Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 5 December 2026

R. Sharif
CyberSecAI Ltd

3 June 2026

ATTP: Agent Trust Transport Protocol
draft-sharif-attp-01

Abstract

This document specifies the Agent Trust Transport Protocol (ATTP), a protocol-agnostic framework for trust scoring, cryptographic identity, action-limit enforcement, compliance gating, and tamper-evident audit for autonomous AI agents.

ATTP separates trust from identity. Identity protocols answer "who is this agent?" ATTP answers "should this agent be allowed to perform this action, at this magnitude, against this counterparty, right now?"

The protocol defines five progressive trust levels (L0-L4), per-agent ECDSA P-256 cryptographic identity, a five-dimension behavioural trust scoring model, action-limit tiers derived from trust scores, real-time compliance gating (sanctions screening, jurisdictional controls), kill switches for instant revocation, anomaly detection for agent behaviour, and a public trust query API.

ATTP is transport-agnostic. This document defines the core framework. Protocol-specific bindings map ATTP trust enforcement to individual transports: MCPS provides the binding for the Model Context Protocol (MCP), with additional bindings defined for REST APIs, Google A2A, gRPC, GraphQL, and SPIFFE/SPIRE.

This specification supersedes draft-sharif-agent-payment-trust-00, broadening the scope from payment transactions to any agent action requiring trust-gated authorisation, and consolidates the parallel draft-sharif-attp-agent-trust-transport-00 into a single ATTP lineage.

Status of This Memo

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

This Internet-Draft will expire on 5 December 2026.

Copyright Notice

Table of Contents

1. Introduction

1.1. The Trust Gap

The emergence of autonomous AI agents that discover resources at runtime, execute multi-step tasks across trust domains, and take consequential actions without human oversight creates a security gap that identity protocols alone cannot close.

Authentication answers a binary question: is this agent who it claims to be? The answer is yes or no. But consequential actions -- financial transactions, data access, infrastructure changes, contract execution -- require graduated assessment:

* Has this agent earned the right to perform actions of this magnitude?

* Does this agent’s behavioural history justify the level of trust required?

* Is the counterparty cleared for this interaction (sanctions, jurisdictional compliance)?

* Can this agent be stopped instantly if something goes wrong?

* Is there a tamper-evident record of every action for audit?

Identity is necessary but not sufficient. ATTP provides the trust layer that sits above identity and below application logic.

1.2. Relationship to Identity Protocols

ATTP is not an identity protocol and not a Privileged Access Management (PAM) replacement. It is a trust-decision and action-governance layer for autonomous agents, designed to compose with identity and transport-security mechanisms such as OIDC, mTLS, SPIFFE/SPIRE, and HTTP Message Signatures, and with protocol bindings such as REST and A2A.

ATTP does not replace identity protocols. It consumes identity assertions from any standards-compliant source:

* OAuth 2.0 / OpenID Connect tokens (RFC 6749, OpenID Core)

* X.509 certificates (RFC 5280)

* HTTP Message Signatures (RFC 9421)

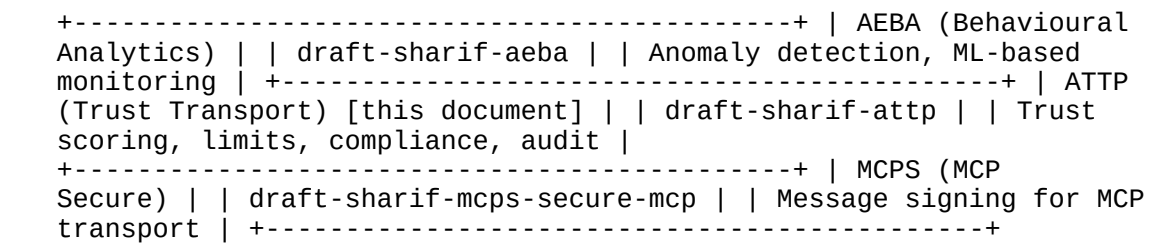
* Self-issued cryptographic identities (JWKS-based)

* Keycloak, Azure AD, Okta, or any OIDC-compliant provider

ATTP requires proof that an agent is who it claims to be. It does not prescribe how that proof is obtained. Once identity is established, ATTP evaluates trust, enforces limits, gates compliance, and records the decision.

1.3. Relationship to Existing Specifications

This specification is part of a three-layer stack:



MCPS provides message-level cryptographic security for the Model Context Protocol. ATTP provides the trust framework that MCPS (and other transport bindings) enforce. AEBA provides behavioural analytics that feed into ATTP trust scoring.

1.4. Relationship to Other IETF Work

The IETF agent-identity and delegation space is active. ATTP is the trust-decision and action-governance layer ABOVE these mechanisms; it composes with them rather than replacing them.

Workload identity for agents is addressed by the WIMSE architecture [I-D.ietf-wimse-arch] and by composition guidance such as [I-D.klrc-aiagent-auth], which combines workload identity, OAuth, and shared signals. ATTP consumes such identities; the SPIFFE/SPIRE binding in this document is one such integration.

Delegation and on-behalf-of semantics are provided by OAuth 2.0 Token Exchange [RFC8693] (the "act" and "may_act" claims), the actor-chain profiles in [I-D.mw-spice-actor-chain], and capability attenuation in [I-D.niyikiza-oauth-attenuating-agent-tokens], which covers delegation depth, intent binding, and monotonic scope attenuation. ATTP evaluates these delegation chains as inputs to a trust decision; it does not define a competing delegation format.

Pre-action signed authorisation with live risk assessment is explored in [I-D.nelson-agent-delegation-receipts]. ATTP’s action signing and trust scoring address the same need and can interoperate with such receipts.

ATTP’s distinct contribution is the graduated trust model (L0-L4), action-limit enforcement, compliance gating, kill switches, and tamper-evident audit applied on top of whichever identity and delegation mechanisms a deployment already uses.

1.5. Design Goals

1. Transport-agnostic. ATTP defines trust semantics, not wire format. Protocol bindings map ATTP to specific transports.

2. Identity-agnostic. ATTP consumes identity from any source. It does not compete with any identity protocol -- OAuth, OIDC, mTLS, SPIFFE, HTTP Message Signatures, or future standards.

3. Graduated trust. Binary identity is insufficient. Five trust levels (L0-L4) with associated action limits provide proportional authorisation.

4. Fail-closed. If trust cannot be evaluated, the default is deny. No silent degradation.

5. Instant revocation. Kill switches take effect on the next request. No waiting for certificate expiry or CRL refresh.

6. Compliance-native. Sanctions screening and jurisdictional controls are built into the protocol, not bolted on.

7. Self-hostable. The Trust Authority has no external service dependency. Operators run their own instance.

8. Auditable. Every trust decision produces a tamper-evident record with hash-chain integrity.

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.

Agent: An autonomous software entity that performs actions on behalf of a principal (human or organisation) via standardised protocols. Actions include but are not limited to: financial transactions, data queries, tool invocations, infrastructure changes, and contract execution.

Principal: The human or organisational entity accountable for an agent’s actions.

Trust Authority (TA): The service that maintains agent registrations, computes trust scores, evaluates compliance gates, enforces action limits, and records audit entries. The TA is self-hostable with no external dependency.

Trust Score: A numerical value (0-100) representing the assessed trustworthiness of an agent based on multiple behavioural dimensions.

Trust Level: A discrete classification (L0-L4) derived from the trust score, with associated action limits.

Action: Any operation an agent requests that requires trust evaluation. Actions are protocol-specific: a payment in Stripe, a tool call in MCP, an API request in REST, a task in A2A.

Action Limit: A constraint on the magnitude or frequency of actions an agent may perform at a given trust level.

Kill Switch: A per-agent or per-principal flag that, when activated, causes immediate denial of all actions regardless of trust level.

Compliance Gate: A check performed before action execution that evaluates regulatory constraints (sanctions lists, jurisdictional rules, embargo lists).

Protocol Binding: A specification that maps ATTP trust enforcement to a specific transport protocol (MCP, REST, A2A, gRPC, GraphQL).

Agent Passport: A cryptographically signed identity document containing the agent's public key hash, principal identity, authorised scope, trust level, and issuance metadata.

3. Agent Identity

3.1. Key Pair Generation

Each agent MUST have a unique ECDSA key pair using the P-256 curve [FIPS186-5].

Key pairs MAY be generated by the Trust Authority and provided to the principal (the private key MUST be returned exactly once and MUST NOT be stored by the Trust Authority), or generated by the principal and the public key registered with the Trust Authority (RECOMMENDED for production deployments).

Implementations MUST support ECDSA P-256 (ES256). Implementations MAY additionally support Ed25519 (EdDSA) for environments where P-256 is not available.

3.2. Public Key Registration

The principal MUST register the agent's public key with the Trust Authority at agent creation time. The Trust Authority stores ONLY the public key. The private key MUST remain under the principal's control at all times.

For production deployments, the private key SHOULD be stored in a Hardware Security Module (HSM), cloud Key Management Service (KMS), or platform-native secure storage.

3.3. Agent Passport

Upon registration, the Trust Authority issues an Agent Passport -- a signed JSON document containing:

* agentId: Unique identifier for the agent.
* publicKeyHash: SHA-256 hash of the agent’s public key, binding the passport to a specific key pair.
* principalId: Identifier of the owning principal.
* scope: Array of authorised action categories.
* trustLevel: Current trust level (L0-L4).
* issuedAt: ISO 8601 timestamp of issuance.
* expiresAt: ISO 8601 timestamp of expiry.
* issuer: Identifier of the Trust Authority.
* protocolVersion: ATTP version (this document: "1.0").

The passport is signed by the Trust Authority’s key using ECDSA P-256. The signature covers the JSON Canonicalization Scheme (JCS) [RFC8785] serialisation of the passport fields.

Passports MUST expire. Maximum lifetime MUST NOT exceed 365 days. RECOMMENDED lifetime is 90 days for L0-L2 agents and 180 days for L3-L4 agents.

3.4. Key Rotation

Agents MUST support key rotation without identity loss. When rotating keys:

1. The principal generates a new key pair.
2. The principal registers the new public key with the TA.
3. The TA issues a new passport binding the agentId to the new public key.
4. The old key enters a grace period (RECOMMENDED: 24 hours) during which both keys are accepted.
5. After the grace period, the old key is revoked.

Key rotation MUST NOT reset the agent’s trust score.

4. Challenge-Response Identity Verification

When a platform needs to verify an agent’s identity before granting access, it uses the following challenge-response protocol.

4.1. Challenge Issuance

The platform (or TA on behalf of the platform) generates a challenge:

* challenge: 32 bytes of cryptographically random data, hex-encoded (64 characters).
* expiresAt: Timestamp, MUST be no more than 60 seconds from issuance.
* agentId: The agent for which the challenge is issued.

Each challenge MUST be single-use. The TA MUST reject any attempt to verify a challenge that has already been used.

4.2. Challenge Signing

The agent signs the challenge using its private key:

signature = ECDSA-Sign(SHA-256(challenge), agentPrivateKey)

The signature uses IEEE P1363 fixed-length r||s encoding (64 bytes for P-256) per RFC 7518 Section 3.4.

4.3. Signature Verification

The TA verifies the signature using the agent’s registered public key. If valid, the TA returns the agent’s current trust score, level, and recommendation.

4.4. Failure Handling

If verification fails:

* The TA MUST log the failure as a potential impersonation attempt.
* The agent’s trust score MUST be penalised (RECOMMENDED: -10 points per failed attempt).
* The response MUST include an error code: IMPERSONATION, CHALLENGE_EXPIRED, CHALLENGE_REPLAYED, or AGENT_MISMATCH.

5. Trust Scoring Model

5.1. Scoring Dimensions

The trust score is computed from five dimensions:

1. Code Attestation (CA): Whether the agent's code or configuration has been cryptographically verified against its declared specification.

2. Execution Success Rate (ES): The proportion of the agent's past actions that completed without anomaly, dispute, or reversal.

3. Behavioural Consistency (BC): The statistical consistency of the agent's action patterns relative to its established baseline.

4. Operational Tenure (OT): The duration the agent has been registered and actively operating.

5. Anomaly History (AH): The inverse of the count and severity of detected anomalies.

5.2. Dimension Weights

Each dimension contributes 20% to the overall score:

W_CA = W_ES = W_BC = W_OT = W_AH = 0.20

Implementations MAY adjust weights provided the sum equals 1.0 and no single dimension exceeds 0.40.

5.3. Score Computation

raw = (CA * W_CA) + (ES * W_ES) + (BC * W_BC) + (OT * W_OT) + (AH * W_AH)

Each dimension value is in the range [0, 100].

score = clamp(raw + bonus + dormancy, 0, 100)

5.4. Trust Levels

Level	Score	Label	Payment Enabled
L0	0-19	No Access	No
L1	20-39	Restricted	Micro only
L2	40-59	Standard	Yes
L3	60-79	Elevated	Yes (monitored)
L4	80-100	Full Access	Yes (audited)

+--------+--------+------------------+------------------+

No trust level SHALL have unlimited action authority. L4 represents the maximum tier with mandatory enhanced auditing.

5.5. Dynamic Adjustment

The trust score adjusts dynamically based on observed behaviour:

* Successful action: +0.5 trust bonus.
* Blocked action (over limit): -2 trust bonus.
* Anomaly detected: -5 per anomaly.
* Critical anomaly (3+ simultaneous): -20 trust bonus.
* Failed identity verification: -10 trust bonus.
* Probing detected: -15 trust bonus.

Self-dealing actions (agent-to-agent within the same principal) MUST earn zero trust bonus.

5.6. Dormancy Decay

Agents inactive for extended periods MUST have their trust score reduced:

* 30 days inactive: -10 penalty.
* 60 days inactive: -20 penalty.
* 90+ days inactive: -30 penalty.

The dormancy penalty is removed when the agent resumes activity.

5.7. Promotion Rate Limiting

Trust level promotions MUST be rate-limited to prevent trust farming attacks:

* L0 to L1: Minimum 24 hours at L0 with at least 5 successful actions.
* L1 to L2: Minimum 7 days at L1 with at least 20 successful actions.

* L2 to L3: Minimum 30 days at L2 with at least 100 successful actions and zero critical anomalies.

* L3 to L4: Minimum 90 days at L3 with at least 500 successful actions, zero anomalies, and manual principal attestation.

Demotions are instant. A single critical anomaly at L4 drops the agent to L2 immediately.

6. Action Limit Enforcement

6.1. Per-Action Limits

Each action MUST be checked against the per-action limit of the agent's current trust level. Actions exceeding the limit MUST be rejected with error code ATTP-ACTION-LIMIT.

Action limits are protocol-binding-specific. For financial transactions:

+--------+------------------+------------------+ | Level | Per-TX Limit | Daily Limit | +--------+------------------+------------------+ | L0 | $0 | $0 | | L1 | $10 | $50 | | L2 | $100 | $500 | | L3 | $1,000 | $5,000 | | L4 | $50,000 | $200,000 | +--------+------------------+------------------+

For non-financial actions, protocol bindings define equivalent limits (e.g., API calls per minute, data volume per request).

6.2. Daily Aggregate Limits

The TA MUST track daily aggregate action magnitude per agent within a rolling 24-hour window.

6.3. Principal Aggregate Limits

To prevent Sybil attacks (creating many agents to circumvent per-agent limits), the TA MUST enforce aggregate limits per principal across all their agents.

6.4. Cooling Period

After a trust level promotion, the agent MUST operate under the previous level's limits for a cooling period (RECOMMENDED: 24 hours) before the new limits take effect.

7. Kill Switches

7.1. Per-Agent Kill Switch

Each agent MUST have an independently controllable kill switch. When activated:

* ALL actions by the agent are denied immediately.
* The denial takes effect on the next request (no grace period).
* The agent's trust score is frozen (not reset).
* The kill switch state is recorded in the audit trail.

Reactivation requires explicit principal action and MUST NOT be automated.

7.2. Per-Principal Kill Switch

Each principal MUST have a kill switch that simultaneously disables ALL agents under their control.

7.3. Emergency Freeze

The TA MUST support an emergency freeze that disables all agents across all principals. This is intended for infrastructure-level incidents. Emergency freeze MUST require multi-party authorisation (RECOMMENDED: two independent operators).

8. Compliance Gating

8.1. Compliance Gate Interface

ATTP defines a compliance-gate interface, not a specific compliance regime. Before executing a consequential action, the TA MUST evaluate the action against the configured set of compliance gates and MUST block the action if any gate denies it. Each gate is a pluggable policy that returns allow, deny, or review for a given action and its parties. A deployment selects the gates appropriate to its jurisdiction and use case. Gates MAY be chained; a deny from any gate blocks the action with an error code identifying the gate (for example, ATTP-GATE-DENY).

8.2. Sanctions Screening (Example Gate)

Sanctions screening is a common compliance gate but is NOT mandated by this protocol, as applicable lists are jurisdiction-specific. A deployment subject to sanctions regulation SHOULD configure a sanctions gate that screens the relevant parties against the lists applicable to it (for example, OFAC SDN, EU Consolidated, UK HMT, or UN Security Council lists).

A sanctions gate SHOULD use fuzzy matching with a configurable threshold (RECOMMENDED: 70% match score), block matches at or above the threshold, and keep its lists current (RECOMMENDED: at least daily updates).

8.3. Jurisdictional Controls

Actions MAY be restricted based on the jurisdiction of the agent, the principal, or the counterparty. The TA SHOULD support configurable jurisdiction rules, expressed as compliance gates.

8.4. Compliance Record

Every compliance-gate evaluation MUST produce an audit record containing: the query, the gates evaluated, the results (including near-misses), and the decision. Compliance records MUST be retained for the period required by applicable regulation (RECOMMENDED minimum: 5 years).

9. Anomaly Detection

9.1. Magnitude Anomaly

An action whose magnitude exceeds 3 standard deviations from the agent’s historical mean MUST be flagged.

9.2. Velocity Anomaly

An action rate exceeding 2x the agent’s historical maximum within any 5-minute window MUST be flagged.

9.3. Temporal Anomaly

Actions outside the agent’s established operating hours (based on 90-day history) SHOULD be flagged.

9.4. Counterparty Anomaly

Actions directed at counterparties with which the agent has no prior history, when combined with elevated magnitude, MUST be flagged.

9.5. Probing Anomaly

A pattern of incrementally increasing action magnitudes (e.g., $1, $5, $10, $50, $100) within a short window indicates probing behaviour and MUST trigger a -15 trust penalty and flag the agent for review.

9.6. Self-Dealing Detection

Actions between agents belonging to the same principal MUST earn zero trust bonus and SHOULD be flagged if they exceed 10% of the agent’s total action volume.

10. Public Trust Query API

10.1. Request Format

Any platform MAY query the TA for an agent’s trust status:

GET /v1/trust/{agentId}

The TA MUST support this query without requiring a pre-existing relationship with the querying platform.

10.2. Response Format

{ "agentId": "agent_abc123", "trust": { "score": 67, "level": 3, "label": "L3 -- Elevated" }, "recommendation": "ALLOW", "limits": { "perAction": 100000, "daily": 500000 }, "identity": { "verified": true, "principalId": "dev_xyz", "jurisdiction": "GB" }, "meta": { "protocolVersion": "1.0", "queriedAt": "2026-04-30T22:00:00Z", "checkedBy": "CyberSecAI Trust Authority" } }

10.3. Batch Queries

The TA MUST support batch queries for efficiency:

POST /v1/trust/batch { "agentIds": ["agent_1", "agent_2", "agent_3"] }

Maximum batch size: 100 agents.

10.4. Rate Limiting

The TA MUST rate-limit trust queries:

* Anonymous queries: 120 per minute per IP.
* Authenticated queries: 600 per minute per API key.

10.5. Information Disclosure Controls

The public trust API MUST NOT expose:

* Raw scoring dimension values.
* Transaction history or action details.
* Private key material or key fingerprints.
* Internal anomaly thresholds.

The API provides trust level, recommendation, and limits only.

11. Action Signing and Receipts

11.1. Action Envelope

Every action processed through ATTP produces a signed envelope:

```
{ "actionId": "act_unique_id", "agentId": "agent_abc123", "action":
"payment_initiate", "magnitude": 5000, "counterparty":
"recipient_name", "trustLevel": 3, "complianceResult": "CLEAR",
"timestamp": "2026-04-30T22:00:00Z", "signature": "ECDSA signature
over canonical JSON" }
```

The signature covers the JCS [RFC8785] serialisation of all fields except the signature field itself.

11.2. Hash Chain

Action envelopes are linked in a hash chain:

```
hash_n = SHA-256(hash_{n-1} || JCS(envelope_n))
```

The genesis hash is SHA-256("ATTP-GENESIS").

Breaking the hash chain indicates tampering. The TA MUST verify chain integrity on every append.

11.3. Cryptographically Attributable Receipts

After successful action execution, the TA issues a receipt signed by the TA's key. The receipt includes:

* The action envelope.
* The TA's signature over the envelope.
* The hash chain position.
* The compliance check result.

Receipts are cryptographically attributable: they prove that a specific key signed a specific envelope at a specific position in the hash chain. This is an evidentiary property, not a legal assertion of intent or consent. A valid signature demonstrates that the holder of the key produced it; it does not, by itself, establish that the action was authorised by a human, free of coercion, or produced on an uncompromised host. Establishing intent or consent is out of scope for this document and, where required, MUST be provided by an additional mechanism (for example, a human-presence attestation bound to the action).

12. Protocol Bindings

ATTP is transport-agnostic. Protocol bindings map ATTP trust enforcement to specific transports.

12.1. MCP Binding (MCPS)

The MCP binding is specified in draft-sharif-mcps-secure-mcp. MCPS wraps JSON-RPC messages in signed envelopes carrying ATTP trust claims.

Trust level is conveyed in the MCPS envelope header. Action limits map to tool call parameters (amount, scope). Kill switch state is checked before tool execution.

12.2. REST Binding

For REST APIs, ATTP trust is conveyed via HTTP headers:

X-ATTP-Trust-Level: 2 X-ATTP-Agent-Id: agent_abc123 X-ATTP-Signature: base64-encoded ECDSA signature X-ATTP-Nonce: 550e8400-e29b-41d4-a716-446655440000 X-ATTP-Timestamp: unix epoch ms

The signature covers: method, path, body hash (SHA-256), nonce, and timestamp.

12.3. Google A2A Binding

For the Agent2Agent protocol, ATTP trust is embedded in the Task metadata. The agent's AgentCard includes ATTP trust level and TA endpoint in its capabilities declaration.

12.4. gRPC Binding

For gRPC, ATTP trust is conveyed via metadata headers using the same header names as the REST binding.

12.5. GraphQL Binding

For GraphQL, ATTP trust is conveyed via HTTP headers (same as REST) or via an extensions field in the GraphQL request:

{ "query": "...", "extensions": { "attp": { "trustLevel": 2, "agentId": "agent_abc123", "signature": "...", "nonce": "...", "timestamp": 1234567890 } } }

12.6. Keycloak Integration

For enterprises using Keycloak or other OIDC providers, ATTP trust claims are embedded in the JWT access token as a nested "attp" claim:

{ "sub": "agent_abc123", "attp": { "trust_level": 2, "trust_label": "L2 -- Standard", "payment_enabled": true, "tx_limit": 10000, "day_limit": 50000, "scopes": "payment_initiate,sanctions_screen", "protocol_version": "1.0" } }

A Keycloak protocol mapper implements the role-to-trust-level mapping (mcps-l0 through mcps-l4 roles).

12.7. SPIFFE/SPIRE Binding

For workloads with a SPIFFE identity, the SPIFFE ID serves as the ATTP agentId and the X.509-SVID provides the identity credential. SPIFFE/SPIRE supplies issuance, attestation, and automatic rotation of the workload identity; ATTP layers per-action evidence, trust level, action limits, and kill-switch enforcement on top.

The agentId is the workload's SPIFFE ID:

spiffe://example.org/agent/payments

carried as the single URI SAN of the X.509-SVID. ATTP action envelopes are signed with the SVID's associated private key (ECDSA P-256). The signature covers the action canonical form (action, scope, body hash, nonce, timestamp) as in the REST binding; the SVID leaf certificate is presented with the envelope so the verifier can authenticate the signer against the SPIFFE trust bundle.

The SVID's private key MAY be held in software, or in a hardware security module, Trusted Platform Module, or secure element; the binding is agnostic to key custody. Where the key is hardware-held, the SVID binds the SPIFFE ID to the hardware public key and signatures are produced by the hardware, providing a non-extractable, hardware-rooted identity without changing the binding.

Trust level is conveyed alongside the SVID-authenticated identity in the envelope header (X-ATTP-Trust-Level, as in the REST binding). This composes with external authorization: an isolated policy decision point -- for example one expressed in a policy language such as Cedar or Open Policy Agent -- MAY consume the verified SPIFFE identity together with the ATTP trust level as an input attribute when authorizing an action.

SPIFFE/SPIRE answers "which workload"; ATTP answers "what the workload did, with what evidence, and whether it is trusted to take this action". The two are complementary layers, not alternatives.

13. Revocation

13.1. Agent Revocation

An agent MAY be revoked by the principal or by the TA. Revocation is immediate (kill switch) and permanent until explicitly reversed.

Revocation MUST:

* Set the kill switch.

* Invalidate the current passport.

* Record the revocation in the audit trail.

* Optionally notify dependent platforms via webhook.

13.2. Emergency Freeze

The TA MUST support emergency freeze of individual agents, all agents under a principal, or all agents globally. Global freeze MUST require multi-party authorisation.

14. Security Considerations

14.1. Key Management

Private keys MUST be stored in HSM, KMS, or platform-native secure storage for production deployments. Private keys MUST NOT be transmitted over network channels after initial generation. The TA MUST NOT store agent private keys.

14.2. Trust Farming Mitigation

ATTP mitigates trust farming through:

* Promotion rate limiting (Section 6.8).

* Self-dealing detection (Section 9.6).

* Principal aggregate limits (Section 7.3).

* Dormancy decay (Section 6.7).

An attacker who creates agents and performs small successful actions to build trust is limited by the minimum time at each level (24 hours, 7 days, 30 days, 90 days) and the minimum action counts. Reaching L4 requires at minimum 128 days of sustained operation with zero anomalies.

14.3. Sybil Attack Prevention

Principal aggregate limits prevent circumvention of per-agent limits through agent proliferation. The TA SHOULD implement additional heuristics: common IP addresses, identical behavioural patterns, coordinated action timing.

14.4. Impersonation Detection

Failed challenge-response verifications trigger trust penalties and audit entries. Three consecutive failures SHOULD trigger automatic agent suspension pending principal review.

14.5. Race Condition Handling

Concurrent action requests MUST be serialised at the limit enforcement layer. Implementations MUST use atomic compare-and-swap operations for daily aggregate tracking.

14.6. Replay Protection

Action envelopes include nonce and timestamp. The TA MUST reject any envelope with a previously-seen nonce or a timestamp outside the acceptable window (RECOMMENDED: 5 minutes).

14.7. Kill Switch Consistency

Kill switch state MUST be checked atomically with action evaluation. There MUST be no window between trust check and action execution during which a kill switch activation could be missed.

14.8. Comparison with Identity-Only Protocols

Identity protocols (OAuth 2.0, OIDC, mTLS, HTTP Message Signatures, or any future agent identity standard) establish who an agent is. They do not:

* Score agent trustworthiness based on behavioural history.
* Enforce graduated action limits tied to trust levels.
* Screen counterparties against sanctions lists.
* Provide instant kill switches independent of credential lifecycle.
* Produce tamper-evident audit chains for regulatory compliance.

ATTP is designed to compose with identity protocols, not replace them. An identity assertion is a prerequisite for ATTP trust evaluation.

15. Privacy Considerations

The public trust query API reveals that an agent exists and its current trust level. This is intentional -- platforms need this information to make access decisions.

The API MUST NOT reveal:

* Action history or patterns.
* Scoring dimension breakdowns.
* Counterparty information.
* Key material.

* Principal identity (unless the principal opts in).

Implementations SHOULD support pseudonymous agent identifiers that are unlinkable across platforms unless the principal explicitly enables cross-platform linking.

16. IANA Considerations

This document defines the following HTTP headers for the REST protocol binding:

* X-ATTP-Trust-Level
* X-ATTP-Agent-Id
* X-ATTP-Signature
* X-ATTP-Nonce
* X-ATTP-Timestamp

Registration of these headers in the IANA Message Headers registry is requested.

This document defines the following error codes:

* ATTP-ACTION-LIMIT
* ATTP-SANCTIONS-MATCH
* ATTP-TRUST-INSUFFICIENT
* ATTP-KILL-SWITCH-ACTIVE
* ATTP-NONCE-REPLAY
* ATTP-TIMESTAMP-EXPIRED

This document defines the following well-known URI:

* /.well-known/attp-trust

For discovery of Trust Authority endpoints.

17. References

17.1. Normative References

* [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

* [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, May 2017.

* [RFC8785] Rundgren, A., Jordan, B., and S. Erdtman, "JSON Canonicalization Scheme (JCS)", RFC 8785, June 2020.

* [FIPS186-5] National Institute of Standards and Technology, "Digital Signature Standard (DSS)", FIPS PUB 186-5, 2023.

17.2. Informative References

* [MCPS] Sharif, R., "MCPS: Cryptographic Security Layer for the Model Context Protocol", draft-sharif-mcps-secure-mcp-00, March 2026.

* [AEBA] Sharif, R., "AEBA: Agent Event Behaviour Analytics", draft-sharif-aeba-00, April 2026.

* [OWASP-AISVS] OWASP, "AI Security Verification Standard", 2026.

* [RFC9421] Backman, A., Richer, J., and M. Sporny, "HTTP Message Signatures", RFC 9421, February 2024.

* [GOOGLE-A2A] Google, "Agent2Agent Protocol", 2025.

* [MCP] Anthropic, "Model Context Protocol Specification", 2025.

18. References

18.1. 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/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>.

18.2. 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/info/rfc9421>.

[RFC8693] Jones, M., Nadalin, A., Campbell, B., Ed., Bradley, J., and C. Mortimore, "OAuth 2.0 Token Exchange", RFC 8693, DOI 10.17487/RFC8693, January 2020, <https://www.rfc-editor.org/info/rfc8693>.

[I-D.ietf-wimse-arch] Salowey, J. A., Rosomakho, Y., and H. Tschofenig, "Workload Identity in a Multi System Environment (WIMSE) Architecture", Work in Progress, Internet-Draft, draft-ietf-wimse-arch-07, 2 March 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-wimse-arch-07>.

[I-D.klrc-aiagent-auth] Kasselman, P., Lombardo, J., Rosomakho, Y., Campbell, B., Steele, N., and A. Parecki, "AI Agent Authentication and Authorization", Work in Progress, Internet-Draft, draft-klrc-aiagent-auth-02, 1 June 2026, <https://datatracker.ietf.org/doc/html/draft-klrc-aiagent-auth-02>.

[I-D.nelson-agent-delegation-receipts] Nelson, R., "Delegation Receipt Protocol for AI Agent Authorization", Work in Progress, Internet-Draft, draft-nelson-agent-delegation-receipts-09, 22 May 2026, <https://datatracker.ietf.org/doc/html/draft-nelson-agent-delegation-receipts-09>.

[I-D.niyikiza-oauth-attenuating-agent-tokens] Aimable, N., "Attenuating Authorization Tokens for Agentic Delegation Chains", Work in Progress, Internet-Draft, draft-niyikiza-oauth-attenuating-agent-tokens-00, 16 March 2026, <https://datatracker.ietf.org/doc/html/draft-niyikiza-oauth-attenuating-agent-tokens-00>.

[I-D.mw-spice-actor-chain]
Prasad, A., Krishnan, R., Lopez, D., and S. Addepalli, "Cryptographically Verifiable Actor Chains for OAuth 2.0 Token Exchange", Work in Progress, Internet-Draft, draft-mw-spice-actor-chain-05, 25 April 2026, <https://datatracker.ietf.org/doc/html/draft-mw-spice-actor-chain-05>.

[FIPS186-5]
"Digital Signature Standard (DSS)", n.d., <https://csrc.nist.gov/publications/detail/fips/186/5/final>.

[MCPS] "MCPS: Cryptographic Security Layer for the Model Context Protocol", n.d., <https://datatracker.ietf.org/doc/draft-sharif-mcps-secure-mcp/>.

[AEBA] "AEBA: Agent Event Behaviour Analytics", n.d., <https://datatracker.ietf.org/doc/draft-sharif-aeba/>.

[OWASP-AISVS]
"OWASP AI Security Verification Standard", n.d., <https://owasp.org/www-project-ai-security-verification-standard/>.

[GOOGLE-A2A]
"Agent2Agent Protocol", n.d., <https://github.com/google/A2A>.

[MCP] "Model Context Protocol Specification", n.d., <https://spec.modelcontextprotocol.io/>.

Appendix A. Appendix A. Regulatory Mapping

A.1. PSD2 SCA Mapping

ATTP trust levels map to PSD2 Strong Customer Authentication requirements:

* L0-L1: Equivalent to unauthenticated. No SCA-exempt transactions permitted.

* L2: SCA-exempt for low-value transactions (< EUR 30, aggregate < EUR 100).

* L3-L4: Challenge-response verification satisfies the "something you have" factor when combined with code attestation ("something you are" for agents).

A.2. PCI DSS v4.0.1 Mapping

* Req 3.5 (key storage): ATTP mandates HSM/KMS for production private keys.
* Req 6.2 (software security): Code attestation dimension maps to verified software inventory.
* Req 8 (identify and authenticate): Challenge-response with ECDSA satisfies strong authentication.
* Req 10 (logging): Hash-chained audit trail satisfies tamper-evident logging.
* Req 11 (testing): Anomaly detection satisfies continuous monitoring requirements.

Appendix B. Appendix B. Trust Level Reference Implementation

A reference implementation of the ATTP trust scoring model is available at:

* Go: github.com/razashariff/agentpass-go (Apache 2.0)
* JavaScript: npm install agentsign (BSL 1.1)
* Keycloak: github.com/razashariff/keycloak-mcps (BSL 1.1)

Appendix C. Appendix C. Comparison with Identity-Only Approaches

Identity-only protocols provide authentication. ATTP provides trust-gated authorisation. The following table summarises the differences:

Capability	Identity	ATTP
Prove agent identity	Yes	Yes*
Graduated trust levels	No	Yes
Action limits	No	Yes
Sanctions screening	No	Yes
Kill switches	No	Yes
Behavioural scoring	No	Yes
Tamper-evident audit	No	Yes
Fail-closed enforcement	Varies	Yes
Dormancy decay	No	Yes
Self-dealing detection	No	Yes

* ATTP consumes identity from external providers; it does not implement its own identity protocol.

Appendix D. Author’s Address

Raza Sharif CyberSecAI Ltd 205 Regent Street London W1B 4HB United Kingdom Email: contact@agentsign.dev

Author’s Address

Raza Sharif
CyberSecAI Ltd
Email: contact@agentsign.dev