A2AX-Core Protocol Specification

Williams Creative

Intended Status: Standards Track
Version 0.1.3

Abstract

This document defines the normative specification of the A2AX-Core Protocol. It standardizes portable identity certificates, cryptographic verification requirements, capability declaration semantics, trust store behavior, and an agent-to-agent handshake mechanism. The protocol is neutral, verifier-controlled, and registry-independent.

Contents

1 Status of This Document

This document specifies version 0.1.3 of the A2AX-Core Protocol.

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” are to be interpreted as described in RFC 2119.

2 Terminology

Agent — Autonomous computational system capable of initiating interactions.

Identity Certificate — Self-contained cryptographically signed artifact binding an Agent identifier to capabilities.

Issuer — Entity that signs Identity Certificates.

Verifier — Entity validating certificates and applying local trust policy.

Trust Store — Local collection of trusted issuer public keys.

3 Separation of Concerns

A2AX-Core enforces strict structural separation between identity, capability, trust, and reputation.

3.1 Identity

Identity is the cryptographic binding between an agent identifier and a public key.

Identity answers: “Who signed this message?”

Identity MUST NOT imply competence, reliability, honesty, or economic value.

3.2 Capability

Capabilities are declared functional scopes cryptographically bound to identity.

Capability answers: “What does this agent claim it can do?”

Capabilities are claims and MUST NOT be interpreted as guarantees.

3.3 Trust

Trust is a verifier-local policy decision.

Trust answers: “Do I accept interaction under my rules?”

Trust MUST remain exclusively under verifier control and MUST NOT be embedded within certificates.

3.4 Reputation (Out of Scope)

Reputation represents accumulated behavioral or economic history over time.

Reputation systems MAY exist externally but MUST NOT be embedded into the A2AX-Core protocol layer.

The core protocol SHALL NOT define scoring, ranking, weighting, or economic valuation mechanisms.

4 Protocol Overview

The protocol defines:

- Identity Certificate structure
- Verification algorithm
- Trust store requirements
- Capability declaration semantics
- Agent-to-Agent (A2A) handshake procedure

5 Identity Certificate Structure

An Identity Certificate MUST include:

- Agent identifier
- Issuer identifier
- Agent public key (Ed25519)
- Capability declarations
- Issuance timestamp
- Expiration timestamp
- Digital signature (Ed25519)

Certificates MUST be self-contained and MUST NOT require registry lookup for validation.

6 Verification Algorithm

Verification MUST proceed in the following order:

1. Validate signature using issuer public key
2. Validate expiration timestamp
3. Validate issuance timestamp
4. Enforce nonce replay protection
5. Apply local trust policy

Verification MUST fail if any step fails.

7 Trust Store Requirements

- Trust stores MUST be locally configurable
- Trust stores MAY be empty
- No mandatory global root of trust SHALL exist

8 Agent-to-Agent Handshake

Handshake procedure:

1. Exchange Identity Certificates
2. Perform verification algorithm
3. Validate replay nonce
4. Negotiate permitted capabilities
5. Establish secure session context

Handshake MUST terminate upon verification failure.

9 Threat Model

The protocol assumes adversaries MAY:

- Attempt to forge identity certificates
- Replay previously valid handshake messages
- Attempt capability escalation
- Compromise non-hardware-protected private keys
- Operate malicious but cryptographically valid agents

The protocol assumes adversaries CANNOT:

- Break Ed25519 cryptographic primitives
- Forge signatures without private key compromise
- Override verifier-controlled trust policy

A2AX-Core guarantees cryptographic authenticity, not behavioral integrity.

10 Adversary Assumptions

Class A — Network Adversary
May intercept, replay, or inject traffic. Mitigated via signature validation, expiration enforcement, and nonce replay protection.

Class B — Malicious Certified Agent
Possesses a valid certificate but behaves dishonestly. Mitigated through verifier-controlled trust policy.

Class C — Compromised Key Holder
Private key material has been exposed. Mitigated via expiration limits and optional short-lived certificates. Global revocation is out of scope.

Class D — Centralization Actor
Attempts to impose mandatory trust anchors or ecosystem control. Mitigated through structural neutrality and locally configurable trust stores.

11 Security Considerations

Implementations MUST protect private keys.
Replay protection MUST be enforced.
Clock synchronization SHOULD be considered.
Hardware-backed key storage is RECOMMENDED where available.

12 Privacy Considerations

Capability declarations SHOULD follow data minimization principles.
Identifiers SHOULD avoid embedding personally identifiable information.

13 Extensibility

Extensions MUST NOT alter core verification semantics.
Future capability namespaces MAY be registered via documented extension processes.

14 Conformance

An implementation conforms to this specification if it:

- Implements the required Identity Certificate structure
- Implements the full Verification Algorithm
- Enforces Trust Store requirements
- Correctly executes the A2A handshake procedure

15 Governance and Versioning

Protocol changes require public proposal, documented rationale, and semantic version updates.
Backward compatibility SHOULD be preserved where possible.

16 Fork Test Guarantee

Removal of any specific organization, service, or ecosystem reference MUST NOT break protocol functionality.