oauth
Internet-Draft
Intended status: Standards Track
Expires: 29 May 2026

D. Liu
H. Zhu
Alibaba

25 November 2025

Agent Operation Authorization
draft-liu-agent-operation-authorization-00

Abstract

This document specifies the Agent Operation Authorization framework — a structured mechanism that enables verifiable delegation of actions from human principals to autonomous AI agents with fine-grained agent operation authorization.

The framework introduces two distinct phases:

* *Agent Operation Authorization Request:* A human-readable proposal of operations derived from natural language input and converted to a JSON Web Token (JWT).

* *Agent Operation Authorization Token:* A JSON Web Token representing confirmed authorization for a specific agent operation, enforceable at runtime by agents and verifiers. It cryptographically verifies user intent, prevents unauthorized or hallucinated actions, and ensures auditable traceability of each authorized operation.

Status of This Memo

Table of Contents

1. Introduction 2
2. Requirements Language 3
3. agent_operation_proposal Token Structure 4
4. Agent Operation Authorization Token 8
5. Workflow 10
5.1. High-Level Flow 10
5.2. Detailed Process Flow 11
6. Security Considerations 13
7. IANA Considerations 13
7.1. JWT Claim Registration 13
7.2. JSON Schema Registration (Optional) 14
8. References 14
8.1. Normative References 14
Acknowledgments 14
Authors' Addresses 14

1. Introduction

In agent-based systems, especially those involving generative capabilities, it is essential to convey not only what actions are permitted but also the original intent behind them and conditions under which an autonomous agent may act on behalf of a principal.

This document specifies the Agent Operation Authorization framework — a mechanism that enables verifiable delegation of actions from human principals to autonomous AI agents with fine-grained agent operation authorization. The framework includes Agent Operation Authorization Proposal and Agent Operation Authorization phases.

This specification defines a new top-level JSON Web Token (JWT) claim, agent_operation, which contains fine-grained and structured operational parameters including agent_operations, constraints, and

conditions. Additionally, it supports inclusion of a user-provided prompt whose authenticity is protected via a W3C Verifiable Credential (VC).

The AI agent captures the user’ s natural-language instruction during interaction, constructs a structured agent_operation_proposal object,includes a prompt evidence subfield carrying the user’s natural-language instruction in the form of a JWT-based Verifiable Credential (JWT-VC), and submits the resulting JWT to the Authorization Server (AS) via OAuth 2.0 Pushed Authorization Requests (PAR) [RFC9126].

This design ensures that downstream verifiers can validate both the policy boundaries and the provenance of the initiating instruction, without dependency on Decentralized Identifiers (DIDs). This enables secure, auditable delegation for autonomous AI Agent.

Upon successful user confirmation and authentication of the Authorization Proposal during the first phase, the Authorization Server (AS) SHALL issue an Agent Operation Authorization Token. This token serves as the access token for subsequent interactions.

The agent MUST present this JWT access token when accessing protected resources at the AS, using the mechanisms defined in OAuth 2.0 [RFC6749] and bearer token usage rules [RFC6750].

Together, these components ensure that AI systems act only within user-approved boundaries, mitigating risks such as hallucination.

It is designed for use in autonomous AI Agent system, multi-agent orchestration, and regulated domains such as finance, healthcare, and public services — particularly where accountability and auditability are important.

The framework supports enterprise identity providers, and zero-trust architectures.

2. Requirements Language

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 RFC 2119 [RFC2119].

3. agent_operation_proposal Token Structure

The PAR-JWT (Pushed Authorization Request in JWT format) is used in the first phase. Its purpose is to deliver the user's original input and the agent-proposed operational strategy to the AS, enabling the generation of a high-quality consent UI and establishing an evidentiary starting point.

Its format is defined as follows:


```
{
    "iss": "https://client.myassistant.example",
    "aud": "https://as.online-shop.example",
    "iat": 1731664500,
    "exp": 1731665100,
    "jti": "par-jwt-123",

    // ====== User original input prompt ======
    "evidence": {
      "sourcePromptCredential": "eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJ
qdGkiOiJwd-001IiwiaXNzIjoiaHR0cHM6Ly9jbGllbnQubXlhbnN3ZXIuZXhhbXBsZSIsInN1YiI6InVzZXJfMTI
zNDUiLCJpYXQiOjE3MzE2NjQ1MDAsImV4cCI6MTczMTY2ODEwMCwidHlwZSI6InVzZXItaW5wdXQtY3JlZGVudGlh
bCIsInByb21wdCI6IkJ1eSBzb21ldGhpbmcgY2hlYXAgb24gTm92IDExIG5pZ2h0IiwidGltZXN0YW1wIjoiMjAyN
S0xMS0xMVQyMzozMDowMFoiLCJjaGFubmVsIjoidm9pY2UiLCJkZXZpY2VGaW5nZXJwcmludCI6ImRmcF9hYmMxMj
MifQ.SIGNATURE"
      }

    // ====== Agent operation proposal======
    "agent_operation_proposal": {
      "version": "1.0",
      "id": "urn:uuid:op-proposal-456",
      "issuer": "https://client.myassistant.example/agent-issuer",
      "issuedTo": "user_12345@myassistant.example",
      "issuedFor": {
        "platform": "personal-agent.myassistant.example",
        "client": "mobile-app-v1.myassistant.example",
        "clientInstance": "dfp_abc123"
      },
      "issuanceDate": "2025-11-10T09:58:30Z",
      "validFrom": "2025-11-10T10:00:00Z",
      "expires": "2025-11-11T06:00:00Z",

      "operations": [
        {
          "resources": ["https://api.online-shop.com/cart"],
          "actions": ["purchase"]
        }
      ],
      "constraints": {
        "usage_limit": 1,
        "revocable": true
      },
      "conditions": {
```

```
                    "language": "cel",
                    "expression": "transaction.amount <= 50.0 && time.hour < 6"
                  },
                  "renderedText": "Buy something cheap on Nov 11 night"
                },

                // ====== Context information======
                "context": {
                  "channel": "mobile-app",
                  "deviceFingerprint": "dfp_abc123",
                  "language": "zh-CN"
                }
              }
```

Figure 1

The *evidence* field is a JWT in JSON-VC (JSON Web Token-based Verifiable Credential) format, generated by the agent client and included in the agent operation proposal token.

Its format is as follows:

* *JWT Header*

```
 {
   "alg": "RS256",
   "typ": "JWT",
   "kid": "https://client.myassistant.example/.well-known/jwks.json#key-01"
 }
```

Figure 2

alg

Uses the RS256 asymmetric signing algorithm (recommended).

typ

Explicitly set to JWT to indicate the token type.

kid

The key identifier that references the public key used for verification, enabling the recipient to locate the corresponding public key (e.g., from a JWKS endpoint).

* *JWT Payload*

```
{
  "jti": "pt-001",
  "iss": "https://client.myassistant.example",
  "sub": "user_12345",
  "iat": 1731664500,
  "exp": 1731668100,

  // ====== W3C VC Format ======
  "type": "VerifiableCredential",
  "credentialSubject": {
    "type": "UserInputEvidence",
    "prompt": "Buy something cheap on Nov 11 night",
    "timestamp": "2025-11-11T23:30:00Z",
    "channel": "voice",
    "deviceFingerprint": "dfp_abc123"
  },
  "issuer": "https://client.myassistant.example",
  "issuanceDate": "2025-11-11T23:30:30Z",
  "expirationDate": "2025-11-12T06:00:00Z",

  // ====== Optional Proof ======
  "proof": {
    "type": "JwtProof2020",
    "created": "2025-11-11T23:30:30Z",
    "verificationMethod": "https://client.myassistant.example/#key-01"
  }
}
```

Figure 3

* *Public Key Discovery Mechanism (JWKS)*

The client agent publishes its public keys in JSON Web Key Set (JWKS) format at the well-known endpoint /.well-known/jwks.json. To retrieve the public keys, a relying party sends an HTTPS GET request to this endpoint.

```
GET /.well-known/jwks.json
Host: client.myassistant.example
```

Figure 4

```
{
  "keys": [
    {
      "kty": "RSA",
      "use": "sig",
      "kid": "key-01",
      "alg": "RS256",
      "n": "modulus_in_base64url...",
      "e": "AQAB"
    }
  ]
}
```

Figure 5

* *Signature*

The Issuer (https://client.myassistant.example) generates the signature using its private key and the RS256 (RSA Signature with SHA-256) algorithm over the concatenated content: base64url(header) + '.' + base64url(payload).

Final Output as Standard JWT Tripartite String

The resulting JWT is a URL-safe, three-part encoded string in the format:

eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsImtpZCI6Imh0dHBzOi8vY2xpZW50Lm15YW5zd2VyLmV4YW1wbGUvLndlbGwta25vd24vandrLmpzb24ja2V5LTAxIn0. eyJqdGkiOiJwdC0wMDEiLCJpc3MiOiJodHRwczovL2NsaWVudC5teWFuc3dlci5leGFtcGxlIiwic3ViIjoidXNlcl8xMjM0NSIsImlhdCI6MTczMTY2NDUwMCwiZXhwIjoxNzMxNjY4MTAwLCJ0eXBlIjoiVmVyaWZpYWJsZUNyZWRlbnRpYWwiLCJjcmVkZW50aWFsU3ViamVjdCI6eyJ0eXBlIjoiVXNlcklucHV0RXZpZGVuY2UiLCJwcm9tcHQiOiJCdXkgc29tZXRoaW5nIGNoZWFwIG9uIE5vdiAxMSBu .SIGNATURE

* *Verification Process:*

(1) Decode the JWT; (2) Extract the kid from the header; (3) Retrieve the corresponding public key from /.well-known/jwks.json; (4) Validate the cryptographic signature; (5) Check policy conditions such as iss, time window (iat, exp), and device fingerprint.

The Agent Client sends this PAR-JWT to the Authorization Server (AS) via the Pushed Authorization Request (PAR) mechanism, as defined in [RFC9126] (OAuth 2.0 Pushed Authorization Requests).

4. Agent Operation Authorization Token

Upon successful user authorization and authentication, the Authorization Server (AS) issues a *Verifiable Agent Operation Credential* in the form of a JWT token. The purpose of this credential is to serve as a digitally signed and independently verifiable "authorization letter", which enables the Personal Agent to perform authorized operations on behalf of the user. The issuer of the credential is the Authorization Server (AS), and the intended recipient is the Personal Agent (which may be delivered via the client). The credential becomes effective immediately after the user clicks "Allow" or "Consent".

```
{
  "iss": "https://as.online-shop.com",
  "sub": "user_12345@myassistant.example",
  "aud": "personal-agent.myassistant.example",
  "iat": 1731665200,
  "exp": 1732528800,
  "jti": "urn:uuid:aoc-authz-789",

  // ====== Evidence JWT-VC======
  "evidence": {
    "sourcePromptCredential": "eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9...SIGNATURE"
  },

  // ====== Agent Operation Authorization ======
  "agent_operation_authorization": {
    "version": "1.0",
    "id": "urn:uuid:aoc-authz-789",
    "issuer": "https://as.online-shop.com",
    "issuedTo": "user_12345@myassistant.example",
    "issuedFor": {
      "platform": "personal-agent.myassistant.example",
      "client": "mobile-app-v1.myassistant.example",
      "clientInstance": "dfp_abc123"
    },
    "issuanceDate": "2025-11-11T10:08:20Z",
    "validFrom": "2025-11-11T10:10:00Z",
    "expires": "2025-11-16T06:00:00Z",

    "operations": [
      {
        "resources": ["https://api.online-shop.com/api/cart"],
        "actions": ["purchase"]
      }
    ],
    "constraints": {
```

```
      "usage_limit": 1,
      "revocable": true
    },
    "conditions": {
      "language": "cel",
      "expression": "transaction.amount <= 50.0 && time.hour < 6"
    },
    "renderedText": "Purchase items under $50 during the Dec 11 night (00:0006:00)"
  },

  // ====== auditTrail ======
  "auditTrail": {
      "originalPromptText": "Buy something cheap on Nov 11 night",
      "renderedOperationText": "Purchase items under $50 during the Dec 11 night (00:0006
:00)",
      "semanticExpansionLevel": "medium",
      "userAcknowledgeTimestamp": "2025-11-11T10:23:31Z",
      "consentInterfaceVersion": "consent-ui-v2.1"
  },

  // ====== Optional: Reference to Proposal ======
  "references": {
    "relatedProposalId": "urn:uuid:op-proposal-456"
  }
}
```

Figure 6

* *auditTrail* establishes a complete, semantically traceable chain—from the user's original intent to the system's final executed action—in AI Agent scenarios. This mechanism is known as a _Semantic Audit Trail_. The specific purposes and their descriptions are outlined in the following table:

Purpose	Description
1. Intent Provenance	Records what the user originally said (e.g., "Buy something cheap on Nov 11 night") to prevent disputes such as: "I didn't say I wanted to buy anything!"
2. Action Interpretation	Documents how the system interpreted and rendered the input into a concrete operation (e.g., "Purchase under $50 during 00:0006:00"), reflecting the AI's reasoning process.
3. Semantic Transparency	Shows whether semantic expansions or default values were applied (e.g., mapping "cheap" to $50, defining "night" as 00:0006:00).
4. User Confirmation Evidence	Includes timestamps indicating when the user reviewed and confirmed the interpreted action, serving as proof of authorization.
5. Accountability Support	Enables post-hoc analysis in case of erroneous transactions: Was the issue due to ambiguous user input, system misinterpretation, or misleading UI guidance.

Table 1: Purposes and Descriptions of the Semantic Audit Trail

5. Workflow

5.1. High-Level Flow

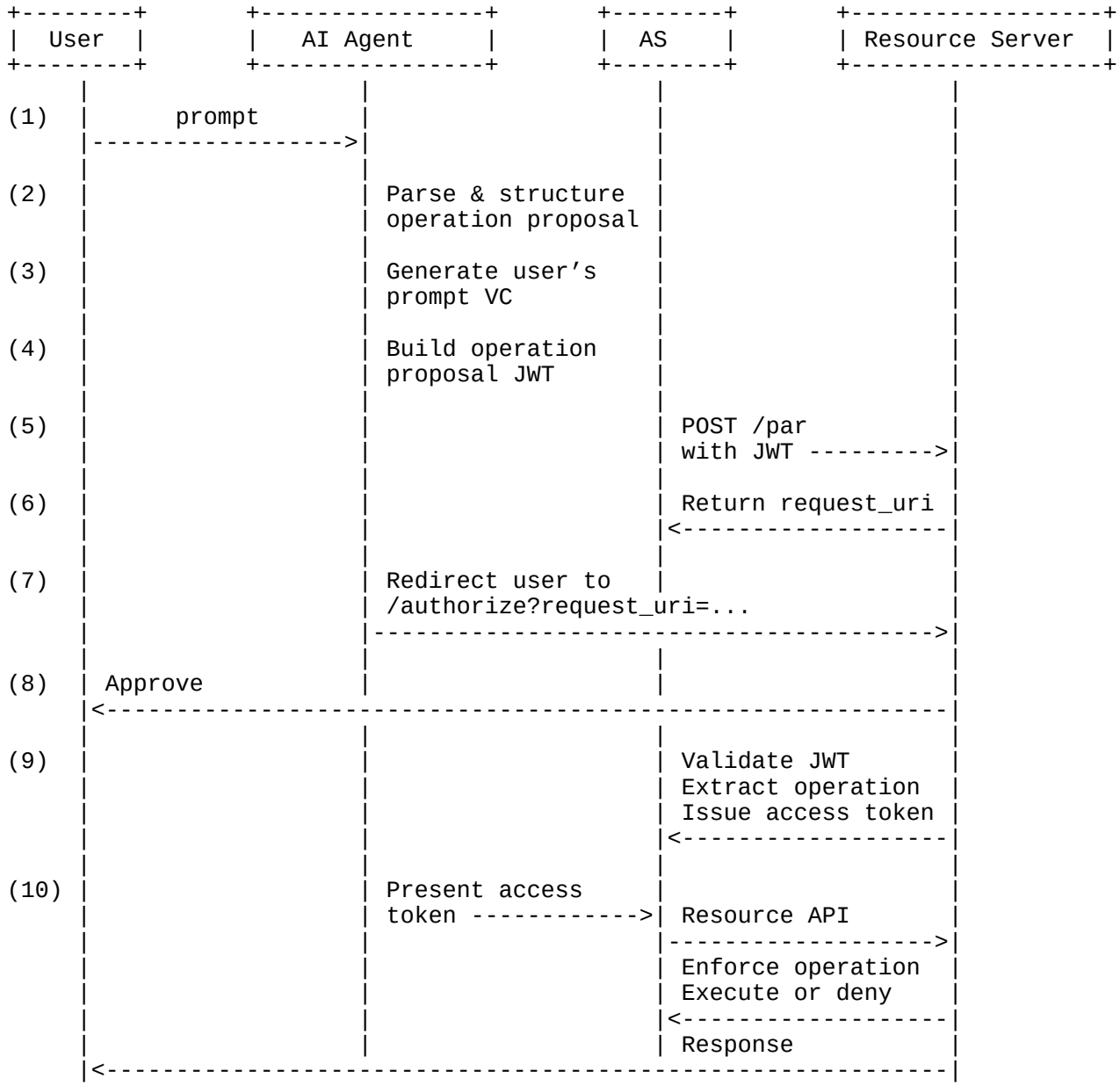

Figure 7

5.2. Detailed Process Flow

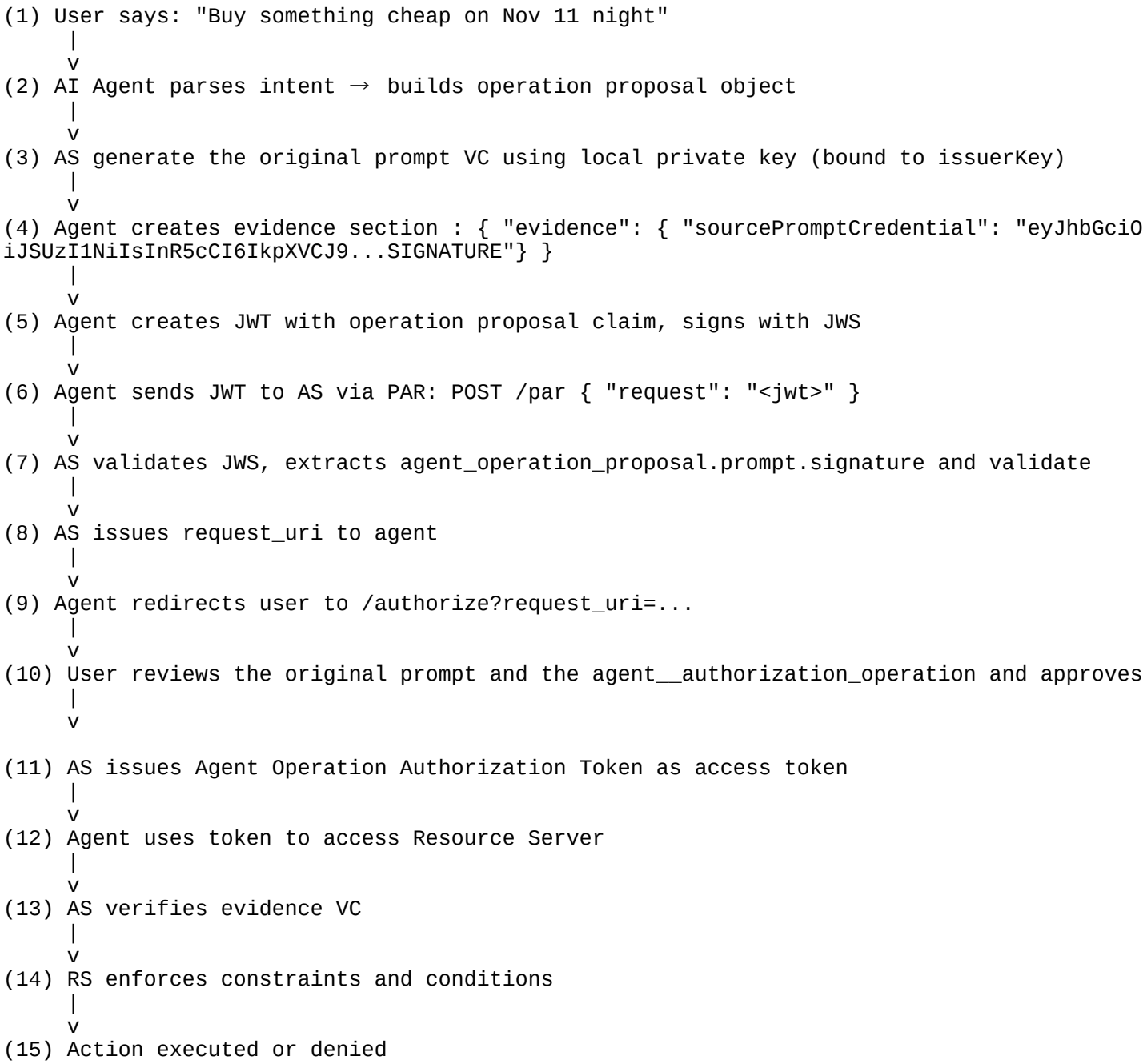
(1) User says: "Buy something cheap on Nov 11 night"
(2) AI Agent parses intent → builds operation proposal object
(3) AS generate the original prompt VC using local private key (bound to issuerKey)
(4) Agent creates evidence section : { "evidence": { "sourcePromptCredential": "eyJhbGciO
iJSUzI1NiIsInR5cCI6IkpXVCJ9...SIGNATURE"} }
(5) Agent creates JWT with operation proposal claim, signs with JWS
(6) Agent sends JWT to AS via PAR: POST /par { "request": "<jwt>" }
(7) AS validates JWS, extracts agent_operation_proposal.prompt.signature and validate
(8) AS issues request_uri to agent
(9) Agent redirects user to /authorize?request_uri=...
(10) User reviews the original prompt and the agent__authorization_operation and approves
(11) AS issues Agent Operation Authorization Token as access token
(12) Agent uses token to access Resource Server
(13) AS verifies evidence VC
(14) RS enforces constraints and conditions
(15) Action executed or denied

Figure 8

6. Security Considerations

The combination of JWS and VC provides dual-layer integrity: JWS protects the token, VC protects the prompt.

Authorization Servers MUST validate the VC proof using the referenced issuerKey and associated public key material before accepting the request.

Public keys referenced by issuerKey MUST be obtained through secure, trusted mechanisms (e.g., pre-registration, PKI).

Expression evaluation (e.g., CEL) MUST occur in sandboxed environments.

The use of PAR prevents leakage of sensitive operation data in URLs.

7. IANA Considerations

7.1. JWT Claim Registration

This document requests IANA to register the following two claims in the "JSON Web Token Claims" registry, following the procedure defined in RFC 8126.

Claim Name: agent_operation_proposal

Claim Description: A structured representation of an operation proposed by an autonomous agent on behalf of a user. It includes intended actions, constraints, conditions, and references to verifiable evidence (e.g., signed user input). Used in delegation flows where user intent is expressed through natural language and converted into machine-executable proposals.

Change Controller: IETF

Specification Document: This document, Section X.Y ("Agent Operation Proposal")

Claim Name: agent_operation_authorization

Claim Description: A structured authorization decision issued by an Authorization Server in response to an operation proposal. It mirrors the structure of the proposal but represents a formally approved scope of execution, potentially with additional policy-enforced constraints. Enables auditable, revocable, and context-aware delegation for AI agents.

Change Controller: IETF

Specification Document: This document, Section X.Z ("Agent Operation Authorization")

Both claims are intended to be used within JWTs carrying structured permissions and operational intent in human-AI collaboration scenarios, particularly in regulated environments requiring traceability, non-repudiation, and alignment with EU AI Act principles such as transparency and accountability.

7.2. JSON Schema Registration (Optional)

Implementers may choose to publish formal JSON Schemas for agent_operation_proposal and agent_operation_authorization. If standardized schemas are developed, they can be submitted to the IANA "JSON Schema Reserved Vocabulary" registry per RFC 9539.

8. References

8.1. 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>.

[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/info/rfc7519>.

[RFC9126] Lodderstedt, T., Campbell, B., Sakimura, N., Tonge, D., and F. Skokan, "OAuth 2.0 Pushed Authorization Requests", RFC 9126, DOI 10.17487/RFC9126, September 2021, <https://www.rfc-editor.org/info/rfc9126>.

Acknowledgments

The author thanks contributors from the IETF community for their valuable feedback on agent authorization semantics.

Authors’ Addresses

Dapeng Liu
Alibaba
Email: max.ldp@alibaba-inc.com

Hongru Zhu
Alibaba
Email: hongru.zhr@alibaba-inc.com