OAuth
Internet-Draft
Intended status: Standards Track
Expires: 3 September 2026

C. Chu
R. Li
H. Wang
T. Li
Huawei Int. Pte Ltd
2 March 2026

OAuth 2.0 Rich Authorization Requests for AS-Attested User Certificates
draft-chu-oauth-as-attested-user-cert-00

Abstract

This document defines an extension to the OAuth 2.0 Rich Authorization Requests (RAR) framework. It introduces a mechanism that allows a Client, such as an autonomous AI Agent, to request an Authorization Server (AS) to include an AS-attested Resource Owner public key certificate within, or bound to, an Access Token.

This mechanism enables the Resource Server (RS) to securely obtain the Resource Owner's trusted public key, which can then be used to verify application-layer delegation evidence (e.g., Verifiable Credentials) signed by the Resource Owner.

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 3 September 2026.

Copyright Notice

Table of Contents

1. 1. Introduction

With the rise of autonomous AI Agents acting on behalf of users, traditional OAuth 2.0 [RFC6749] scopes often fail to provide the fine-grained, user-signed intent verification required for high-stakes operations. A robust solution involves the Resource Owner (RO) issuing a Verifiable Credential (VC) to the Client. To exercise these delegated rights, the Client generates and presents a Verifiable Presentation (VP)-derived from the original VC-to the Resource Server (RS) to prove specific, user-signed delegation boundaries.

However, a trust gap exists: while the RO uses its private key to sign the VC, the RS lacks a standard mechanism to verify if the corresponding public key genuinely belongs to the RO. This specification bridges that gap by leveraging Rich Authorization Requests (RAR) [RFC9396]. It allows a Client to request an AS-attested certificate of the RO's public key. The RS can then extract this certificate from the Access Token to verify the signature of the RO-issued evidence contained within the VP.

2. 2. Conventions and Terminology

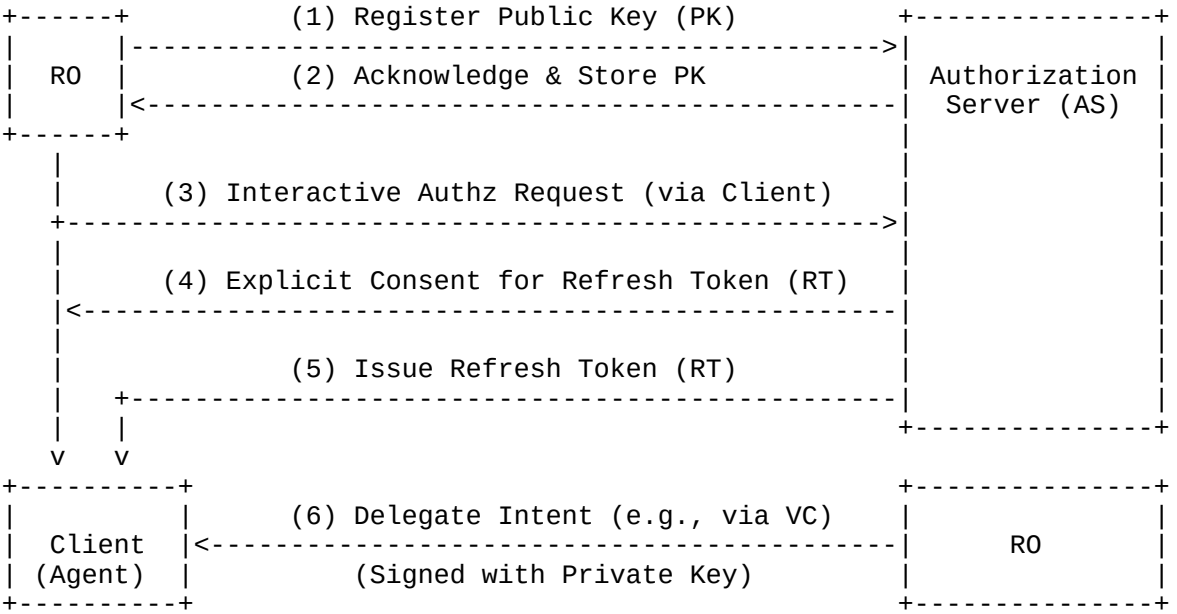
The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 RFC2119 [RFC8174].

3. 3. Protocol Overview

The protocol consists of two phases: Initial Bootstrapping and Autonomous Execution.

3.1. 3.1. Phase 1: Initial Bootstrapping and Registration

The RO registers their public key and authorizes the Client to act on their behalf.



1. *Public Key Registration:* The RO registers their public key (PK) with the AS.

2. *Storage:* The AS stores the PK and binds it to the RO's identity.

3. *Authorization Request:* The Client initiates an interactive OAuth 2.0 flow.

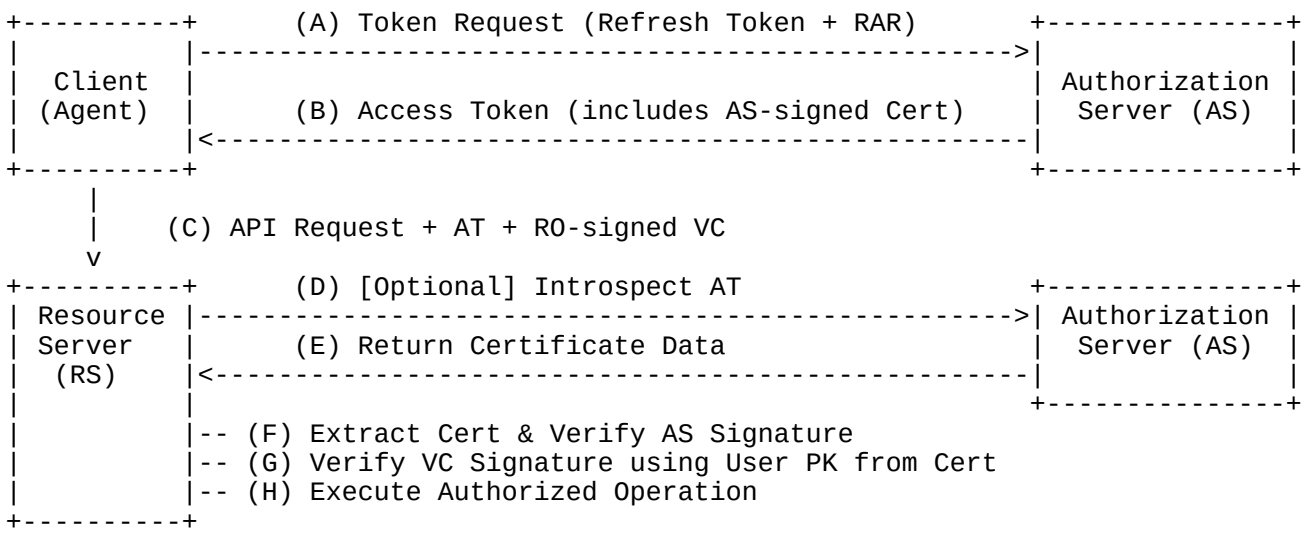
4. *Consent:* The AS authenticates the RO and obtains consent for a Refresh Token (RT).

5. *Token Issuance:* The AS issues an RT to the Client.

6. *Intent Delegation:* The RO provides the Client with signed evidence (e.g., a VC) out-of-band.

3.2. 3.2. Phase 2: Autonomous Execution

The Client uses the Refresh Token to request an Access Token containing the RO's certificate.



* *(A) Token Request:* The Client calls the AS /token endpoint with grant_type=refresh_token and RAR authorization_details.

* *(B) AT Issuance:* The AS validates the RT and issues an Access Token (AT) containing the RO's certificate.

* *(C) Resource Request:* The Client presents the AT and the RO-signed VC to the RS.

* *(D-E) Introspection:* The RS may retrieve certificate data via introspection if the AT is opaque.

* *(F-G) Verification:* The RS verifies the certificate and then uses the RO’s PK to verify the VC.

4. 4. Authorization Details Type

This section defines the new RAR authorization_details type urn:ietf:params:oauth:as-attested-user-cert. This type enables the Client to explicitly request the RO’s public key attestation.

4.1. 4.1. The authorization_details Object

The following fields are defined for this RAR type:

* *type* (REQUIRED): MUST be set to urn:ietf:params:oauth:as-attested-user-cert.

* *cert_format* (OPTIONAL): A string indicating the preferred format for the attestation. Supported values include x509 (for [RFC5280] certificates) and jwk (for [RFC7517] JSON Web Keys). If omitted, the AS MAY return a format based on its local policy.

* *intended_rs* (OPTIONAL): An array of strings containing the identifiers (e.g., URIs) of the Resource Servers. This allows the AS to restrict the audience of the issued certificate.

4.2. 4.2. Example Token Request

‘‘‘http POST /token HTTP/1.1 Host: as.example.com Content-Type: application/x-www-form-urlencoded

grant_type=refresh_token &refresh_token=tGzv3JOkF0XG5Qx2TlKWIA &client_id=ai-agent-client-123 &authorization_details=[{ "type": "urn:ietf:params:oauth:as-attested-user-cert", "cert_format": "x509", "intended_rs": ["https://rs.example.com/api/v1 (https://rs.example.com/api/v1)"] }] ‘‘‘

5. 5. Authorization Server Processing

When the AS receives a token request with the urn:ietf:params:oauth:as-attested-user-cert type, it MUST perform the following processing steps:

1. *Grant Validation:* Validate the refresh_token or other provided grant according to [RFC6749].

2. *RO Identification:* Identify the RO associated with the authenticated session or grant.

3. *Public Key Retrieval:* Retrieve the RO’s registered public key from the AS’s secure storage. If no public key is registered for this RO, the AS MUST return an invalid_request error.

4. *Certificate Generation:* Generate a digital attestation (e.g., an X.509 certificate) that binds the RO’s identity (the sub value) to the retrieved public key. The AS MUST sign this attestation using its own private key.

5. *Issuance:* Include the attestation data in the authorization_details claim of the issued Access Token.

6. 6. Access Token and Introspection

The AS MUST provide the RO’s certificate data to the RS, either by embedding it directly in a structured token (e.g., JWT) or by returning it via an introspection response.

6.1. 6.1. JWT Access Token Payload

For JWT-formatted Access Tokens [RFC9068], the AS MUST include the certificate in the authorization_details array:

```
{
  "iss": "[https://as.example.com](https://as.example.com)",
  "sub": "user_12345",
  "aud": "[https://rs.example.com/api/v1](https://rs.example.com/api/v1)",
  "exp": 1718293600,
  "authorization_details": [
    {
      "type": "urn:ietf:params:oauth:as-attested-user-cert",
      "certificate_data": "MIIDdzCCAl+gAwIBAgIE...<Base64_Encoded_Cert>..."
    }
  ]
}
```

6.2. 6.2. Token Introspection Response

If the Access Token is opaque, the RS MUST use Token Introspection [RFC7662]. The AS response MUST include the same authorization_details object as defined above:

```
{
  "active": true,
  "scope": "read write",
  "authorization_details": [
    {
      "type": "urn:ietf:params:oauth:as-attested-user-cert",
      "certificate_data": "MIIDdzCCAl+gAwIBAgIE..."
    }
  ]
}
```

7. 7. Resource Server Processing

The RS MUST follow these steps to verify the AI Agent’s request:

1. *AT Validation:* Validate the Access Token (signature, expiration, and audience).

2. *Certificate Extraction:* Extract the certificate_data from the authorization_details claim.

3. *AS Signature Verification:* Verify the AS’s signature on the certificate using the AS’s well-known public key. This confirms that the RO’s public key has not been tampered with.

4. *Application Evidence Verification:* Extract the RO’s public key from the verified certificate and use it to verify the signature of the application-layer evidence (e.g., the VP/VC provided by the AI Agent).

5. *Execution:* Proceed with the requested operation only if all cryptographic signatures are valid.

8. 8. Security Considerations

8.1. 8.1. Integrity and Authenticity of the Resource Owner’s Public Key

The AS-attested certificate is the primary defense against key substitution attacks. The RS MUST NOT use any RO public key that is not accompanied by a valid AS signature. Any discrepancy in the AS signature MUST result in the immediate rejection of the request.

8.2. 8.2. Proof-of-Possession (PoP)

To prevent token/credential leakage risks, it is RECOMMENDED that the Client's request to the RS be protected by a Proof-of-Possession mechanism (e.g., DPoP or mTLS), ensuring the AT and the RO-signed evidence are bound to the current Client.

8.3. 8.3. Validation of Delegated Intent Boundaries

The RS MUST strictly parse and validate the claims within the presented VP to ensure the requested action falls within the explicit boundaries signed by the RO.

9. 9. IANA Considerations

9.1. 9.1. OAuth Authorization Details Types Registry

This document requests the registration of the following type in the "OAuth Authorization Details Types" registry:

Type Value	Description	Change Controller	Reference
urn:ietf:params:oauth:as-attested-user-cert	Request an AS-attested RO public key certificate	IETF	[[This Document]]

Table 1

10. 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/rfc/rfc2119>.

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, DOI 10.17487/RFC6749, October 2012, <https://www.rfc-editor.org/rfc/rfc6749>.

[RFC6750] Jones, M. and D. Hardt, "The OAuth 2.0 Authorization Framework: Bearer Token Usage", RFC 6750, DOI 10.17487/RFC6750, October 2012, <https://www.rfc-editor.org/rfc/rfc6750>.

[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/rfc/rfc7519>.

[RFC7662] Richer, J., Ed., "OAuth 2.0 Token Introspection", RFC 7662, DOI 10.17487/RFC7662, October 2015, <https://www.rfc-editor.org/rfc/rfc7662>.

[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/rfc/rfc8174>.

[RFC9396] Lodderstedt, T., Richer, J., and B. Campbell, "OAuth 2.0 Rich Authorization Requests", RFC 9396, DOI 10.17487/RFC9396, May 2023, <https://www.rfc-editor.org/rfc/rfc9396>.

Authors' Addresses

Cheng-Kang Chu
Huawei Int. Pte Ltd
Email: chu.cheng.kang@huawei.com

Ruochen Li
Huawei Int. Pte Ltd
Email: li.ruochen@h-partners.com

Haiguang Wang
Huawei Int. Pte Ltd
Email: wang.haiguang.shieldlab@huawei.com

Tieyan Li
Huawei Int. Pte Ltd
Email: Li.Tieyan@huawei.com