETF

## 13. References

### 13.1. 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>>.
- [RFC7515] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Signature (JWS)", RFC 7515, DOI 10.17487/RFC7515, May 2015, <<https://www.rfc-editor.org/rfc/rfc7515>>.
- [RFC7517] Jones, M., "JSON Web Key (JWK)", RFC 7517, DOI 10.17487/RFC7517, May 2015, <<https://www.rfc-editor.org/rfc/rfc7517>>.
- [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>>.
- [RFC8032] Josefsson, S. and I. Liusvaara, "Edwards-Curve Digital Signature Algorithm (EdDSA)", RFC 8032, DOI 10.17487/RFC8032, January 2017, <<https://www.rfc-editor.org/rfc/rfc8032>>.

- [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>>.
- [RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <<https://www.rfc-editor.org/rfc/rfc8259>>.
- [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>>.
- [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/rfc/rfc8785>>.
- [RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, <<https://www.rfc-editor.org/rfc/rfc6838>>.

### 13.2. Informative References

- [RFC9421] Backman, A., Ed., Richer, J., Ed., and M. Sporny, "HTTP Message Signatures", RFC 9421, DOI 10.17487/RFC9421, February 2024, <<https://www.rfc-editor.org/rfc/rfc9421>>.
- [WEBBOTAUTH]  
Meunier, T. and S. Major, "Web Bot Auth Architecture", Work in Progress, Internet-Draft, draft-meunier-web-bot-auth-architecture, 2026, <<https://datatracker.ietf.org/doc/draft-meunier-web-bot-auth-architecture/>>.
- [KYAPAY] IETF, "KYAPay Profile", Work in Progress, Internet-Draft, draft-skyfire-kyapayprofile-01, <<https://datatracker.ietf.org/doc/draft-skyfire-kyapayprofile-01/>>.
- [LEI] Global Legal Entity Identifier Foundation, "Legal Entity Identifier", 2026, <<https://www.gleif.org/>>.

## Acknowledgements

The author thanks the participants of the IETF web-bot-auth working group for their engagement, and in particular David Schinazi and Rifaat Shekh-Yusef for their guidance in directing this work toward the DISPATCH working group. The author also acknowledges the complementary work in draft-skyfire-kyapayprofile-01, which addresses agent-side identity in a manner that this document's merchant-side framework is designed to complement.

## Author's Address

Robb Anders  
RegisteredBrands.AI  
Email: [founder@registeredbrands.ai](mailto:founder@registeredbrands.ai)  
URI: <https://registeredbrands.ai>