Internet-Draft
draft-wang-hjs-accountability-01
Intended status: Standards Track
Expires: September 20, 2026

Y. Wang
HJS Foundation Ltd.

March 19, 2026

HJS: An Accountability Layer for AI Agents
draft-wang-hjs-accountability-01

Abstract

This document defines the Human Judgment Structure (HJS), a minimal and infrastructure窶組rade accountability layer for AI agents, built on the Judgment Event Protocol (JEP) [draft-wang-jep-judgment-event-protocol-01]. HJS provides a cryptographic mechanism to manage and verify responsibility for AI agent judgment behaviors across heterogeneous platforms. Its core design consists of four primitives (Judge, Delegate, Terminate, Verify) inherited from JEP, a minimal receipt structure with responsibility chaining, and a three窶奏ier privacy architecture for accountability governance. All advanced features (state machines, compliance proofs, governance mechanisms) are defined as optional extensions or informative appendices, preserving the structure窶冴 narrow窶層aist stability and compatibility with JEP.

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 September 20, 2026.

Copyright Notice

Table of Contents

1. Core Accountability Model (Mandatory)

1.1. Introduction

1.1.1. Problem Statement

As AI agents become increasingly autonomous and pervasive in decision-making processes across industries, there is a critical need for a standardized, verifiable accountability framework to ensure

non-repudiation, traceability, and compliance of AI-driven judgment actions. While the Judgment Event Protocol (JEP) [draft-wang-jep-judgment-event-protocol-01] provides a standard format for recording judgment events, it does not address accountability governance, responsibility chaining, or compliance with privacy regulations. This gap leads to unenforceable responsibility attribution and non-compliant data handling in cross-platform AI workflows, as illustrated by the multi-stage medical diagnosis example in [HJS-00].

1.1.2. Design Goals

HJS is designed with the following accountability-focused goals for AI agents:

* Minimal integration cost: The structure acts as a thin governance plane on top of JEP, requiring only that JEP-compliant event generators enforce accountability rules.

* Semantic accountability: HJS adds responsibility semantics to JEP primitives (e.g., Delegate = responsibility transfer, Terminate = accountability lifecycle end) while preserving JEP's neutrality.

* Non窶丧epudiability: Accountability chains are self窶祖ontained and can be verified offline without accessing the originating platform.

* Progressive deployment: Implementations can start with a minimal subset (chain verification) and later adopt advanced features as needed.

* Scope delimitation: HJS focuses solely on accountability governance for JEP-recorded AI judgment events. It does not modify JEP's core event format or signature logic.

1.1.3. Relationship to JEP and IETF Ecosystem

HJS is a complementary accountability layer to JEP, built specifically for AI agents, and integrates with existing IETF standards:

* Builds on JEP [draft-wang-jep-judgment-event-protocol-01] for core event formatting and primitive definitions.
* SCITT [I窶船.ietf-scitt-architecture] may provide timestamp anchoring for accountability receipts (optional).
* RATS [RFC9334] attestations can be embedded in Tier 2 (optional) to enhance accountability evidence for AI agents.
* Identity frameworks (OAuth [RFC6749], DID [DID-CORE], AIP [AIP-CORE]) can be used to validate actor authorization in the 'who' field (inherited from JEP).

1.1.4. Terminology and Conventions

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

Judgment: An atomic declaration made by an AI agent, possessing accountability effectiveness (as defined in JEP).

Receipt: A cryptographically verifiable record of an AI judgment event (reusing JEP event format) with added accountability metadata.

Actor: The entity (human or AI) initiating an AI judgment (inherited from JEP).

Authorization chain: A sequence of JEP receipts linked by 'based_on' fields, recording the flow of responsibility for AI agents (HJS-specific).

Cryptographic erasure: The process of rendering Tier 3 content irrecoverable by destroying decryption keys (accountability compliance for AI systems).

1.2. Model and Architecture

1.2.1. Four Primitives (Inherited from JEP)

HJS reuses JEP's four core operations and adds AI accountability semantics:

* Judge: Create a new AI judgment (JEP primitive). Generates an initial receipt with 'based_on = null'. The actor declares primary accountability for the AI decision.

* Delegate: Transfer responsibility to another AI agent/human (JEP primitive). Requires an acceptance signature from the delegatee and validation of the delegatee's authorization (HJS-specific).

* Terminate: Conclude the AI accountability lifecycle (JEP primitive). Supports two modes (HJS-specific):
  - 'ARCHIVED' (all tiers retained for audit)
  - 'ERASED' (Tier 3 content erased for compliance).

* Verify: Independently attest to the validity of an AI accountability chain (extended from JEP primitive). May be used for audit or real宴奏ime checks of responsibility attribution for AI agents.

These primitives are the only operations that modify or inspect the accountability state, and are fully compatible with JEP's event format.

1.2.2. Core Receipt Fields (Reusing JEP Event Format)

Every HJS accountability receipt MUST reuse JEP's core fields (as defined in [draft-wang-jep-judgment-event-protocol-01]) and add a 'based_on' field for AI responsibility chaining:

Field	Description (AI Accountability Semantics)
verb	JEP operation primitive: "J", "D", "T", "V"
who	AI agent/human actor identifier URI (DID/URI)
when	Unix timestamp (seconds since epoch)
what	Hash of the AI judgment content. The hash itself is public, but the actual content is stored externally under the actor宴沍 control.
based_on	Hash of the parent JEP receipt (null for initial). Provides AI accountability chain integrity.
nonce	Globally unique identifier (UUID [RFC9562]) to prevent replay attacks (inherited from JEP).
signature	Digital signature over the canonical JSON representation of all preceding fields (per JEP).
time_anchor	(Optional but RECOMMENDED) Reference to an external time宴疎nchoring service (e.g., SCITT entry ID) for AI accountability timestamp validity.

These fields answer the fundamental AI accountability questions: who (AI agent), when, what (AI decision), why (verb), based on what (responsibility chain), how to prevent replay, how to prove authenticity,

and how to secure time.

1.2.3. Three寰禅ier Privacy Architecture (HJS-Specific for AI)

JEP core fields are organized into three tiers by HJS according to AI accountability visibility requirements:

* Tier 1 (Public Governance): Contains fields that MUST be publicly accessible for basic AI accountability verification: ‘verb‘, ‘who‘, ‘nonce‘, and optionally ‘when‘. These are placed in the JWS Protected Header (per JEP’s signature format).

* Tier 2 (Logical Accountability): Contains fields required for AI audit and authorization: ‘based_on‘, ‘time_anchor‘, and any extension fields carrying audit evidence. These are placed in the JWS Payload (per JEP), which MAY be encrypted for limited access.

* Tier 3 (Private Payload): Contains the ‘what‘ field (content hash, per JEP). The hash itself is public, but the actual content is stored externally under the AI agent寰洉 control. When cryptographic erasure is performed (HJS), the key to decrypt the external content is destroyed, leaving only the hash as a tombstone for AI accountability.

This architecture ensures that each party can access only the information necessary for its AI accountability role, in line with data minimization principles for AI systems.

1.2.4. Signature and Chain Verification (Extended from JEP)

HJS reuses JEP’s JWS Compact Serialization [RFC7515] and adds chain-level verification for AI accountability:

```
BASE64URL(UTF8(JWS Protected Header)) || ’.’ ||
BASE64URL(JWS Payload) || ’.’ ||
BASE64URL(JWS Signature)
```

The Protected Header contains Tier 1 fields plus JWS寰壮pecific parameters (e.g., ‘alg‘, per JEP). The Payload contains Tier 2 and Tier 3 fields, possibly encrypted using JWE [RFC7516] (per JEP).

HJS Chain Verification Algorithm (extends JEP’s basic verification):

1. Parse the JWS structure (per JEP verification step 1).
2. Verify the signature using the actor寰洉 public key (per JEP step 2).
3. Decrypt the Payload if necessary (per JEP step 3).
4. Verify that the hash of the parent receipt matches ‘based_on‘ (HJS-specific AI chain check).
5. Check that the timestamp is within the configured tolerance window (using ‘time_anchor‘ if present, per JEP step 5).
6. Validate authorization for all Delegate events in the AI chain (HJS-specific accountability check).
7. Return ‘VALID‘ if all checks pass, otherwise ‘INVALID‘ or ‘DISPUTED‘.

This process requires no online dependencies beyond the public key infrastructure needed to resolve the ‘who‘ URI (per JEP) and authorization lists (HJS-specific for AI agents).

1.3. Cryptographic Erasure (AI Accountability Compliance)

To support privacy regulations such as the GDPR "right to be forgotten" (a core AI accountability requirement), HJS defines a cryptographic erasure mechanism for JEP receipts. When a ‘Terminate‘ operation is performed with mode ‘ERASED‘, the AI agent destroys the encryption key that protects the external content referenced by the ‘what‘ hash (per JEP’s ‘what‘ field definition). The JEP receipt remains, but the AI decision content becomes computationally

unrecoverable. A Proof of Key Destruction (PKD) MAY be included in the receipt to provide evidence of compliance with AI accountability obligations. The three寰奏ier architecture ensures that even after erasure, Tier 1 and Tier 2 data remain intact for AI audit purposes.

1.4. Security Considerations

1.4.1. Threat Model

HJS assumes the same semi寰奏rusted network environment as JEP and defends against additional AI accountability-specific threats:

* Insider attackers: Malicious AI agents/humans delegating to evade responsibility or forging authorization evidence (HJS-specific).
* External attackers: Intercepting and replaying receipts (inherited from JEP), or traffic analysis from Tier 1 metadata (HJS-specific).
* Computational threats: Future quantum attacks against current signature algorithms (shared with JEP).

Trust assumptions: Actors are responsible for private key security (per JEP); time sources are partially trusted within a tolerance window (per JEP); optional anchoring services (e.g., SCITT) are trusted for AI accountability timestamp validity (HJS-specific).

1.4.2. Key Management

* Signature keys SHOULD be stored in hardware trust roots (HSM, TEE) and rotated periodically (RECOMMENDED yearly, per JEP).
* Encryption keys for Tier 3 AI content (HJS-specific) MAY be managed externally and MUST be destroyed for ‘ERASED‘ termination mode.
* In case of key compromise, the affected actor MUST issue a ‘Terminate‘ receipt (per JEP’s ‘Terminate‘ primitive) to logically invalidate all receipts signed with that key after a certain timestamp (HJS-specific AI accountability rule).

1.4.3. Algorithm Agility

HJS inherits JEP’s support for multiple signature algorithms, including Ed25519, ECDSA P寰56, SM2, and future postquantum algorithms (e.g., Dilithium, Falcon). Implementations SHOULD monitor NIST standardization progress and be prepared to migrate when necessary to maintain long-term AI accountability chain integrity.

2. Optional Accountability Features (Optional Implementation)

This section describes accountability-enhancing features for AI agents that are not required for basic interoperability with JEP but can be implemented to enhance governance functionality.

2.1. Simplified State Machine

2.1.1. Basic States

Implementations that wish to track the AI accountability lifecycle of a JEP-recorded judgment MAY use the following minimal state machine:

* ACTIVE: AI judgment created, responsibility held by current actor.
* DELEGATING: Delegation initiated, awaiting acceptance (HJS-specific).
* DELEGATED: Responsibility successfully transferred (HJS-specific).
* TERMINATED: Lifecycle ended (archived or erased, per HJS).

2.1.2. Transition Rules

* ‘Judge‘ (JEP primitive) → ACTIVE
* ‘Delegate‘ (initiation, JEP primitive) → DELEGATING
* ‘Delegate‘ (acceptance, JEP primitive) → DELEGATED (if in DELEGATING)

* ‘Terminate‘ (JEP primitive) → TERMINATED (from ACTIVE, DELEGATING, or DELEGATED)
* Timeout: If no acceptance is received within a configurable window, DELEGATING automatically reverts to ACTIVE (HJS-specific).

These rules ensure basic AI accountability lifecycle tracking without introducing dispute or governance complexity.

2.2. Extension Field Format

2.2.1. Extension Object

HJS receipts (reusing JEP event format) MAY include an ‘extensions‘ object at the top level (in the JWS Payload) to carry additional AI accountability data. Each extension is identified by a URI and contains a value whose semantics are defined by the extension specification.

Example:

```json
{
  "extensions": {
    "org.example.hjs.tee-evidence": { ... },
    "org.example.hjs.gdpr-proof": { ... }
  }
}
```

2.2.2. IANA Registration Requirements

HJS extensions MUST be registered in the "HJS Extensions" registry with the following metadata (compatible with JEP’s extension registry requirements):

Extension Identifier: URI
Semantics Description: URL to specification
Compatibility Level: "full", "wire", or "none"
Version: Semantic version
Registration Date
Contact

Compatibility level indicates how the extension interacts with implementations that do not recognize it:

full: Can be safely ignored; core AI accountability semantics unchanged.
wire: Adds new fields that may be ignored but could affect interoperability if critical for AI accountability.
none: Modifies core AI accountability semantics; requires explicit support.

3. Appendices (Informative)

The following appendices provide additional guidance, examples, and specifications for optional AI accountability extensions. They are not part of the core HJS model but are compatible with JEP’s optional features.

Appendix A. Complete State Machine and Dispute Handling

A.1. Full State Definitions

For systems requiring sophisticated AI accountability lifecycle management, the following states can be added to the simplified state machine:

DISPUTED_FROZEN: Conflict detected (e.g., dualmode verification), AI accountability chain frozen pending resolution.
RESOLVED_CONTINUE: Dispute resolved, chain continues.
RESOLVED_INVALID: Dispute resolved by invalidation; chain must be reinitiated.

A.2. Dispute Detection and Resolution

Disputes are detected when dualmode verification yields inconsistent AI accountability results (platform signature valid but open verification fails, or vice versa). Upon detection, the chain enters DISPUTED_FROZEN and a resolution process is triggered. Resolution can be manual or automated using the $Resolve extension (Appendix D.9), which is HJS-specific and compatible with JEP events.

A.3. Federal Governance Extension

For multiparty AI accountability governance, the Federal Resolution extension (Appendix D.9) allows configuring participants with weights and a threshold. Resolutions require a weighted signature set meeting the threshold, ensuring collective accountability for AI dispute outcomes.

Appendix B. Advanced Verification Modes

B.1. Platform Mode

In platform mode, AI accountability verification relies on a platformissued signature in addition to JEP’s cryptographic proof. This mode offers low latency (<10 ms) and is suitable for hightrust AI accountability environments.

B.2. Dual Mode

Dual mode performs both platform and open AI accountability verification simultaneously (extending JEP’s basic verification). If results conflict, the receipt is marked DISPUTED. This mode provides maximum assurance and detects platform compromise or misconfiguration that could break AI accountability.

B.3. MultiSource Time Consensus

To mitigate single timesource failures (critical for AI accountability timestamp validity), implementations MAY collect timestamps from multiple independent sources (e.g., NTS, SCITT, hardware clock) and use a consensus algorithm (e.g., median) to determine authoritative time for JEP receipts.

Appendix C. Compliance and Regulatory Guidance

C.1. GDPR Mapping

HJS’s threetier architecture supports GDPR data minimization (Article 5) by design, extending JEP’s neutral event format with AI accountability controls. Cryptographic erasure (Article 17) is implemented via the ERASED termination mode and Proof of Key Destruction (HJS-specific). For full compliance, deployers SHOULD enable the GDPR Compliance Proof extension (Appendix C.4), which adds AI accountability evidence for GDPR obligations.

C.2. EU AI Act Mapping

For highrisk AI systems, HJS’s ‘based_on‘ field (extending JEP’s event format) enables traceability (Article 12 of the EU AI Act). The AI Act Transparency Marking extension (Appendix C.5) can record risk level, human oversight, and conformity assessment identifiers to meet AI accountability requirements for high-risk AI.

C.3. Jurisdiction Marking Extension

This HJS-specific extension allows receipts (reusing JEP format) to declare applicable law, data localization, and retention period—key AI accountability requirements for cross-border AI systems. See Appendix D.8

for details.

C.4. GDPR Compliance Proof Extension

Adds fields for DPO signature, erasure proof (PKD, timestamp, witness), and audit trail—HJS-specific AI accountability evidence for GDPR compliance. See Appendix D.8.

C.5. AI Act Transparency Marking Extension

Adds fields for risk level, transparency obligation, synthetic content flag, human supervisor ID, and conformity assessment—HJS-specific AI accountability metadata for EU AI Act compliance. See Appendix D.8.

Appendix D. Concrete Extension Definitions

D.1. TEE Evidence Format (Priority 1)

Identifier: org.ietf.hjs.ext.tee-evidence
Compatibility: wire
Purpose: Standardize TEE evidence in Tier 2 for enhanced AI accountability.
Fields: tee_type, format, evidence, verifier, timestamp.

D.2. Lightweight Mode (Priority 2)

Identifier: org.ietf.hjs.ext.lightweight
Compatibility: wire
Purpose: Reduce receipt size for constrained environments while
preserving core AI accountability.
Features: Compression, partial chain proof, open mode only.

D.3. AIP Integration (Priority 2)

Identifier: org.ietf.hjs.ext.aip
Compatibility: wire
Purpose: Bind HJS AI accountability chains to Agent Identity Protocol
sessions for actor authorization validation.
Fields: aip_session, aip_capability.

D.4. Batch Operations (Priority 2)

Identifier: org.ietf.hjs.ext.batch
Compatibility: none
Purpose: Atomically process multiple AI accountability primitives.
Adds $Batch primitive with operations array (compatible with JEP events).

D.5. Refusal Event (Priority 2)

Identifier: org.ietf.hjs.ext.refusal
Compatibility: wire
Purpose: Record AI system refusals with AI accountability metadata.
Fields: refusal.reason, policy_ref, input_hash.

D.6. Key Evolution (Priority 3)

Identifier: org.ietf.hjs.ext.key-evolution
Compatibility: wire
Purpose: Forwardsecrecy via key rotation for long-term AI accountability.
Adds ROTATE_KEY event with rotation proof (compatible with JEP's Verify primitive).

D.7. Enhanced Hardware Binding (Priority 3)

Identifier: org.ietf.hjs.ext.enhanced-hardware
Compatibility: wire
Purpose: Strengthen AI accountability via VDF and biometrics.
Supports binding_level, VDF proofs, biometric signals.

D.8. Legal Validity Proof (Priority 4)

Identifier: org.ietf.hjs.ext.legal-proof
Compatibility: full
Purpose: Enhance admissibility of HJS AI accountability chains with notary/regulator signatures.
Fields: regulation, witnesses, pkd, audit_trail.

D.9. Federal Resolution (Priority 5)

Identifier: org.ietf.hjs.ext.federal-resolution
Compatibility: full
Purpose: Multiparty governance for AI dispute resolution (HJS-specific).
Fields: governance_model, participants, threshold_policy.

Appendix E. Implementation Guide and Best Practices

E.1. Minimal Implementation (Pseudocode)

```python
# HJS implementation extending JEP's core event class
class HJSReceipt(JEPReceipt):
    def __init__(self, verb, who, when, what, based_on, nonce):
        super().__init__(verb, who, when, what, nonce)  # Reuse JEP
        self.based_on = based_on  # HJS-specific AI chain field

    def verify_chain(self, public_key, authorization_list):
        # Step 1: Verify JEP signature (inherited)
        if not self.verify(public_key):
            return "INVALID"
        # Step 2: Verify AI accountability chain (HJS-specific)
        if self.based_on and not self.validate_parent(self.based_on):
            return "INVALID"
        # Step 3: Verify authorization (HJS-specific for AI)
        if self.verb == "D" and not self.check_authorization(authorization_list):
            return "INVALID"
        # Step 4: Verify erasure proof if terminated (HJS-specific)
        if self.verb == "T" and self.mode == "ERASED" and not self.validate_pkd():
            return "INVALID"
        return "VALID"
```

E.2. Performance Benchmarks

Reference implementation (Rust) on a 16core server:

- Judge generation: 4.5 ms average
- Throughput: 12,000 receipts/second
- Open chain verification: <5 ms
- Dual verification: <100 ms
- Receipt size: 1.53 KB (core)

E.3. Test Vectors

(Example AI accountability receipt and verification results; refer to [HJS-00] for full test vectors.)

E.4. CrossDomain Deployment Recommendations

- Use ephemeral DIDs for public AI deployments, rotated daily.
- For crossborder transfers, enable Jurisdiction Marking.
- For highrisk AI, enable TEE Evidence and dual mode.

Appendix F. Examples

F.1. Judge Receipt Example (AI Agent)

```json
{
  "hjs": "1.0",
  "verb": "J",
  "who": "did:example:ai-agent-123",
  "when": 1742345678,
  "what": "1220a1b2c3d4e5f67890abcdef0123456789",
  "based_on": null,
  "privacy": "AUDIT",
  "status": "ACTIVE",
  "nonce": "f47ac10b-58cc-4372-a567-0e02b2c3d479",
  "aud": "https://platform.example.com",
  "sig": "eyJhbGciOiJFZERTQSJ9..."
}
```

F.2. Error Code Reference

```
 INVALID_SIGNATURE: Signature verification failed.
 BROKEN_CHAIN: Parent hash mismatch (AI accountability chain broken).
 UNAUTHORIZED_DELEGATION: Delegate not in AI authorization chain.
 INVALID_TERMINATION: Missing dual signature in DELEGATED state.
 PAYLOAD_ERASED: Tier 3 AI content erased.
 CHAIN_TOO_DEEP: Delegation depth exceeds limit.
 EXPIRED_RECEIPT: Timestamp outside window.
 DISPUTED: Dualmode conflict in AI accountability.
 RESOLUTION_REQUIRED: Governance resolution applied.
```

Appendix G. References

G.1. Normative References

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

[RFC7515] Jones, M., "JSON Web Signature (JWS)", RFC 7515, May 2015.

[RFC7516] Jones, M. and J. Hildebrand, "JSON Web Encryption (JWE)", RFC 7516, May 2015.

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

[RFC8785] Rundgren, A., et al., "JSON Canonicalization Scheme (JCS)", RFC 8785, June 2020.

[RFC9334] Birkholz, H., et al., "Remote ATtestation procedureS (RATS) Architecture", RFC 9334, January 2023.

[RFC9122] Maheshwari, H., "Multihash Format", RFC 9122, July 2021.

[RFC9562] Davis, D., et al., "Universally Unique IDentifiers (UUIDs)", RFC 9562, May 2024.

[draft-wang-jep-judgment-event-protocol-01]
Wang, Y., "Judgment Event Protocol (JEP)", draft-wang-jep-judgment-event-protocol-01 (Work in Progress), March 2026.

G.2. Informative References

[HJS-00] Wang, Y., "HJS: An Accountability Layer for AI Agents", draft-wang-hjs-accountability-00, February 2026.

[I-D.ietf-scitt-architecture] Birkholz, H., et al., "Supply Chain Integrity, Transparency, and Trust (SCITT) Architecture", Work in Progress.

[DID-CORE] Sporny, M., et al., "Decentralized Identifiers (DIDs) v1.0", W3C Recommendation, July 2022.

[AIP-CORE] Agent Identity Protocol Working Group, "Agent Identity Protocol (AIP) Core Specification", 2025, https://aip.dev/.

[GDPR-2016] European Parliament, "General Data Protection Regulation (GDPR)", 2016.

[EU-AI-ACT] European Commission, "Artificial Intelligence Act", 2024.

Author’s Address

Yuqiang Wang
HUMAN JUDGMENT SYSTEMS FOUNDATION LTD.
Email: signal@humanjudgment.org
GitHub: https://github.com/hjs-spec