

Web Authorization Protocol
Internet-Draft
Intended status: Standards Track
Expires: 29 January 2026

A. Parecki
Okta
28 July 2025

Global Token Revocation
draft-parecki-oauth-global-token-revocation-05

Abstract

Global Token Revocation enables parties such as a security incident management tool or an external Identity Provider to send a request to an Authorization Server to indicate that it should revoke all of a user's existing tokens and require that the user re-authenticates before issuing new tokens.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at <https://drafts.aaronpk.com/global-token-revocation/draft-parecki-oauth-global-token-revocation.html>. Status information for this document may be found at <https://datatracker.ietf.org/doc/draft-parecki-oauth-global-token-revocation/>.

Discussion of this document takes place on the Web Authorization Protocol Working Group mailing list (<mailto:oauth@ietf.org>), which is archived at <https://mailarchive.ietf.org/arch/browse/oauth/>. Subscribe at <https://www.ietf.org/mailman/listinfo/oauth/>.

Source for this draft and an issue tracker can be found at <https://github.com/aaronpk/global-token-revocation>.

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 29 January 2026.

Copyright Notice

Copyright (c) 2025 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	3
2. Conventions and Definitions	3
2.1. Terminology	3
2.2. Roles	4
3. Token Revocation	4
3.1. Revocation Endpoint	4
3.2. Revocation Request	4
3.3. Revocation Expectations	6
3.4. Revocation Response	6
3.4.1. Successful Response	6
3.4.2. Error Response	6
4. Revocation of Access Tokens	7
5. Authorization Server Metadata	7
6. Security Considerations	8
6.1. Authentication of Revocation Request	8
6.2. Enumeration of User Accounts	8
6.3. Malicious Authorization Server	9
7. IANA Considerations	9
7.1. OAuth Authorization Server Metadata	9
8. References	10
8.1. Normative References	10
8.2. Informative References	10
Appendix A. Relationship to Related Specifications	11
A.1. RFC7009: Token Revocation	11
A.2. OpenID Connect Front-Channel Logout	12

A.3. OpenID Connect Back-Channel Logout	12
A.4. Shared Signals Framework	13
Appendix B. Document History	13
Acknowledgments	14
Author's Address	15

1. Introduction

An OAuth Authorization Server issues tokens in response to a user authorizing a client. A party external to the OAuth Authorization Server may wish to instruct the Authorization Server to revoke all tokens belonging to a particular user, and prevent the server from issuing new tokens for that user until the user re-authenticates.

For example, a security incident management tool may detect anomalous behaviour on a user's account, or if the user logged in through an enterprise Identity Provider, the Identity Provider may want to revoke all of a user's tokens in the event of a security incident or upon the employee's termination.

This specification describes a new API endpoint on an Authorization Server that can accept requests from external parties to revoke all tokens associated with a given user.

2. Conventions and Definitions

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.

2.1. Terminology

This specification uses the terms "Access Token", "Authorization Code", "Authorization Endpoint", "Authorization Server" (AS), "Client", "Client Authentication", "Client Identifier", "Client Secret", "End-User", "Grant Type", "Protected Resource", "Redirection URI", "Refresh Token", "Resource Owner", "Resource Server" (RS) and "Token Endpoint" defined by [RFC6749], and the terms "OpenID Provider" (OP) and "ID Token" defined by [OpenID].

This specification uses the term "Identity Provider" (IdP) to refer to the Authorization Server or OpenID Provider that is used for End-User authentication.

TODO: Replace RFC6749 references with OAuth 2.1

2.2. Roles

In a typical OAuth deployment, the OAuth client obtains tokens from the authorization server when a user logs in and authorizes the client. In many cases, the method by which a user authenticates at the authorization server is through an external identity provider.

For example, a mobile chat application is an OAuth Client, and obtains tokens from its backend server which stores the chat messages. The mobile chat backend plays the OAuth roles of "Resource Server" and "Authorization Server".

In some cases, the user will log in to the Authorization Server using an external (e.g. enterprise) Identity Provider. In that case, when a user logs in to the chat application, the backend server may play the role of an OAuth client (or OpenID or SAML "relying party") to the Identity Provider in a new authorization or authentication flow.

3. Token Revocation

A revocation request is an HTTP POST request containing a subject identifier to the Global Token Revocation endpoint, which starts the process of revoking all tokens for the identified subject.

3.1. Revocation Endpoint

The Global Token Revocation endpoint is a URL at the authorization server which accepts HTTP POST requests with parameters in the HTTP request message body using the application/json format. The Global Token Revocation endpoint URL MUST use the https scheme.

If the authorization server supports OAuth Server Metadata ([RFC8414]), the authorization server SHOULD include the URL of their Global Token Revocation endpoint in their authorization server metadata document using the `global_token_revocation_endpoint` parameter as defined in Section 5.

The authorization server MAY alternatively register the endpoint directly with tools that will use it.

3.2. Revocation Request

The request is a POST request with an application/json body containing a single property `sub_id`, the value of which is a Security Event Token Subject Identifier as defined in "Subject Identifiers for Security Event Tokens" [RFC9493].

In practice, this means the value of `sub_id` is a JSON object with a property `format`, and at least one additional property depending on the value of `format`.

The request **MUST** also be authenticated, the particular authentication method and means by which the authentication is established is out of scope of this specification, but may include OAuth 2.0 Bearer Token [RFC6750] or a client authentication JWT [RFC7523].

The following example requests that all tokens for a user identified by an email address be revoked using the Email Identifier Format as defined in Section 3.2.2 of [RFC9493]:

```
POST /global-token-revocation
Host: example.com
Content-Type: application/json
Authorization: Bearer f5641763544a7b24b08e4f74045
```

```
{
  "sub_id": {
    "format": "email",
    "email": "user@example.com"
  }
}
```

If the user identifier at the authorization server is known by the system making the revocation request, the request can use the "Opaque Identifier" format as defined in Section 3.2.4 of [RFC9493] to provide the user identifier:

```
POST /global-token-revocation
Host: example.com
Content-Type: application/json
Authorization: Bearer f5641763544a7b24b08e4f74045
```

```
{
  "sub_id": {
    "format": "opaque",
    "id": "e193177dfdc52e3dd03f78c"
  }
}
```

If it is expected that the authorization server knows about the user identifier at the IdP, the request can use the "Issuer and Subject Identifier" format as defined in Section 3.2.3 of [RFC9493]:

```
POST /global-token-revocation
Host: example.com
Content-Type: application/json
Authorization: Bearer f5641763544a7b24b08e4f74045
```

```
{
  "sub_id": {
    "format": "iss_sub",
    "iss": "https://issuer.example.com/",
    "sub": "af19c476f1dc4470fa3d0d9a25"
  }
}
```

3.3. Revocation Expectations

Upon receiving a revocation request, authorizing the request, and validating the identified user, the Authorization Server:

- * MUST revoke all active refresh tokens
- * SHOULD invalidate all access tokens, although it is recognized that it might not be technically feasible to invalidate access tokens (see Section 4 below)
- * MUST re-authenticate the user before issuing new access tokens or refresh tokens

3.4. Revocation Response

This specification indicates success and error conditions by using HTTP response codes, and does not define the response body format or content.

3.4.1. Successful Response

To indicate that the request was successful and revocation of the requested set of tokens has begun, the server returns an HTTP 204 response.

3.4.2. Error Response

The following HTTP response codes can be used to indicate various error conditions:

- * ***400 Bad Request***: The request was malformed, e.g. an unrecognized or unsupported type of subject identifier.
- * ***401 Unauthorized***: Authentication provided was invalid.

- * ***403 Forbidden***: Insufficient authorization, e.g. missing scopes.
- * ***404 User Not Found***: The user indicated by the subject identifier was not found.
- * ***422 Unable to Process Request***: Unable to log out the user.

4. Revocation of Access Tokens

OAuth 2.0 allows deployment flexibility with respect to the style of access tokens. The access tokens may be self-contained (e.g. [RFC9068]) so that a resource server needs no further interaction with an authorization server issuing these tokens to perform an authorization decision of the client requesting access to a protected resource. A system design may, however, instead use access tokens that are handles (commonly referred to as "reference tokens") referring to authorization data stored at the authorization server.

While these are not the only options, they illustrate the implications for revocation. In the latter case of reference tokens, the authorization server is able to revoke an access token by removing it from storage. In the former case, without storing tokens, it may be impossible to revoke tokens without taking additional measures. One such measure is to use [I-D.ietf-oauth-status-list] to maintain a distributed and easily-compressed list of token revocation statuses.

For this reason, revocation of access tokens is optional in this specification, since it may pose too significant of a burden for implementers. It is not required to revoke access tokens to be able to return a success code to the caller.

5. Authorization Server Metadata

The following authorization server metadata parameters [RFC8414] are introduced to signal the server's capability and policy with respect to Global Token Revocation.

"global_token_revocation_endpoint": The URL of the authorization server's global token revocation endpoint.

"global_token_revocation_endpoint_auth_methods_supported": OPTIONAL. JSON array containing a list of client authentication methods supported by this introspection endpoint. The valid client authentication method values are those registered in the IANA "OAuth Token Endpoint Authentication Methods" registry [IANA.oauth-parameters] or those registered in the IANA "OAuth Access Token Types" registry [IANA.oauth-parameters]. (These

values are and will remain distinct, due to Section 7.2 of [RFC8414].) If omitted, the set of supported authentication methods MUST be determined by other means.

6. Security Considerations

6.1. Authentication of Revocation Request

While Section 3.2 requires that the revocation request is an authenticated request, the specifics of the authentication are out of scope of this specification.

Since the revocation request ultimately has wide-reaching effects (a user is expected to be logged out of all devices), this presents a new Denial of Service attack vector. As such, the authentication used for this request SHOULD be narrowly scoped to avoid granting unnecessary privileges to the caller.

For example, if using OAuth Bearer Tokens, the token SHOULD be issued with a single scope that enables it to perform only the revocation request, and no other type of token issued should include this scope.

If the authorization server is multi-tenant (supports multiple customers) through different identity providers, each identity provider SHOULD use its own scoped credential that is only authorized to revoke tokens for users within the same tenant.

6.2. Enumeration of User Accounts

Typically, an API that accepts a user identifier and returns different statuses depending on whether the user exists would provide an attack vector allowing enumeration of user accounts. This specification does require a "User Not Found" response, so would normally fall under this category. However, requests to the endpoint defined by this specification are required to be authenticated, so this is not considered a public endpoint.

If the tool making the request is compromised, and the attacker can impersonate the requests from this tool (either by coercing the tool to make the request, or by extracting the credentials), then the attacker would be able to enumerate user accounts. However, since the request is not just testing the presence of a user account, but actually revoking the tokens associated with the user if successful, this would likely be easily visible in any audit logs, as many users' tokens would be revoked in a short period of time.

To mitigate some of the concerns of providing such a powerful API endpoint, the users that a particular client can request revocation for SHOULD be limited, and the authentication of the request SHOULD be used to scope the possible user revocation list to only users authorized to the client as described in Section 6.1.

For example, a multi-tenant identity provider that uses different signing keys for users associated with different tenants, can also use the same signing keys to authenticate revocation requests, such as creating a JWT to use as client authentication as described in [RFC7523]. This enables the authorization server receiving the request to only accept revocation requests for users that are associated with the particular tenant at the identity provider.

6.3. Malicious Authorization Server

From the point of view of an identity provider that supports integrations with multiple downstream applications, there is an opportunity for a downstream application to maliciously set up a Global Token Revocation endpoint to harvest user identifiers and authentication of the revocation requests.

Similarly as described in Section 6.1 above, each integration SHOULD be using separate authentication credentials, and each credential SHOULD be scoped as narrowly as possible, such that a malicious server that receives this authentication cannot replay it anywhere else to perform any actions on other systems.

7. IANA Considerations

7.1. OAuth Authorization Server Metadata

IANA has (TBD) registered the following values in the IANA "OAuth Authorization Server Metadata" registry of [IANA.oauth-parameters] established by [RFC8414].

Metadata Name: global_token_revocation_endpoint

Metadata Description: URL of the authorization server's global token revocation endpoint.

Change Controller: IESG

Specification Document: Section X of [[this specification]]

Metadata Name:
global_token_revocation_endpoint_auth_methods_supported

Metadata Description: OPTIONAL. Indicates the list of client authentication methods supported by this endpoint.

Change Controller: IESG

Specification Document: Section X of [[this specification]]

8. References

8.1. Normative References

- [IANA.oauth-parameters]
IANA, "OAuth Parameters",
<<https://www.iana.org/assignments/oauth-parameters>>.
- [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>>.
- [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>>.
- [RFC8414] Jones, M., Sakimura, N., and J. Bradley, "OAuth 2.0 Authorization Server Metadata", RFC 8414, DOI 10.17487/RFC8414, June 2018, <<https://www.rfc-editor.org/rfc/rfc8414>>.
- [RFC9493] Backman, A., Ed., Scurtescu, M., and P. Jain, "Subject Identifiers for Security Event Tokens", RFC 9493, DOI 10.17487/RFC9493, December 2023, <<https://www.rfc-editor.org/rfc/rfc9493>>.

8.2. Informative References

- [I-D.ietf-oauth-status-list]
Looker, T., Bastian, P., and C. Bormann, "Token Status List (TSL)", Work in Progress, Internet-Draft, draft-ietf-oauth-status-list-12, 7 July 2025, <<https://datatracker.ietf.org/doc/html/draft-ietf-oauth-status-list-12>>.

- [OpenID] Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., and C. Mortimore, "OpenID Connect Core 1.0 incorporating errata set 2", December 2023, <https://openid.net/specs/openid-connect-core-1_0.html>.
- [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>>.
- [RFC7009] Lodderstedt, T., Ed., Dronia, S., and M. Scurtescu, "OAuth 2.0 Token Revocation", RFC 7009, DOI 10.17487/RFC7009, August 2013, <<https://www.rfc-editor.org/rfc/rfc7009>>.
- [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/rfc/rfc7523>>.
- [RFC9068] Bertocci, V., "JSON Web Token (JWT) Profile for OAuth 2.0 Access Tokens", RFC 9068, DOI 10.17487/RFC9068, October 2021, <<https://www.rfc-editor.org/rfc/rfc9068>>.

Appendix A. Relationship to Related Specifications

A.1. RFC7009: Token Revocation

OAuth 2.0 Token Revocation [RFC7009] defines an endpoint for authorization servers that an OAuth client can use to notify the authorization server that a previously-obtained access or refresh token is no longer needed.

The request is made by the OAuth client. The input to the Token Revocation request is the token itself, as well as the client's own authentication credentials.

This differs from the Global Token Revocation endpoint which does not take a token as an input, but instead takes a user identifier as input. It is not called by OAuth clients, but is instead called by an external party such as a security monitoring tool or an identity provider that the user used to authenticate at the authorization server.

A.2. OpenID Connect Front-Channel Logout

OpenID Connect Front-Channel Logout (https://openid.net/specs/openid-connect-frontchannel-1_0.html) provides a mechanism for an OpenID Provider to log users out of Relying Parties by redirecting the user agent.

While the logout request is the same direction as this draft describes, this relies on the redirection of the user agent, so is only applicable when the user is actively interacting with the application in a web browser.

The Global Token Revocation request works regardless of whether the user is actively using the application, and is also applicable to non-web based applications.

A.3. OpenID Connect Back-Channel Logout

OpenID Connect Back-Channel Logout (https://openid.net/specs/openid-connect-backchannel-1_0.html) provides a mechanism for an OpenID Provider to log users out of a Relying Party by making a back-channel POST request containing the user identifier of the user to log out.

This is the most similar existing logout specification to Global Token Revocation. However, there are still a few key differences that make it insufficient for the use cases enabled by Global Token Revocation.

OpenID Connect Back-Channel Logout requires Relying Parties to clear state of any sessions for the user, but doesn't mention anything about access tokens. It also says that refresh tokens issued with the `offline_access` scope "SHOULD NOT be revoked". This is a concretely different outcome than is described by Global Token Revocation, which requires the revocation of all refresh tokens for the user regardless of whether the refresh token was issued with the `offline_access` scope.

OpenID Connect Back-Channel Logout also assumes that the Relying Party implements OpenID Connect, which creates implementation challenges to use it when the Relying Party actually integrates with the identity provider using other specifications such as SAML.

Additionally, OpenID Connect Back-Channel Logout identifies the user using the sub claim of an ID token. This limits the applicability, since there is no mechanism to identify the user by email address or other identifier that might be known between the identity provider and authorization server. Global Token Revocation instead relies on Security Event Token Subject Identifiers ([RFC9493]) which provide multiple options for identifying the user.

Global Token Revocation works regardless of the protocol that the user uses to authenticate, so works equally well with OpenID Connect and SAML integrations.

A.4. Shared Signals Framework

The Shared Signals Framework at the OpenID Foundation provides two specifications that have functionality related to session and token revocation.

Continuous Access Evaluation Profile (CAEP) (https://openid.net/specs/openid-caep-specification-1_0.html) defines several event types that can be sent between cooperating parties. In particular, the "Session Revoked" event can be sent from an identity provider to an authorization server when the user's session at the identity provider was revoked. The main difference between this and the Global Token Revocation request is that the CAEP event is a signal that may or may not be acted upon by the receiver, whereas the Global Token Revocation request is a command that has a defined list of expected outcomes.

Risk Incident Sharing and Coordination (RISC) (https://openid.net/specs/openid-risc-profile-specification-1_0.html) defines events that have somewhat stronger defined meanings compared to CAEP. In particular, the "Account Disabled" event has clear meaning and strongly implies that a receiver should also disable the specified account. However, RISC also has a mechanism for a user to opt out of sending events for their account, so it does not provide the same level of assurance as a Global Token Revocation request.

Lastly, it is more complex to set up a receiver for CAEP and RISC events compared to a receiver for the Global Token Revocation request, so if the receiver is only interested in supporting the revocation use cases, it is much simpler to support the single POST request described in this draft.

Appendix B. Document History

((To be removed from the final specification))

-05

- * Editorial clarifications
- * Added specific references to subject identifier formats

-04

- * Edits for clarity
- * Fixed prose description renaming subject to sub_id

-03

- * Renamed property from subject to sub_id for consistency with JWT claim name defined in RFC9493
- * Added reference to draft-ietf-oauth-status-list
- * Added additional security considerations for authentication of the revocation request and malicious authorization servers

-02

- * Added security consideration around enumeration of user accounts
- * Added an appendix describing the differences between this and related logout specifications

-01

- * Clarified revocation expectations
- * Better definition of endpoint
- * Added section defining endpoint in Authorization Server Metadata

-00

- * Initial Draft

Acknowledgments

The authors would like to thank the following people for their contributions and reviews of this specification: Apoorva Deshpande, George Fletcher, Karl McGuinness, Mike Jones.

Author's Address

Aaron Parecki

Okta

Email: aaron@parecki.com

URI: <https://aaronparecki.com>