International Standard

ISO/IEC 26137

Information technology — OpenID connect — OpenID connect back-channel logout 1.0 incorporating errata set 1

First edition
2024-10

Reference number
ISO/IEC 26137:2024(en)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

ISO and IEC draw attention to the possibility that the implementation of this document may involve the use of (a) patent(s). ISO and IEC take no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, ISO and IEC had not received notice of (a) patent(s) which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at www.iso.org/patents and https://patents.iec.ch. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by the OpenID Foundation (OIDF) (as OpenID Connect Back-Channel Logout 1.0 incorporating errata set 1) and drafted in accordance with its editorial rules. It was adopted, under the JTC 1 PAS procedure, by Joint Technical Committee ISO/IEC JTC 1, *Information technology*.

Any feedback or questions on this document should be directed to the user’s national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

Abstract

OpenID Connect 1.0 is a simple identity layer on top of the OAuth 2.0 protocol. It enables Clients to verify the identity of the End-User based on the authentication performed by an Authorization Server, as well as to obtain basic profile information about the End-User in an interoperable and REST-like manner.

This specification defines a logout mechanism that uses direct back-channel communication between the OP and RPs being logged out; this differs from front-channel logout mechanisms, which communicate logout requests from the OP to RPs via the User Agent.

Table of Contents

Information technology — OpenID Connect — OpenID Connect Back-Channel Logout 1.0 incorporating errata set 1

1. Introduction

TOC

OpenID Connect 1.0 is a simple identity layer on top of the OAuth 2.0 [RFC6749] protocol. It enables Clients to verify the identity of the End-User based on the authentication performed by an Authorization Server, as well as to obtain basic profile information about the End-User in an interoperable and REST-like manner.

This specification defines a logout mechanism that uses direct back-channel communication between the OP and RPs being logged out; this differs from front-channel logout mechanisms, which communicate logout requests from the OP to RPs via the User Agent.

An upside of back-channel communication is that it can be more reliable than communication through the User Agent, since in the front-channel, the RP's browser session must be active for the communication to succeed. (If the RP's browser tab was subsequently used to navigate to an unrelated page, the RP session will be active unless the user uses the back button to return to it.) Both the OpenID Connect Session Management 1.0 [OpenID.Session] and OpenID Connect Front-Channel Logout 1.0 [OpenID.FrontChannel] specifications use front-channel communication, which communicate logout requests from the OP to RPs via the User Agent.

A downside of back-channel communication is that the session state maintained between the OP and RP over the front-channel, such as cookies and HTML5 local storage, are not available when using back-channel communication. As a result, all needed state must be explicitly communicated between the parties. Furthermore, RPs must implement an application-specific method of terminating RP sessions with the OP upon receiving back-channel logout requests; this can be more complicated than simply clearing cookies and HTML5 local storage state, which is often all that has to happen to implement logout in response to front-channel logout requests.

Another significant limitation of back-channel logout is that the RP's back-channel logout URI must be reachable from all the OPs used. This means, for instance, that the RP cannot be behind a firewall or NAT when used with public OPs.

The OpenID Connect RP-Initiated Logout 1.0 [OpenID.RPInitiated] specification complements these specifications by defining a mechanism for a Relying Party to request that an OpenID Provider log out the End-User.

This specification can be used separately from or in combination with OpenID Connect RP-Initiated Logout 1.0, OpenID Connect Session Management 1.0, and/or OpenID Connect Front-Channel Logout 1.0.

The previous version of this specification is:

- OpenID Connect Back-Channel Logout 1.0 (final) [OpenID.BackChannel.Final]

1.1. Requirements Notation and Conventions

TOC

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].

In the .txt version of this specification, values are quoted to indicate that they are to be taken literally. When using these values in protocol messages, the quotes MUST NOT be used as part of the value. In the HTML version of this specification, values to be taken literally are indicated by the use of `this fixed-width font`.

1.2. Terminology

TOC

This specification uses the terms "Authorization Server", "Client", and "Client Identifier" defined by OAuth 2.0 [RFC6749], the term "User Agent" defined by RFC 7230 [RFC7230], the terms "Session" and "Session ID" defined by OpenID Connect Front-Channel Logout 1.0 [OpenID.FrontChannel] and the terms defined by OpenID Connect Core 1.0 [OpenID.Core] and JSON Web Token (JWT) [JWT].

This specification also defines the following term:

Logout Token

JSON Web Token (JWT) [JWT] similar to an ID Token that contains Claims about the logout action being requested.

2. Back-Channel Logout

TOC

2.1. Indicating OP Support for Back-Channel Logout

TOC

If the OpenID Provider supports OpenID Connect Discovery 1.0 [OpenID.Discovery], it uses this metadata value to advertise its support for back-channel logout:

backchannel_logout_supported

OPTIONAL. Boolean value specifying whether the OP supports back-channel logout, with `true` indicating support. If omitted, the default value is `false`.

It SHOULD also register this related metadata value:

backchannel_logout_session_supported

OPTIONAL. Boolean value specifying whether the OP can pass a `sid` (session ID) Claim in the Logout Token to identify the RP session with the OP. If supported, the `sid` Claim is also included in ID Tokens issued by the OP. If omitted, the default value is `false`.

The `sid` (session ID) Claim used in ID Tokens and as a Logout Token parameter has the following definition (which is identical to the corresponding definition in OpenID Connect Front-Channel Logout 1.0 [OpenID.FrontChannel]):

sid

OPTIONAL. Session ID - String identifier for a Session. This represents a Session of a User Agent or device for a logged-in End-User at an RP. Different `sid` values are used to identify distinct sessions at an OP. The `sid` value need only be unique in the context of a particular issuer. Its contents are opaque to the RP. Its syntax is the same as an OAuth 2.0 Client Identifier.

2.2. Indicating RP Support for Back-Channel Logout

TOC

Relying Parties supporting back-channel-based logout register a back-channel logout URI with the OP as part of their client registration.

The back-channel logout URI MUST be an absolute URI as defined by Section 4.3 of [RFC3986]. The back-channel logout URI MAY include an `application/x-www-form-urlencoded` formatted query component, per Section 3.4 of [RFC3986], which MUST be retained when adding additional query parameters. The back-channel logout URI MUST NOT include a fragment component.

If the RP supports OpenID Connect Dynamic Client Registration 1.0 [OpenID.Registration], it uses this metadata value to register the back-channel logout URI:

backchannel_logout_uri

> OPTIONAL. RP URL that will cause the RP to log itself out when sent a Logout Token by the OP. This URL SHOULD use the `https` scheme and MAY contain port, path, and query parameter components; however, it MAY use the `http` scheme, provided that the Client Type is `confidential`, as defined in Section 2.1 of OAuth 2.0 [RFC6749], and provided the OP allows the use of `http` RP URIs.

It SHOULD also register this related metadata value:

backchannel_logout_session_required

> OPTIONAL. Boolean value specifying whether the RP requires that a `sid` (session ID) Claim be included in the Logout Token to identify the RP session with the OP when the `backchannel_logout_uri` is used. If omitted, the default value is `false`.

2.3. Remembering Logged-In RPs

TOC

OPs supporting back-channel logout need to keep track of the set of logged-in RPs so that they know what RPs to contact at their back-channel logout URIs to cause them to log out. Some OPs track this state using a "visited sites" cookie. OPs are encouraged to send logout requests to them in parallel.

TOC

2.4. Logout Token

OPs send a JWT similar to an ID Token to RPs called a Logout Token to request that they log out. ID Tokens are defined in Section 2 of [OpenID.Core].

The following Claims are used within the Logout Token:

iss

> REQUIRED. Issuer Identifier, as specified in Section 2 of [OpenID.Core].

sub

> OPTIONAL. Subject Identifier, as specified in Section 2 of [OpenID.Core].

aud

> REQUIRED. Audience(s), as specified in Section 2 of [OpenID.Core].

iat

> REQUIRED. Issued at time, as specified in Section 2 of [OpenID.Core].

exp

> REQUIRED. Expiration time, as specified in Section 2 of [OpenID.Core].

jti

> REQUIRED. Unique identifier for the token, as specified in Section 9 of [OpenID.Core].

events

> REQUIRED. Claim whose value is a JSON object containing the member name `http://schemas.openid.net/event/backchannel-logout`. This declares that the JWT is a Logout Token. The

corresponding member value MUST be a JSON object and SHOULD be the empty JSON object `{}`.

sid

OPTIONAL. Session ID - String identifier for a Session. This represents a Session of a User Agent or device for a logged-in End-User at an RP. Different `sid` values are used to identify distinct sessions at an OP. The `sid` value need only be unique in the context of a particular issuer. Its contents are opaque to the RP. Its syntax is the same as an OAuth 2.0 Client Identifier.

A Logout Token MUST contain either a `sub` or a `sid` Claim, and MAY contain both. If a `sid` Claim is not present, the intent is that all sessions at the RP for the End-User identified by the `iss` and `sub` Claims be logged out.

The following Claim MUST NOT be used within the Logout Token:

nonce

PROHIBITED. A `nonce` Claim MUST NOT be present. Its use is prohibited to make a Logout Token syntactically invalid if used in a forged Authentication Response in place of an ID Token.

Logout Tokens MAY contain other Claims. Any Claims used that are not understood MUST be ignored.

A Logout Token MUST be signed and MAY also be encrypted. The same keys are used to sign and encrypt Logout Tokens as are used for ID Tokens. If the Logout Token is encrypted, it SHOULD replicate the `iss` (issuer) claim in the JWT Header Parameters, as specified in Section 5.3 of [JWT].

It is RECOMMENDED that Logout Tokens be explicitly typed. This is accomplished by including a `typ` (type) Header Parameter with a value of `logout+jwt` in the Logout Token. See Section 4.1 for a discussion of the security and interoperability considerations of using explicit typing.

NOTE: The Logout Token is compatible with the Security Event Token (SET) [RFC8417] specification, but uses a more specific `typ` (type) value.

A non-normative example JWT Claims Set for a Logout Token follows:

```
{
 "iss": "https://server.example.com",
 "sub": "248289761001",
   "aud": "s6BhdRkqt3",
   "iat": 1471566154,
   "exp": 1471569754,
   "jti": "bWJq",
   "sid": "08a5019c-17e1-4977-8f42-65a12843ea02",
   "events": {
     "http://schemas.openid.net/event/backchannel-
logout": {}
     }
  }
```

2.5. Back-Channel Logout Request

TOC

The OP uses an HTTP POST to the registered back-channel logout URI to trigger the logout actions by the RP. The POST body uses the `application/x-www-form-urlencoded` encoding and must include a `logout_token` parameter containing a Logout Token from the OP for the RP identifying the End-User to be logged out.

The POST body MAY contain other values in addition to `logout_token`. Values that are not understood by the implementation MUST be ignored.

The following is a non-normative example of such a logout request (with most of the Logout Token contents omitted for brevity):

```
  POST /backchannel_logout HTTP/1.1
  Host: rp.example.org
  Content-Type: application/x-www-form-urlencoded

  logout_token=eyJhbGci ... .eyJpc3Mi ... .T3BlbklE ...
```

The OP should not retransmit a Back-Channel Logout Request unless the OP suspects that previous transmissions may have failed due to potentially recoverable errors (such as network outage or temporary service interruption at either the OP or RP). In this case, the OP SHOULD delay retransmission for an appropriate amount of time to avoid overwhelming the RP. In all other cases, the OP SHOULD NOT retransmit a Back-Channel Logout Request. (Note that this is the same retransmission logic as specified in Push-Based Security Event Token (SET) Delivery Using HTTP [RFC8935].)

TOC

2.6. Logout Token Validation

Upon receiving a logout request at the back-channel logout URI, the RP MUST validate the Logout Token as follows:

1. If the Logout Token is encrypted, decrypt it using the keys and algorithms that the Client specified during Registration that the OP was to use to encrypt ID Tokens. If ID Token encryption was negotiated with the OP at Registration time and the Logout Token is not encrypted, the RP SHOULD reject it.

2. Validate the Logout Token signature in the same way that an ID Token signature is validated, with the following refinements.

3. Validate the `alg` (algorithm) Header Parameter in the same way it is validated for ID Tokens. Like ID Tokens, selection of the algorithm used is governed by the `id_token_signing_alg_values_supported` Discovery parameter and the `id_token_signed_response_alg` Registration parameter when they are used; otherwise, the value SHOULD be the default of `RS256`. Additionally, an `alg` with the value `none` MUST NOT be used for Logout Tokens.

4. Validate the `iss`, `aud`, `iat`, and `exp` Claims in the same way they are validated in ID Tokens.

5. Verify that the Logout Token contains a `sub` Claim, a `sid` Claim, or both.

6. Verify that the Logout Token contains an `events` Claim whose value is JSON object containing the member name `http://schemas.openid.net/event/backchannel-logout`.

7. Verify that the Logout Token does not contain a `nonce` Claim.

8. Optionally verify that another Logout Token with the same `jti` value has not been recently received.

9. Optionally verify that the `iss` Logout Token Claim matches the `iss` Claim in an ID Token issued for the current session or a recent session of this RP with the OP.

10. Optionally verify that any `sub` Logout Token Claim matches the `sub` Claim in an ID Token issued for the current session or a recent session of this RP with the OP.
11. Optionally verify that any `sid` Logout Token Claim matches the `sid` Claim in an ID Token issued for the current session or a recent session of this RP with the OP.

If any of the validation steps fails, reject the Logout Token and return an HTTP 400 Bad Request error. Otherwise, proceed to perform the logout actions.

2.7. Back-Channel Logout Actions

TOC

After receiving a valid Logout Token from the OpenID Provider, the RP locates the session(s) identified by the `iss` and `sub` Claims and/or the `sid` Claim. The RP then clears any state associated with the identified session(s). The mechanism by which the RP achieves this is implementation specific. If the identified End-User is already logged out at the RP when the logout request is received, the logout is considered to have succeeded.

In the case that the RP is also an OP serving as an identity provider to downstream logged-in sessions, it is desirable for the logout request to the RP to likewise trigger downstream logout requests. This is achieved by having the RP/OP send logout requests to its downstream RPs as part of its logout actions.

Refresh tokens issued without the `offline_access` property to a session being logged out SHOULD be revoked. Refresh tokens issued with the `offline_access` property normally SHOULD NOT be revoked.

2.8. Back-Channel Logout Response

TOC

If the logout succeeded, the RP MUST respond with HTTP 200 OK. However, note that some Web frameworks will substitute an HTTP 204 No Content response for an HTTP 200 OK when the HTTP body is empty. Therefore, OPs should be prepared to also process an HTTP 204 No Content response as a successful response.

If the logout request was invalid or the logout failed, the RP MUST respond with HTTP 400 Bad Request. The response MAY include an HTTP body consisting of a JSON object with `error` and `error_description` parameters conveying the nature of the error that occurred, which can assist with debugging. These error response parameters are used as specified in Section 5.2 of OAuth 2.0 [RFC6749]. Like in OAuth 2.0, the parameters are included in the entity-body of the HTTP response using the `application/json` media type. Also like in OAuth 2.0, the `error` parameter is REQUIRED when a response body is present and the `error_description` parameter is OPTIONAL. An `error` value of `invalid_request` MAY be used to indicate that there was a problem with the syntax of the logout request. Note that the information conveyed in the response body is intended to help debug deployments; it is not intended that implementations use different `error` values to trigger different runtime behaviors.

The RP's response SHOULD include the `Cache-Control` HTTP response header field with a `no-store` value, keeping the response from being cached to prevent cached responses from interfering with future logout requests. An example of this is:

```
  Cache-Control: no-store
```

3. Implementation Considerations

TOC

This specification defines features used by both Relying Parties and OpenID Providers that choose to implement Back-Channel Logout. All of these Relying Parties and OpenID Providers MUST implement the features that are listed in this specification as being "REQUIRED" or are described with a "MUST". No other implementation considerations for implementations of Back-Channel Logout are defined by this specification.

4. Security Considerations

TOC

The signed Logout Token is required in the logout request to prevent denial of service attacks by enabling the RP to verify that the logout request is coming from a legitimate party.

The kinds of Relying Parties that can be logged out by different implementations will vary. Implementations should make it clear, for instance, whether they are capable of logging out native applications or only Web RPs.

OPs are encouraged to use short expiration times in Logout Tokens, preferably at most two minutes in the future, to prevent captured Logout Tokens from being replayable.

4.1. Cross-JWT Confusion

TOC

As described in Section 2.8 of [RFC8725], attackers may attempt to use a JWT issued for one purpose in a context that it was not intended for. The mitigations described for these attacks can be applied to Logout Tokens.

One way that an attacker might attempt to repurpose a Logout Token is to try to use it as an ID Token. As described in Section 2.4, inclusion of a `nonce` Claim in a Logout Token is prohibited to prevent its misuse as an ID Token.

Another way to prevent cross-JWT confusion is to use explicit typing, as described in Section 3.11 of [RFC8725]. One would explicitly type a Logout Token by including a `typ` (type) Header Parameter with a value of `logout+jwt` (which is registered in Section 5.3.1. Note that the `application/` portion of the `application/logout+jwt` media type name is omitted when used as a `typ` Header Parameter value, as described in the `typ` definition in Section 4.1.9 [JWS]. Including an explicit type in issued Logout Tokens is a best practice. Note however, that requiring explicitly typed Logout Tokens will break most existing deployments, as existing OPs and RPs are already commonly using untyped Logout Tokens. However, requiring explicit typing would be a good idea for new deployment profiles where compatibility with existing deployments is not a consideration.

5. IANA Considerations

TOC

5.1. OAuth Dynamic Client Registration Metadata Registration

TOC

This specification registers the following client metadata definitions in the IANA "OAuth Dynamic Client Registration Metadata" registry [IANA.OAuth.Parameters] established by [RFC7591]:

5.1.1. Registry Contents

TOC

- Client Metadata Name: `backchannel_logout_uri`
- Client Metadata Description: RP URL that will cause the RP to log itself out when sent a Logout Token by the OP
- Change Controller: OpenID Foundation Artifact Binding Working Group - openid-specs-ab@lists.openid.net
- Specification Document(s): Section 2.2 of this specification
- Client Metadata Name: `backchannel_logout_session_required`
- Client Metadata Description: Boolean value specifying whether the RP requires that a `sid` (session ID) Claim be included in the Logout Token to identify the RP session with the OP when the `backchannel_logout_uri` is used
- Change Controller: OpenID Foundation Artifact Binding Working Group - openid-specs-ab@lists.openid.net
- Specification Document(s): Section 2.2 of this specification

5.2. OAuth Authorization Server Metadata Registry

TOC

This specification registers the following metadata names in the IANA "OAuth Authorization Server Metadata" registry [IANA.OAuth.Parameters] established by [RFC8414].

5.2.1. Registry Contents

TOC

- Metadata Name: `backchannel_logout_supported`
- Metadata Description: Boolean value specifying whether the OP supports back-channel logout, with `true` indicating support
- Change Controller: OpenID Foundation Artifact Binding Working Group - openid-specs-ab@lists.openid.net
- Specification Document(s): Section 2.1 of this specification
- Metadata Name: `backchannel_logout_session_supported`
- Metadata Description: Boolean value specifying whether the OP can pass a `sid` (session ID) Claim in the Logout Token to identify the RP session with the OP
- Change Controller: OpenID Foundation Artifact Binding Working Group - openid-specs-ab@lists.openid.net
- Specification Document(s): Section 2.1 of this specification