Cryptographic Binding of AI Agent Identity to Stated Intent

Ramprasad Anandam Gaddam
Independent Researcher
Gurgaon, India
ram@vouch-protocol.com

January 2026

Abstract

This paper presents the Vouch Protocol, a cryptographic framework for binding AI agent identity to stated intent using Ed25519 digital signatures in JSON Web Signature (JWS) format. Unlike existing authentication mechanisms—bearer tokens, API keys, OAuth 2.0, and mutual TLS—which provide connection-level or session-level authentication, the Vouch Protocol enables request-level attestation with cryptographic non-repudiation. We formalize the problem of intent binding in autonomous agent systems, define the protocol specification, and analyze its security properties. The protocol addresses the growing need for verifiable action attestation as AI agents increasingly operate autonomously in high-stakes environments. A reference implementation is publicly available.[1]

Keywords: AI agents, digital signatures, authentication, non-repudiation, decentralized identity, intent binding

1 Introduction

The proliferation of autonomous AI agents in production environments has created a fundamental authentication gap. These agents—ranging from automated trading systems to AI coding assistants—perform actions with significant consequences, yet existing authentication mechanisms fail to provide verifiable proof of both *identity* (who authorized the action) and *intent* (what specific action was authorized).

Consider an AI agent that claims to be "Alice's Shopping Agent" and requests a $500 purchase. A receiving service must answer three critical questions:

1. **Identity verification:** Is this agent genuinely authorized by Alice?
2. **Intent binding:** Did the agent specifically authorize *this* purchase?
3. **Non-repudiation:** Can Alice later deny authorization?

Current authentication mechanisms, designed primarily for human users or static service-to-service communication, fail to address these requirements adequately.

1.1 Contributions

This paper makes the following contributions:

- A formal problem statement for identity-intent binding in AI agent systems

[1] This paper is part of the Vouch Protocol Defensive Disclosure Series (PAD-001). Full series: https://vouch-protocol.com/docs/disclosures/

- The Vouch Protocol specification for cryptographic intent attestation
- Security analysis demonstrating achieved properties
- Comparison with existing authentication mechanisms
- Reference implementation and deployment considerations

2 Background and Preliminaries

2.1 Cryptographic Primitives

Definition 1 (Digital Signature Scheme)**.** *A digital signature scheme consists of three algorithms:*

- $\mathsf{KeyGen}() \to (sk, pk)$*: Generates a secret key* sk *and public key* pk
- $\mathsf{Sign}(sk, m) \to \sigma$*: Produces signature* σ *on message* m
- $\mathsf{Verify}(pk, m, \sigma) \to \{0, 1\}$*: Verifies signature validity*

The Vouch Protocol employs Ed25519 [Josefsson and Liusvaara, 2017], an Edwards-curve Digital Signature Algorithm providing 128-bit security with compact 64-byte signatures.

Definition 2 (JSON Web Signature)**.** *A JSON Web Signature (JWS) [Jones et al., 2015a] represents signed content as a three-part structure:*

$$\mathsf{JWS} = \mathsf{Base64Url}(\mathsf{header}) \,||\, \text{`.'} \,||\, \mathsf{Base64Url}(\mathsf{payload}) \,||\, \text{`.'} \,||\, \mathsf{Base64Url}(\sigma) \tag{1}$$

Definition 3 (Decentralized Identifier)**.** *A Decentralized Identifier (DID) [Sporny et al., 2022a] is a URI that resolves to a DID Document containing public keys and service endpoints:*

$$\mathsf{DID} \xrightarrow{\mathsf{resolve}} \mathsf{DID\ Document} \ni pk \tag{2}$$

2.2 Threat Model

We consider an adversary $\mathcal{A}$ with the following capabilities:

- Passive observation of network traffic
- Active message injection and modification
- Possession of valid tokens from previous sessions
- Compromise of non-target agent credentials

The adversary's goal is to impersonate a legitimate agent or cause unauthorized actions to be attributed to a victim agent.

3 Analysis of Existing Mechanisms

We analyze four prevalent authentication mechanisms with respect to identity binding, intent binding, non-repudiation, replay resistance, and support for autonomous operation.

3.1 Bearer Tokens

Bearer tokens [Hardt, 2012] grant access to any party possessing the token string.

Definition 4 (Bearer Token Authentication)**.** *Authentication succeeds if* $\mathsf{token} \in \mathsf{ValidTokens}$*, regardless of presenter identity.*

Limitations:

- No identity binding: Tokens are not cryptographically tied to holder
- No intent binding: Token authorizes any action within scope
- No non-repudiation: Token possession is shared knowledge
- Replay vulnerability: Intercepted tokens remain valid

3.2 API Keys

API keys are shared secrets known to both client and server.

Limitations:

- Shared secret paradigm: Key exposure compromises all holders
- Server-side storage: Requires trusted key management
- No third-party verification: Only the issuing server can authenticate

3.3 OAuth 2.0

OAuth 2.0 [Hardt, 2012] provides delegated authorization through user-interactive flows.

Limitations for AI agents:

- Human interaction required: Authorization flows expect browser-based consent
- Coarse-grained scopes: "access:calendar" vs. "create:meeting:3pm:tuesday"
- Centralized dependency: Requires OAuth provider availability

3.4 Mutual TLS

Mutual TLS [Rescorla, 2018] provides bilateral certificate-based authentication at the transport layer.

Limitations:

- Connection-level: Authenticates the channel, not individual requests
- No intent binding: Certificate does not specify authorized actions
- Certificate management complexity: Rotation and revocation overhead

3.5 Comparative Analysis

Table 1 summarizes the authentication properties.

4 The Vouch Protocol

4.1 Protocol Overview

The Vouch Protocol binds agent identity to specific intent through cryptographic attestation. Each request includes a signed token containing the precise action the agent authorizes.

Table 1: Authentication mechanism comparison

Property	Bearer	API Key	OAuth	mTLS	Vouch
Identity binding	×	×	∼	✓	✓
Intent binding	×	×	×	×	✓
Non-repudiation	×	×	×	∼	✓
Replay resistance	×	×	∼	×	✓
Autonomous operation	✓	✓	×	✓	✓
Decentralized	✓	✓	×	×	✓

4.2 Token Structure

Definition 5 (Vouch Token). *A Vouch token is a JWS with the following payload structure:*

```
{
  "jti": "<unique-identifier>",
  "iss": "<agent-DID>",
  "iat": <issued-timestamp>,
  "exp": <expiration-timestamp>,
  "vouch": {
    "version": "1.0",
    "payload": {
      "action": "<action-identifier>",
      ...action-specific-parameters
    }
  }
}
```

Field semantics:

- `jti`: JWT ID for replay prevention (UUID recommended)
- `iss`: Issuer DID resolving to agent's public key
- `iat/exp`: Temporal validity window (typically 60-300 seconds)
- `vouch.payload`: The intent being attested

4.3 Signing Process

The agent constructs and signs the token as follows:

$$\text{token} = \mathsf{JWS.Sign}(sk_{\mathsf{agent}}, \mathsf{payload}) \tag{3}$$

where sk_{agent} is the agent's Ed25519 private key.

4.4 Verification Process

Verification Algorithm:

1. Parse JWS to extract header, payload, signature
2. Extract `iss` (DID) from payload
3. Resolve DID to obtain public key pk
4. Verify: $\mathsf{Ed25519.Verify}(pk, \mathsf{payload}, \sigma) = 1$

5. Check: $\mathsf{now} < \mathsf{exp}$
6. Check: $\mathsf{jti} \notin \mathsf{SeenTokens}$
7. Add jti to $\mathsf{SeenTokens}$
8. Return $(\mathsf{iss}, \mathsf{vouch.payload})$

5 Security Analysis

Property 1 (Authentication)**.** *A valid Vouch token cryptographically proves the agent possessing sk authorized the request.*

Argument: By the unforgeability of Ed25519 under chosen-message attacks, an adversary without sk cannot produce a valid signature.

Property 2 (Intent Binding)**.** *The signed payload includes the complete action specification, binding identity to specific intent.*

Argument: The `vouch.payload` is included in the signed message. Modifying any parameter invalidates the signature.

Property 3 (Non-Repudiation)**.** *An agent cannot deny having authorized a request for which a valid signature exists.*

Argument: Only the holder of sk can produce valid signatures. The signature serves as cryptographic evidence.

Property 4 (Replay Resistance)**.** *Tokens cannot be replayed due to unique `jti` and expiration enforcement.*

Argument: Verifiers maintain $\mathsf{SeenTokens}$ set and reject duplicates. Short expiration windows bound the replay window.

6 Related Work

The Vouch Protocol builds upon and differentiates from several bodies of prior work.

6.1 Token-Based Authentication

JSON Web Tokens (JWT) [Jones et al., 2015b] provide a standardized format for claims between parties. The IETF has extensively documented JWT security considerations. Our work extends JWTs with the `vouch` claim for explicit intent binding—a concept not present in standard JWT usage patterns.

Macaroons, introduced by Birgisson et al., provide contextual caveats for capability-based authorization. While macaroons support chained attenuation, they do not provide non-repudiation as signatures are HMAC-based (symmetric).

6.2 Decentralized Identity

The W3C Decentralized Identifiers (DIDs) specification [Sporny et al., 2022a] enables self-sovereign identity. DID methods like `did:web` and `did:key` provide varying tradeoffs between resolution complexity and decentralization. Our protocol is DID-method agnostic, supporting any resolvable DID.

Verifiable Credentials [Sporny et al., 2022b] provide a data model for tamper-evident credentials. While VCs focus on issued credentials ("Alice has a degree"), Vouch focuses on action attestation ("Alice's agent authorized this transfer").

6.3 Workload Identity

SPIFFE (Secure Production Identity Framework for Everyone) provides workload identity through SVIDs (SPIFFE Verifiable Identity Documents). SPIRE implements the SPIFFE specification for Kubernetes and VM environments. These systems provide strong identity but lack per-request intent binding.

6.4 Code Signing and Attestation

Sigstore provides keyless signing for software artifacts using OIDC identity. In-toto provides software supply chain attestations. These systems focus on software artifacts rather than runtime action authorization.

6.5 Agent Authentication Protocols

Prior work on agent authentication, such as the Agent Communication Language (ACL) from FIPA, focused on message interchange formats rather than cryptographic verification. Recent work on LLM agent architectures has not yet addressed systematic authentication.

7 Limitations and Discussion

7.1 Key Management

The protocol requires agents to securely store private keys. Key compromise enables impersonation until revocation. Companion work (PAD-003) addresses this through an identity sidecar pattern.

7.2 DID Resolution Trust

Verification requires trusting the DID resolution mechanism. Compromised DID documents could enable impersonation. Caching and multi-resolution strategies can mitigate availability and integrity concerns.

7.3 Adoption Bootstrapping

New identity systems face a "cold start" problem. To address this, companion work (PAD-008) enables bootstrapping from existing SSH keys registered with GitHub, providing zero-friction adoption.

7.4 Scope

This protocol addresses authentication and intent binding. Authorization policies (determining *whether* an authenticated agent *should* perform an action) remain the responsibility of consuming services.

8 Implementation

A reference implementation is available at `https://github.com/vouch-protocol/vouch`, providing:

- Python library: `pip install vouch-protocol`
- CLI tools: `vouch sign`, `vouch verify`

- Browser extension for signature verification
- Integration with Git for commit signing

The implementation has been tested with Ed25519 keys and supports `did:web` and `did:key` resolution methods.

9 Conclusion

This paper presented the Vouch Protocol, addressing the authentication gap for autonomous AI agents. By cryptographically binding identity to intent, the protocol provides verifiable action attestation with non-repudiation. The protocol builds on established standards (JWT, JWS, DIDs) while introducing the novel `vouch` claim for explicit intent binding.

Future work includes delegation chain support (PAD-002), sidecar architecture patterns (PAD-003), and hybrid identity bootstrapping from existing SSH keys (PAD-008).

Prior Art Declaration

This document is published as a defensive prior art disclosure under the Creative Commons CC0 1.0 Universal Public Domain Dedication. The methods and systems described herein are hereby released into the public domain to prevent patent monopolization. Any party implementing similar functionality after the publication date of this document cannot claim novelty for patent purposes.
Reference Implementation: https://github.com/vouch-protocol/vouch

Cryptographically Signed Document

Signed by: Ramprasad Anandam Gaddam (`github:rampyg`)
Verify: https://v.vouch-protocol.com/p/pad001

This document's authenticity can be verified by computing its SHA-256 hash and checking against the signature registered at the verification URL above.

References

D. Hardt. The OAuth 2.0 Authorization Framework. RFC 6749, IETF, October 2012. URL https://www.rfc-editor.org/rfc/rfc6749.

M. Jones, J. Bradley, and N. Sakimura. JSON Web Signature (JWS). RFC 7515, IETF, May 2015a. URL https://www.rfc-editor.org/rfc/rfc7515.

M. Jones, J. Bradley, and N. Sakimura. JSON Web Token (JWT). RFC 7519, IETF, May 2015b. URL https://www.rfc-editor.org/rfc/rfc7519.

S. Josefsson and I. Liusvaara. Edwards-Curve Digital Signature Algorithm (EdDSA). RFC 8032, IETF, January 2017. URL https://www.rfc-editor.org/rfc/rfc8032.

E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446, IETF, August 2018. URL https://www.rfc-editor.org/rfc/rfc8446.

Manu Sporny, Dave Longley, Markus Sabadello, Drummond Reed, Orie Steele, and Christopher Allen. Decentralized Identifiers (DIDs) v1.0. Recommendation, W3C, July 2022a. URL https://www.w3.org/TR/did-core/.

Manu Sporny, Grant Noble, Dave Longley, Daniel Burnett, Brent Zundel, and Kyle Den Hartog. Verifiable Credentials Data Model v1.1. Recommendation, W3C, March 2022b. URL https://www.w3.org/TR/vc-data-model/.