Individual Submission
Internet-Draft
Intended status: Informational
Expires: 18 October 2026

G. Moros
Midas Labs

16 April 2026

Browser Session Establishment Using OAuth 2.0 Token Exchange and Short-Lived Authorization Codes

draft-moros-oauth-browser-session-handoff-00

Abstract

This document specifies a usage profile that composes OAuth 2.0 Token Exchange [RFC8693] with a short-lived, single-use authorization code to establish an authenticated browser session at a Relying Party (RP) on behalf of a user authenticated at an Identity Provider (IdP) operating an independent OAuth 2.0 Security Token Service (STS). The server-to-server leg uses RFC 8693 to convey user identity and authorization context to the RP's STS, which issues an RP-scoped access token. A short-lived opaque code is then used to mediate delivery of session state into the user's browser without ever exposing the access token on the front channel.

The profile is designed to avoid the class of front-channel token leakage (browser history, HTTP Referer header, intermediary access logs) that motivated the deprecation of the OAuth 2.0 implicit grant in [RFC9700]. The profile also defines a minimum claims contract on the RP-issued access token to enable stateless authorization enforcement at the RP without synchronous callbacks to the IdP.

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 18 October 2026.

Table of Contents

1. Introduction

It is increasingly common for an organization (the "IdP-side party") authenticated at an Identity Provider to wish to present its users with a product surface operated by a second organization (the "RP-side party") on an RP-controlled domain, without requiring the user to re-authenticate at the RP. The OAuth 2.0 Token Exchange specification [RFC8693] provides a mechanism by which the IdP-side party's backend can exchange a subject token representing the user for an access token usable at the RP. However, RFC 8693 is, by its own Section 1, scoped to the server-to-server exchange and explicitly leaves usage context out of scope:

> "The security tokens obtained may be used in a number of contexts, the specifics of which are also beyond the scope of this specification."

A naive extension of the exchange into the browser leg -- for example, placing the RP-issued access token in a redirect URL -- reintroduces the same class of front-channel token exposure that motivated deprecation of the OAuth 2.0 implicit grant. The current OAuth 2.0 Security Best Current Practice [RFC9700] makes this clear and recommends against transporting access tokens in the front channel.

This document specifies a profile that composes RFC 8693 with a short-lived, single-use authorization code to close the browser-leg gap. The profile is intentionally narrow: it does not modify RFC 8693, does not require changes to the IdP, and adds a single RP-side endpoint beyond the standard exchange endpoint. It is inspired by the authorization code flow of OAuth 2.0 [RFC6749] Section 4.1, adapted for a post-exchange browser handoff rather than an end-user authorization step.

A secondary contribution of this document is a minimum claims contract on the RP-issued access token, enabling the RP to enforce authorization statelessly without synchronous introspection or out-of-band synchronization with the IdP. This contract is designed to be compatible with existing JWT-based access tokens and does not require new registered claims beyond those already established in [RFC7519] and [RFC8693].

1.1. Design Goals

The profile defined in this document aims to satisfy the following:

1. The RP-issued access token MUST NOT appear in any URL, browser history entry, HTTP Referer header, or intermediary access log at any point in the protocol flow.

2. The browser handoff artifact MUST be single-use and time-bound such that replay beyond a narrow window is not useful to an attacker.

3. The RP MUST be able to enforce authorization decisions using the issued access token alone, without synchronous callbacks to the IdP.

4. The IdP remains the sole source of truth for user identity, tenancy, and permissions.

5. The profile MUST compose with unmodified RFC 8693 token exchange semantics.

1.2. Non-Goals

The following are explicitly out of scope for this document:

* Real-time revocation propagation for active sessions. Section 6.9 discusses accepted staleness and points to out-of-band mechanisms.

* RP-originated authorization semantics that the IdP has no knowledge of. A provisioning-based synchronization (for example, via SCIM) is an appropriate companion mechanism and is left to the deployment.

* Cryptographic token binding mechanisms beyond those provided by TLS and standard JWT validation.

1.3. 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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.4. Terminology

This document uses the terms "access token", "authorization server", "client", "resource server", and "token endpoint" as defined in [RFC6749], and the terms "Claim" and "JWT Claims Set" as defined in [RFC7519]. In addition, the following terms are defined:

IdP (Identity Provider)
: The organization and associated infrastructure at which the end user is already authenticated. The IdP operates the authorization server that issues the subject token presented in the RFC 8693 exchange.

IdP Backend
: A server-side component operated by the IdP-side organization that initiates the token exchange on behalf of an authenticated end user.

RP (Relying Party)
: The organization and associated infrastructure operating the destination product surface.

RP STS
: The Security Token Service operated by the RP. Implements the RFC 8693 token exchange grant type and issues RP-scoped access tokens.

RP Session Endpoint
: An RP-operated HTTP endpoint that accepts a handoff code from the browser and establishes an authenticated session.

Handoff Code
: A short-lived, single-use, opaque string generated by the RP that indirectly references a cached RP-issued access token.

RP Access Token
: The access token issued by the RP STS in response to the RFC 8693 exchange. In this profile, it is a JWT [RFC7519] signed by the RP STS and verifiable via a JWKS [RFC7517] published by the RP STS.

2. Protocol Overview

The profile has two trust legs: a server-to-server exchange (Leg A) and a browser-mediated handoff (Legs B and C). The RP access token is present only in server-side contexts from the moment of issuance through the establishment of the browser session.

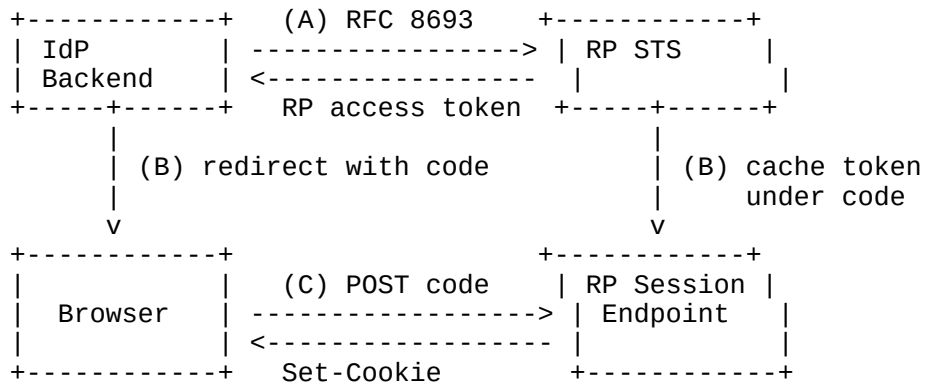

Figure 1: High-Level Flow

Leg A is a standard RFC 8693 exchange: the IdP Backend presents a subject token obtained from the IdP to the RP STS. The RP STS validates the subject token against the IdP's published JWKS, applies policy, and returns an RP access token.

Leg B introduces the handoff code. The RP (either as part of the exchange response or via a separate endpoint; see Section 4.3) generates a handoff code and stores the RP access token in a server-side cache keyed by the code. The IdP Backend then responds to the user's browser with an HTTP redirect whose URL carries only the code.

Leg C is redemption. The browser, now on the RP-controlled domain, POSTs the code to the RP Session Endpoint. The RP atomically retrieves and invalidates the cached access token, establishes a server-side session, and sets a session cookie scoped to the RP domain.

3. Token Exchange (Leg A)

3.1. Trust Setup

The IdP MUST publish a JWKS [RFC7517] at a stable URL. The RP STS MUST retrieve and cache this JWKS and use it to validate subject token signatures. Symmetric keys (MUST NOT) be used for subject token signing in this profile.

The RP STS MUST publish its own JWKS at a stable URL for validation of RP-issued access tokens by downstream RP components.

The IdP Backend MUST authenticate to the RP STS when initiating an exchange. [RFC7523] client authentication using a private key (private_key_jwt) is RECOMMENDED. client_secret_basic MAY be used in non-production environments but SHOULD NOT be used in production deployments.

3.2. Exchange Request

The IdP Backend sends an RFC 8693 token exchange request to the RP STS:

```
POST /oauth2/token HTTP/1.1
Host: sts.rp.example
Content-Type: application/x-www-form-urlencoded
Authorization: <client authentication per Section 3.1>

grant_type=urn:ietf:params:oauth:grant-type:token-exchange
&audience=https://rp.example/
&subject_token=<IdP-issued JWT for the user>
&subject_token_type=urn:ietf:params:oauth:token-type:jwt
&requested_token_type=urn:ietf:params:oauth:token-type:access_token
```

Figure 2: Token Exchange Request

3.3. Exchange Response

On success, the RP STS returns a standard RFC 8693 response per Section 2.2.1 of [RFC8693]:

```
HTTP/1.1 200 OK
Content-Type: application/json
Cache-Control: no-store

{
  "access_token": "<RP-issued JWT>",
  "issued_token_type":
      "urn:ietf:params:oauth:token-type:access_token",
  "token_type": "Bearer",
  "expires_in": 1800
}
```

Figure 3: Token Exchange Response

3.4. Validation Performed by the RP STS

The RP STS MUST, at minimum:

1. Validate the IdP Backend’s client authentication.

2. Validate the signature of the subject_token against the IdP's JWKS.

3. Validate the iss, aud, exp, nbf, and iat claims of the subject token per Section 4 of [RFC7519].

4. Confirm that the subject token was intended for use as a subject in this exchange. A dedicated claim value or scope registered between IdP and RP at onboarding SHOULD be used. Acceptance of arbitrary IdP-issued tokens minted for unrelated purposes MUST NOT be permitted.

5. Apply RP-side policy (rate limits, denylists, tenant eligibility).

6. Project identity and authorization claims onto the RP access token per Section 5.

On any validation failure, the RP STS MUST respond per Section 2.2.2 of [RFC8693] and MUST NOT issue a token.

4. Browser Handoff (Legs B and C)

4.1. Handoff Code Generation and Storage

On issuing an RP access token intended for browser-based session establishment, the RP generates a handoff code and persists the mapping from code to RP access token (and associated metadata) in a short-lived server-side store.

The handoff code MUST have the following properties:

Opacity: No information about the user, tenant, or token is derivable from the code's value.

High entropy: At least 128 bits of cryptographically random material, base64url-encoded. 256 bits RECOMMENDED.

Single-use: The cached mapping MUST be deleted atomically on first successful redemption. Any subsequent redemption attempt MUST fail.

Short-lived: A Time-To-Live (TTL) of 60 seconds is RECOMMENDED. TTL MUST NOT exceed 120 seconds.

Client-bound (advisory): A coarse client fingerprint (for example, a

User-Agent hash) captured at issuance SHOULD be recorded alongside the cached entry. On redemption, mismatch SHOULD be logged and MAY cause rejection. Binding to client IP address is NOT RECOMMENDED: mobile carrier NAT, corporate egress, and VPN transitions can cause legitimate redemptions to originate from a different IP than the redirect.

4.2. Redirect to the RP Domain

Having obtained a handoff code, the IdP Backend responds to the user's browser with an HTTP redirect:

```
HTTP/1.1 302 Found
Location: https://rp.example/session/handoff?code=<handoff_code>
Cache-Control: no-store
```

Figure 4: Redirect to RP

A state parameter MAY also be included for CSRF protection against the initiating IdP page, per standard OAuth 2.0 redirect hygiene.

4.3. Code Issuance Shape

Two implementation shapes are conformant; the interface contract is relevant to the IdP-side implementer:

Combined response: The RFC 8693 response is extended with a handoff code alongside the access token. This minimizes round trips but requires the IdP Backend to handle a non-standard response extension.

Separate issuance endpoint: After the RFC 8693 exchange, the IdP Backend calls a second RP endpoint, passing the RP access token and receiving a handoff code. This is cleaner separation between the standard (RFC 8693) and profile-specific concerns and is the RECOMMENDED shape.

4.4. Code Redemption

The browser, having arrived at the RP handoff page, client-side code on that page MUST immediately POST the code to the RP Session Endpoint:

```
POST /session/redeem HTTP/1.1
Host: rp.example
Content-Type: application/json
Origin: https://rp.example

{"code": "<handoff_code>"}
```

Figure 5: Redemption Request

The request is same-origin. The RP Session Endpoint MUST verify the Origin header.

4.5. Session Establishment

On receiving a redemption request, the RP Session Endpoint MUST:

1. Atomically retrieve and delete the code’s cached mapping.

2. Validate the cached RP access token’s exp.

3. Validate the optional client binding from Section 4.1.

4. Create a server-side session record keyed by a server-generated session identifier and store the access token’s claims as the session’s authorization context.

5. Set the session cookie:

```
HTTP/1.1 200 OK
Content-Type: application/json
Set-Cookie: rp_session=<session_id>;
            Domain=rp.example;
            Path=/;
            HttpOnly;
            Secure;
            SameSite=Lax;
            Max-Age=1800
Cache-Control: no-store

{"redirect": "/app/home"}
```

Figure 6: Redemption Response (Success)

The cookie MUST carry the HttpOnly and Secure attributes. SameSite=Lax is the RECOMMENDED default; SameSite=Strict SHOULD be used where the post-login flow permits.

4.6. Failure Modes

On any redemption failure -- unknown code, already-consumed code, expired code, client binding mismatch, expired cached token -- the RP Session Endpoint MUST return a generic error response. Distinguishing failure reasons in the response body MUST NOT occur, to avoid enabling code-probing oracles. Distinct failure reasons SHOULD be recorded in server logs with correlation identifiers for

operational visibility.

5. Access Token Claims Contract

This section specifies the minimum set of claims the RP-issued access token MUST carry to support stateless authorization enforcement at the RP.

5.1. Required Claims

In addition to the standard JWT claims iss, sub, aud, exp, iat, and jti defined in [RFC7519], the following claims MUST appear:

tenant_id (string): The tenancy context in which the user is acting. The RP MUST scope all authorization decisions and data access to this value for the session lifetime.

perms (array of strings): A flat list of permission strings representing authorization grants within the tenant. The RP MUST enforce access decisions against this set. See Section 5.2.

scope (string): Space-delimited scope values per Section 3.3 of [RFC6749], constraining what the access token itself is usable for.

5.2. Permissions Format

Permissions SHOULD be expressed as colon-delimited resource:action strings, for example reports:read, records:write, admin:users:read.

This shape is preferred over named roles because it allows the RP to enforce directly without maintaining a role-to-permission mapping. Where the IdP’s authorization model is role-based, a snapshot of derived permissions SHOULD be materialized into the perms claim at token mint time; a roles claim MAY additionally be included for informational purposes, but the RP MUST enforce against perms.

5.3. Optional Claims

The following claims MAY appear:

email, name: User profile information for display. Not authorization-relevant.

act: Per Section 4.1 of [RFC8693]. Included if the exchange involves delegation. The RP MUST apply access control decisions considering the actor, per RFC 8693 Section 4.1.

5.4. Token and Session Lifetime

The access token exp SHOULD be between 15 and 60 minutes. Shorter values improve the timeliness of authorization changes at the cost of exchange volume. The RP session cookie lifetime MAY exceed the access token lifetime if the RP implements silent re-exchange; such a mechanism is out of scope for this document.

6. Security Considerations

6.1. Front-Channel Exposure

The primary motivation for this profile is to prevent the RP access token from appearing in any front-channel context. This profile ensures the access token is present only in: the RFC 8693 response over TLS; the RP-side cache keyed by the handoff code; and the RP-side session record. The token does not appear in any URL, browser storage, HTTP Referer header, or intermediary log.

6.2. Handoff Code Exposure

The handoff code appears in a URL and is therefore exposed to browser history, access logs at the RP domain, and (potentially) Referer headers if the handoff page links outward. Because the code is single-use and short-lived, exposure after consumption is harmless. Exposure during the redemption window is the only attack surface.

Mitigations:

* TTL not exceeding 120 seconds compresses the attack window.

* Single-use ensures no persistent exposure.

* TLS everywhere removes network-level observation.

* The optional client binding check detects user-agent substitution.

* The handoff page MUST set Referrer-Policy: no-referrer.

6.3. Subject Token Misuse

The RP STS MUST validate that the presented subject token was intended for use in an exchange to the RP. An IdP-issued token minted for an unrelated purpose MUST NOT be accepted. Acceptance SHOULD be gated on an explicit aud value or dedicated scope agreed between IdP and RP at onboarding.

6.4. Handoff Page Hygiene

The RP handoff page MUST:

* Set Referrer-Policy: no-referrer.
* Set Cache-Control: no-store.
* Contain no third-party scripts, trackers, or external resources.
* Perform the redemption POST immediately on load, without user interaction.
* On failure, redirect to a generic error page without reflecting the code or derived information in the URL.

6.5. Session Cookie Attributes

The session cookie MUST be set with:

* HttpOnly, preventing JavaScript access.
* Secure, preventing transmission over plaintext HTTP.
* SameSite=Lax or SameSite=Strict.
* A Domain attribute scoped to the RP origin only.
* A conservative Max-Age aligned with the session lifetime.

6.6. Transport Security

All HTTP interactions in this profile MUST use TLS 1.2 or higher. Plaintext HTTP MUST be rejected at all endpoints.

6.7. Clock Skew

RP STS, IdP Backend, and IdP clocks SHOULD be synchronized. A small skew tolerance of 30 to 60 seconds SHOULD be applied when validating exp, nbf, and iat. The handoff code TTL SHOULD NOT be inflated to absorb clock skew.

6.8. Rate Limiting

The RP STS SHOULD rate-limit exchange requests per authenticated client. The RP Session Endpoint MUST rate-limit redemption attempts per source IP to defend against code-probing, notwithstanding the impracticality of guessing high-entropy codes.

6.9. Revocation and Staleness

Because the RP enforces authorization from the access token alone, permission changes made at the IdP between token-mint events are not visible to the RP until the next exchange. Deployments that require near-real-time revocation SHOULD adopt one of: OAuth 2.0 Token Introspection [RFC7662]; a revocation signal channel (for example, webhook or event stream); or sufficiently short token lifetimes combined with silent re-exchange. Selection among these is a deployment concern and is out of scope for this profile.

6.10. General OAuth Security Considerations

The security considerations of [RFC6749], [RFC6819], [RFC8693], and [RFC9700] apply in full to this profile.

7. Privacy Considerations

The RP access token and the subject token may both carry privacy-sensitive information about the end user, including identifiers, tenancy, and permissions. Both MUST be transmitted only over TLS. Deployments SHOULD practice data minimization: the perms claim should contain only permissions relevant to the RP’s scope, not the user’s full rights at the IdP.

The handoff code itself carries no personal information and is not a privacy concern beyond the exposure of its existence in URL contexts.

8. IANA Considerations

This document has no IANA actions. It does not define new grant types, token type identifiers, JWT claims, or OAuth parameters. All identifiers used are drawn from existing registrations established by [RFC6749], [RFC7519], and [RFC8693].

9. References

9.1. 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>.

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

[RFC7517] Jones, M., "JSON Web Key (JWK)", RFC 7517, DOI 10.17487/RFC7517, May 2015, <https://www.rfc-editor.org/info/rfc7517>.

[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>.

[RFC7523] Jones, M., Campbell, B., and C. Mortimore, "JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants", RFC 7523, DOI 10.17487/RFC7523, May 2015, <https://www.rfc-editor.org/info/rfc7523>.

[RFC7662] Richer, J., Ed., "OAuth 2.0 Token Introspection", RFC 7662, DOI 10.17487/RFC7662, October 2015, <https://www.rfc-editor.org/info/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/info/rfc8174>.

[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>.

9.2. Informative References

[RFC6819] Lodderstedt, T., Ed., McGloin, M., and P. Hunt, "OAuth 2.0 Threat Model and Security Considerations", RFC 6819, DOI 10.17487/RFC6819, January 2013, <https://www.rfc-editor.org/info/rfc6819>.

[RFC9700] Lodderstedt, T., Bradley, J., Labunets, A., and D. Fett, "Best Current Practice for OAuth 2.0 Security", BCP 240, RFC 9700, DOI 10.17487/RFC9700, January 2025, <https://www.rfc-editor.org/info/rfc9700>.

Acknowledgements

This profile draws directly on the design patterns of [RFC8693] and Section 4.1 of [RFC6749]. Reviewers are thanked for their feedback, which motivated the explicit separation of the server-to-server exchange from the browser handoff.

Author’s Address

Gio Moros
Midas Labs
Canada
Email: Gio@midaslabs.ca