International Standard

ISO/IEC 26135

Information technology — OpenID connect — OpenID connect session management 1.0

First edition
2024-10

Reference number
ISO/IEC 26135:2024(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 Session Management 1.0) 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

TOC

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 document describes how to manage sessions for OpenID Connect, including when to log out the End-User.

Table of Contents

Information technology — OpenID Connect — OpenID Connect Session Management 1.0

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 complements the OpenID Connect Core 1.0 [OpenID.Core] specification by defining how to monitor the End-User's login status at the OpenID Provider on an ongoing basis so that the Relying Party can log out an End-User who has logged out of the OpenID Provider.

Both this specification and the OpenID Connect Front-Channel Logout 1.0 [OpenID.FrontChannel] specification use front-channel communication, which communicate logout requests from the OP to RPs via the User Agent. In contrast, the OpenID Connect Back-Channel Logout 1.0 [OpenID.BackChannel] specification uses direct back-channel communication between the OP and RPs being logged out. 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 these other three specifications.

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 document, 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 document, 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 Endpoint", "Authorization Server", "Client", and "Client Identifier" defined by OAuth 2.0 [RFC6749], the term "User Agent" defined by RFC 7230 [RFC7230], and the terms defined by OpenID Connect Core 1.0 [OpenID.Core].

This specification also defines the following term:

Session

> Continuous period of time during which an End-User accesses a Relying Party relying on the Authentication of the End-User performed by the OpenID Provider.

IMPORTANT NOTE TO READERS: The terminology definitions in this section are a normative portion of this specification, imposing requirements upon implementations. All the capitalized words in the text of this specification, such as "Session", reference these defined terms. Whenever the reader encounters them, their definitions found in this section must be followed.

2. Creating and Updating Sessions

TOC

In OpenID Connect, the session at the RP typically starts when the RP validates the End-User's ID Token. Refer to the OpenID Connect Core 1.0 [OpenID.Core] specification to find out how to obtain an ID Token and validate it. When the OP supports session management, it MUST also return the Session State as an additional `session_state` parameter in the Authentication Response and SHOULD also return the Session State as an additional `session_state` parameter in the Authentication Error Response. The OpenID Connect Authentication Response is specified in Section 3.1.2.5 of OpenID Connect Core 1.0. The OpenID Connect Authentication Error Response is specified in Section 3.1.2.6 of OpenID Connect Core 1.0.

This parameter is:

session_state

Session State. JSON [RFC7159] string that represents the End-User's login state at the OP. It MUST NOT contain the space (" ") character. This value is opaque to the RP. This is REQUIRED if session management is supported.

The Session State value is initially calculated on the server. The same Session State value is also recalculated by the OP iframe in the User Agent. The generation of suitable Session State values is specified in Section 3.2, and is based on a salted cryptographic hash of Client ID, origin URL, and OP User Agent state. For the origin URL, the server can use the origin URL of the Authentication Response, following the algorithm specified in Section 4 of RFC 6454 [RFC6454].

3. Session Status Change Notification

TOC

It is highly desirable to be able to determine the login status of the End-User at the OP. To do so, it is possible to repeat the Authentication Request with `prompt=none`. However, this causes network traffic and this is problematic on the mobile devices that are becoming increasingly popular. Therefore, once the session is established with the Authentication Request and Response, it is desirable to be able to check the login status at the OP without causing network traffic by polling a hidden OP iframe from an RP iframe with an origin restricted postMessage as follows.

3.1. RP iframe

TOC

The RP loads an invisible iframe from itself. This iframe MUST know:

- the ID of the OP iframe, as described in Section 3.2, so that it can postMessage to the OP iframe, and
- the origin URL of the OP iframe, so that it can ensure messages are dispatched to and only processed when originating from the OP.

The RP iframe polls the OP iframe with postMessage at an interval suitable for the RP application. With each postMessage, it sends the session state defined in Section 3.2. The RP iframe MUST enforce that it only processes messages from the origin of the OP frame. It MUST reject postMessage requests from any other source origin, to prevent cross-site scripting attacks.

The postMessage from the RP iframe delivers the following concatenation as the data:

- Client ID + " " + Session State

It also has to be able to receive the postMessage back from the OP iframe. The received data will either be `changed` or `unchanged` unless the syntax of the message sent was determined by the OP to be malformed, in which case the received data will be `error`. Upon receipt of `changed`, the RP MUST perform re-authentication with `prompt=none` to obtain the current session state at the OP. Upon receipt of `error`, the RP MUST NOT perform re-authentication with `prompt=none`, so as to not cause potential infinite loops that generate network traffic to the OP.

Following is non-normative example pseudo-code for the RP iframe:

```
  var stat = "unchanged";
  var mes = client_id + " " + session_state;
  var targetOrigin = "https://server.example.com"; //
Validates origin
  var opFrameId = "op";
  var timerID;

  function check_session()   {
    var win =
window.parent.frames[opFrameId].contentWindow
    win.postMessage(mes, targetOrigin);
  }

  function setTimer() {
    check_session();
    timerID = setInterval(check_session, 5 * 1000);
  }

  window.addEventListener("message", receiveMessage,
false);

  function receiveMessage(e) {
    if (e.origin !== targetOrigin) {
      return;
    }
    stat = e.data;
```