|  |
| --- |
| **[TOC](" \l "toc)** |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Draft | N. Sakimura, Ed. | |  | NRI | |  | J. Bradley, Ed. | |  | Independent | |  | B. de Medeiros | |  | Google | |  | M. Jones | |  | Microsoft | |  | E. Jay | |  | MGI1 | |  | C. Mortimore | |  | Salesforce | |  | October 14, 2011 | |

# OpenID Connect Basic Client 1.0 - draft 14

### Abstract

OpenID Connect 1.0 is an identity layer on top of OAuth 2.0 protocol. It allows a web site to verify the identity of the user based on the authentication performed by an Authorization Server, as well as to obtain basic profile information about the user in an interoperable and RESTful manner.

OpenID Connect Basic Client is a profile of the OpenID Connect Standard 1.0 Specification that is designed to be easy to read and implement for Relying Parties. OpenID Providers should consult the main specification. This profile omits implementation and security considerations for OpenID Providers.

### Table of Contents

[**1.**](#rnc)  Requirements Notation and Conventions  
[**2.**](#terminology)  Terminology  
[**3.**](#anchor1)  Protocol Flows  
    [**3.1.**](#anchor2)  OpenID Connect Scopes  
    [**3.2.**](#anchor3)  Implicit Flow  
        [**3.2.1.**](#rf_prep)  Client Prepares an Authorization Request  
        [**3.2.2.**](#implicit_req)  Client Sends a Request to the Authorization Server  
        [**3.2.3.**](#anchor4)  Authorization Server Obtains the End-User Consent/Authorization  
        [**3.2.4.**](#implicit_res)  Authorization Server Sends the End-User Back to the Client  
    [**3.3.**](#anchor5)  Check ID Endpoint  
        [**3.3.1.**](#anchor6)  Check ID Request  
        [**3.3.2.**](#anchor8)  Check ID Response  
        [**3.3.3.**](#anchor10)  Error Codes  
[**4.**](#userinfo)  UserInfo Endpoint  
    [**4.1.**](#anchor11)  Requesting UserInfo  
    [**4.2.**](#id_res)  Client Receives UserInfo Response  
        [**4.2.1.**](#anchor12)  Error Response  
[**5.**](#disco_reg)  Discovery and Registration  
[**6.**](#qss)  Query String Serialization  
[**7.**](#security_considerations)  Security Considerations  
    [**7.1.**](#assertion_manufacture)  Assertion Manufacture/Modification  
    [**7.2.**](#assertion_disclosure)  Assertion Disclosure  
    [**7.3.**](#assertion_redirect)  Assertion Redirect  
    [**7.4.**](#assertion_reuse)  Assertion Reuse  
    [**7.5.**](#assertion_substitution)  Assertion Substitution  
    [**7.6.**](#auth_req_disclosure)  Authentication Request Disclosure  
    [**7.7.**](#authn_proc_threats)  Authentication Process Threats  
    [**7.8.**](#anchor13)  Implicit Flow Threats  
    [**7.9.**](#anchor14)  Availability  
[**8.**](#privacy_considerations)  Privacy Considerations  
[**9.**](#IANA)  IANA Considerations  
[**10.**](#rfc.references1)  Normative References  
[**Appendix A.**](#anchor16)  Acknowledgements  
[**Appendix B.**](#anchor17)  Document History  
[**§**](#rfc.authors)  Authors' Addresses

|  |
| --- |
| [**TOC**](#toc) |

### 1.  Requirements Notation and Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [[RFC2119] (Bradner, S., “Key words for use in RFCs to Indicate Requirement Levels,” March 1997.)](#RFC2119) .

Throughout 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.

|  |
| --- |
| [**TOC**](#toc) |

### 2.  Terminology

The following definitions define additional terminology used in this specification in addition to those defined in [OAuth 2.0 (Hammer-Lahav, E., Ed., Recordon, D., and D. Hardt, “OAuth 2.0 Authorization Protocol,” September 2011.)](#OAuth.2.0) [OAuth.2.0].

Relying Party (RP)

An application requiring identity information from an OpenID Provider.

OpenID Provider (OP)

A service capable of providing identity information to a Relying Party.

Assertion

A set of Claims about the End-User that are attested to by the OpenID Provider and Resource Servers.

Claim

A piece of information about an Entity that a Claims Provider asserts about that Entity.

Claims Provider

An Authorization Server that can return claims about a user.

Entity

Something that has separate and distinct existence and that can be identified in context.

ID Token

A token that contains information about the authentication event. It is a signed token, but can be treated as opaque by clients that use the Check ID Endpoint. Relying Parties wanting to process the token directly should refer to the OpenID Connect Standard 1.0 specification.

Check ID Endpoint

A resource that, when presented with an ID Token by the client, returns authentication information about the user session represented by that ID Token.

UserInfo Endpoint

A protected resource that, when presented with an access token by the client, returns authorized information about the user represented by that access token.

|  |
| --- |
| [**TOC**](#toc) |

### 3.  Protocol Flows

Authorization requests can follow one of two paths; the implicit flow or the authorization code flow. The authorization code flow is suitable for clients that can securely maintain a client secret between themselves and the Authorization Server whereas, the implicit flow is suitable for clients that cannot. Clients that do not support TLS MUST use the authorization code flow to prevent the interception of access tokens.

The OpenID Connect Basic Client profile only documents clients using the implicit flow. OpenID Providers MUST support both flows. Clients wanting to use the authorization code flow and OpenID Providers should consult the OpenID Connect Standard 1.0 specification.

|  |
| --- |
| [**TOC**](#toc) |

### 3.1.  OpenID Connect Scopes

OpenID Connect clients use scopes as defined in 3.3 of [OAuth 2.0 (Hammer-Lahav, E., Ed., Recordon, D., and D. Hardt, “OAuth 2.0 Authorization Protocol,” September 2011.)](#OAuth.2.0) [OAuth.2.0] to specify what access privileges are requested for access tokens. The scopes associated with access tokens determine what resources will be available when they are used to access OAuth 2.0 protected endpoints. For OpenID Connect, scopes request what information is to be made available from the UserInfo Endpoint, and to request an id\_token. OAuth 2.0 allows additional scopes to be specified, as extensions. This specification only describes scopes that are part of OpenID Connect.

OpenID Connect defines the following scopes:

openid

REQUIRED. Informs the Authorization Server that the client is making an OpenID request. If the openid scope is not specified, the server SHOULD treat the request as a generic OAuth 2.0 request, and perform no OpenID Connect processing.

profile

OPTIONAL. Requests default profile information.

email

OPTIONAL. Requests an email address.

address

OPTIONAL. Requests address information.

Multiple scopes MAY be requested by creating a space delimited, case sensitive list of scope values.

The User may decline a scope request by the client.

To increase conversion, a site may elect to only request a subset of the information from the UserInfo Endpoint.

The following is a non-normative example of a Scope Request.

scope=openid profile email phone

|  |
| --- |
| [**TOC**](#toc) |

### 3.2.  Implicit Flow

The implicit flow consists of the following steps:

1. Client Prepares an Authorization Request containing the desired request parameters.
2. Client sends a request to the Authorization Server.
3. Authorization Server Authenticates the End-User.
4. Authorization Server Obtains the End-User Consent/Authorization.
5. Authorization Server Sends the End-User back to the Client with an Access Token and ID Token.

|  |
| --- |
| [**TOC**](#toc) |

### 3.2.1.  Client Prepares an Authorization Request

When the user wishes to access a Protected Resource, and the End-User Authorization has not yet been obtained, the Client prepares an Authorization Request to the authorization endpoint.

The scheme used in the Authorization Endpoint URL MUST be HTTPS.

Clients MAY construct the request using the HTTP GET or the HTTP POST method as defined in [RFC 2616 (Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, “Hypertext Transfer Protocol -- HTTP/1.1,” June 1999.)](#RFC2616) [RFC2616].

If using the HTTP GET method, the parameters MUST be serialized using [Query String Serialization (Query String Serialization)](#qss). If using the HTTP POST method, the request parameters MUST be added to the HTTP request entity-body using the application/x-www-form-urlencoded format as defined by [[W3C.REC‑html401‑19991224] (Raggett, D., Hors, A., and I. Jacobs, “HTML 4.01 Specification,” December 1999.)](#W3C.REC-html401-19991224).

This profile further constrains the following request parameters:

response\_type

It MUST include token and id\_token, as a space separated list. This requests both an access\_token and id\_token to be returned in the URL fragment of the response.

Other REQUIRED parameters in the request include the following:

client\_id

The OAuth client identifier.

scope

It MUST include openid as one of the space separated strings. Optional scope strings of profile, email, and address are also supported.

redirect\_uri

A redirection URI where the response will be sent. This MUST be pre-registered with the provider.

nonce

A random, unique string value used to mitigate the replay attack. This value will be returned from the Check ID Endpoint.

The request MAY contain the following optional parameters:

state

RECOMENDED. An opaque value used to maintain state between the request and the callback, used to protect against XSRF attacks.

display

A string value that specifies how the Authorization Server displays the authentication page to the user. The following values are supported:

none

This value informs the Authorization Server that it MUST NOT display any authentication or confirmation user interface pages. An error is returned if either the user is not already authenticated or the client does not have pre-configured approval for the requested scopes. This can be used as a method to check for existing authentication and/or approval.

prompt

A space delimited, case sensitive list of string values that specifies how the Authorization Server prompts the user for reauthentication and reapproval. The possible values are:

login

The Authorization Server MUST prompt the user for reauthentication.

consent

The Authorization Server MUST prompt the user for reapproval before returning information to the client.

select\_account

The Authorization Server MUST prompt the user to select a user account if the account has multiple accounts associated with it.

This can be used by the client to make sure that the user is still present for the current session or to bring attention to the request. If this parameter is used in conjunction with the display parameter set to "none", an error is returned.

The following is a non-normative example of an Authorization Request URL. Note that the line wraps within the values are for display purpose only:

https://server.example.com/authorize?

response\_type=token%20id\_token

&client\_id=s6BhdRkqt3

&redirect\_uri=https%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb

&scope=openid%20profile

&state=af0ifjsldkj

&nonce=n-0S6\_WzA2Mj

|  |
| --- |
| [**TOC**](#toc) |

### 3.2.2.  Client Sends a Request to the Authorization Server

Having constructed the Authorization Request, the client sends it to the Authorization Endpoint. This MAY happen via HTTPS redirect, hyperlinking, or any other secure means of directing the User-Agent to the URL.

Following is a non-normative example using HTTP redirect. Note: Line wraps are for display purpose only.

HTTP/1.1 302 Found

Location: https://server.example.com/authorize?

response\_type=token%20id\_token

&client\_id=s6BhdRkqt3

&redirect\_uri=https%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb

&scope=openid%20profile

&state=af0ifjsldkj

&nonce=n-0S6\_WzA2Mj

|  |
| --- |
| [**TOC**](#toc) |

### 3.2.3.  Authorization Server Obtains the End-User Consent/Authorization

The Authorization Server obtains an authorization decision, for the requested scopes. This can done by presenting the user with a dialogue that allows the user to recognize what he is consenting to and obtain his consent or by establishing approval via other means (for example, via previous administrative approval).

The openid scope grants the RP access to the user identifier of the authenticated user of the session.

All other scopes are optional. It is up to the OpenID Provider to determine if an error should be returned in the case of the user declining to authorize scopes other than openid.

|  |
| --- |
| [**TOC**](#toc) |

### 3.2.4.  Authorization Server Sends the End-User Back to the Client

Once the authorization is determined, the Authorization Server returns a positive or negative response.

|  |
| --- |
| [**TOC**](#toc) |

### 3.2.4.1.  End-User Grants Authorization

If the resource owner grants the access request, the Authorization Server issues an access token and delivers it to the client by adding the following parameters to the fragment component of the redirection URI using the application/x-www-form-urlencoded format as defined in Section 4.2.2 of [OAuth 2.0 (Hammer-Lahav, E., Ed., Recordon, D., and D. Hardt, “OAuth 2.0 Authorization Protocol,” September 2011.)](#OAuth.2.0) [OAuth.2.0]

In the implicit flow, the entire response is returned in the fragment component of the redirect URL, as defined in 4.2.2 of [OAuth 2.0 (Hammer-Lahav, E., Ed., Recordon, D., and D. Hardt, “OAuth 2.0 Authorization Protocol,” September 2011.)](#OAuth.2.0) [OAuth.2.0]

access\_token

REQUIRED. The Access Token for the UserInfo Endpoint.

token\_type

REQUIRED. The value MUST be "bearer"

id\_token

REQUIRED. The ID Token for the Check ID Endpoint.

state

REQUIRED if the state parameter is present in the request. Clients MUST verify that the state value is equal to the exact value of state parameter in the Authorization Request.

expires\_in

OPTIONAL. The expiration time in seconds of the access\_token

The client can then use the Access Token to access protected resources at Resource Servers.

The following is a non-normative example. Line wraps after the second line is for the display purpose only:

HTTP/1.1 302 Found

Location: https://client.example.com/#

access\_token=SlAV32hkKG&

token\_type=bearer&

id\_token=1234567.SlAV32hkKG.abcde1234&

expires\_in=3600&

&state=af0ifjsldkj

|  |
| --- |
| [**TOC**](#toc) |

### 3.2.4.2.  End-User Denies Authorization or Invalid Request

If the user denies the authorization or the user authentication fails, the Authorization Server MUST return the negative authorization response as defined in 4.2.2.1 of [OAuth 2.0 (Hammer-Lahav, E., Ed., Recordon, D., and D. Hardt, “OAuth 2.0 Authorization Protocol,” September 2011.)](#OAuth.2.0) [OAuth.2.0]. No other parameters SHOULD be returned.

|  |
| --- |
| [**TOC**](#toc) |

### 3.3.  Check ID Endpoint

The Check ID Endpoint validates the id\_token and returns a text [JSON (Crockford, D., “The application/json Media Type for JavaScript Object Notation (JSON),” July 2006.)](#RFC4627) [RFC4627] object which contains information about the end user session associated with the supplied id\_token.

Clients MUST verify the validity of the id\_token using the Check ID Endpoint.

The id\_token is used to manage the signon event and user identifier, separately from access to the UserInfo Endpoint and other OAuth 2.0 protected resources that the user is granting access to. The id\_token is audience restricted to a particular client via the audience and nonce.

A full explanation of how to use the id\_token to perform session management can be found in the [OpenID Connect Session Management 1.0 (de Medeiros, B., “OpenID Connect Session Management 1.0,” September 2011.)](#OpenID.Session) [OpenID.Session]

|  |
| --- |
| [**TOC**](#toc) |

### 3.3.1.  Check ID Request

To request the information about the authentication performed on the user and verify the id\_token, the following parameters MAY be sent to the Check ID Endpoint as query parameters in a GET or as form encoded in a POST request:

id\_token

REQUIRED. The id\_token obtained from an OpenID Connect authorization request.

The following is a non-normative example of a request to the Check ID Endpoint:

Request:

GET /op/check\_id?id\_token=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiIsImtpZCI6

ImNsaWVudC5leGFtcGxlLmNvbSJ9.eyJpc3N1ZXIiOiJodHRwOlwvXC9zZXJ2ZXIuZXhhbXBs

ZS5jb20iLCJjbGllbnRfaWQiOiJjbGllbnQuZXhhbXBsZS5jb20iLCJhdWRpZW5jZSI6ImNsa

WVudC5leGFtcGxlLmNvbSIsImlkIjoidXNlcl8yMzQyMzQiLCJleHAiOjEzMDM4NTI4ODB9.a

JwagC6501Da-zK-X8Az9B-Y625aSEfxVuBpFEDjOxQ

Host: server.example.com

|  |
| --- |
| [**TOC**](#toc) |

### 3.3.1.1.  Request Verification

Clients MUST verify the request with the following:

1. When connecting to the Endpoint, verify that the TLS peer matches that of the Endpoint.
2. Make sure that the Endpoint that the client connected to is a trusted Check ID Endpoint for the Entity via pre-configuration meta-data or discovery.

|  |
| --- |
| [**TOC**](#toc) |

### 3.3.2.  Check ID Response

The response is a text [JSON (Crockford, D., “The application/json Media Type for JavaScript Object Notation (JSON),” July 2006.)](#RFC4627) [RFC4627] object using the application/json media type with the following parameters.

iss

REQUIRED. The unique identifier of the issuer of the response.

user\_id

REQUIRED. A locally unique and never reassigned identifier for the user, which is intended to be consumed by the Client. e.g. 24400320 or AItOawmwtWwcT0k51BayewNvutrJUqsvl6qs7A4. It MUST NOT exceed 255 ASCII characters in length.

aud

REQUIRED. This member identifies the audience that this ID Token is intended for. It is client\_id of the RP.

exp

REQUIRED. Type Integer. Identifies the expiration time on or after which the ID Token MUST NOT be accepted for processing. The processing of this parameter requires that the current date/time MUST be before the expiration date/time listed in the value. Implementers MAY provide for some small leeway, usually no more than a few minutes, to account for clock skew. The value is number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the desired date/time. See [RFC 3339 (Klyne, G., Ed. and C. Newman, “Date and Time on the Internet: Timestamps,” July 2002.)](#RFC3339) [RFC3339] for details regarding date/times in general and UTC in particular.

iso29115

OPTIONAL. (entity authentication assurance): Specifies the X.eaa / [ISO/IEC29115 (McCallister, E., “ITU-T Recommendation X.eaa | ISO/IEC 2nd CD 29115 -- Information technology - Security techniques - Entity authentication assurance framework,” .)](#ISO29115) [ISO29115] entity authentication assurance level of the authentication performed.

nonce

REQUIRED. Clients MUST verify that the nonce value is equal to the exact value of nonce parameter in the Authorization Request.

The following is a non-normative example of a request to the Check ID Endpoint:

Request:

GET /op/check\_id?id\_token=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiIsImtpZCI6

ImNsaWVudC5leGFtcGxlLmNvbSJ9.eyJpc3N1ZXIiOiJodHRwOlwvXC9zZXJ2ZXIuZXhhbXBs

ZS5jb20iLCJjbGllbnRfaWQiOiJjbGllbnQuZXhhbXBsZS5jb20iLCJhdWRpZW5jZSI6ImNsa

WVudC5leGFtcGxlLmNvbSIsImlkIjoidXNlcl8yMzQyMzQiLCJleHAiOjEzMDM4NTI4ODB9.a

JwagC6501Da-zK-X8Az9B-Y625aSEfxVuBpFEDjOxQ

Host: server.example.com

Response:

HTTP/1.1 200 OK

Content-Type: application/json

{

"iss": "http://server.example.com",

"user\_id": "Jane Doe",

"aud": "http://client.example.net",

"exp":1311281970

}

|  |
| --- |
| [**TOC**](#toc) |

### 3.3.2.1.  Response Verification

To verify the validity of the Response, the client MUST do the following:

1. The Check ID Endpoint has not returned an error for the ID Token being expired or invalid.
2. Check that the iss is equal to that of the pre-configured or discovered Issuer Identifier for the user session.
3. Check that the returned nonce is equal to the nonce in the Authorization Request.
4. Verify that the aud (audience) contained within the response is the client's client\_id.

|  |
| --- |
| [**TOC**](#toc) |

### 3.3.3.  Error Codes

In addition to the error codes defined in Section 5.2 of [OAuth 2.0 (Hammer-Lahav, E., Ed., Recordon, D., and D. Hardt, “OAuth 2.0 Authorization Protocol,” September 2011.)](#OAuth.2.0) [OAuth.2.0], this specification defines the following error codes.

invalid\_id\_token

The ID Token is not valid for the requested resource, malformed, is in an incorrect format, or expired.

|  |
| --- |
| [**TOC**](#toc) |

### 4.  UserInfo Endpoint

To obtain the additional attributes and tokens, the client makes a GET or POST request to the UserInfo Endpoint.

NOTE: The UserInfo Endpoint response is not guaranteed to be about the Subject in the session. Therefore, it MUST NOT be used as an assertion about the user in the session unless the user\_id matches the user\_id in the ID Token.

|  |
| --- |
| [**TOC**](#toc) |

### 4.1.  Requesting UserInfo

Clients MAY send requests with the following parameters to the UserInfo Endpoint to obtain further information about the user. The UserInfo Endpoint is an [OAuth 2.0 (Hammer-Lahav, E., Ed., Recordon, D., and D. Hardt, “OAuth 2.0 Authorization Protocol,” September 2011.)](#OAuth.2.0) [OAuth.2.0] protected resource that complies with the [Bearer Token (Jones, M., Ed., Recordon, D., and D. Hardt, “The OAuth 2.0 Protocol: Bearer Tokens,” September 2011.)](#OAuth.2.0.Bearer) [OAuth.2.0.Bearer] specification. As such, the access token SHOULD be specified via the HTTP Authorization header.

access\_token

REQUIRED. The access\_token obtained from an OpenID Connect authorization request. This parameter MUST only be sent using one method through either HTTP Authorization header or query string.

schema

REQUIRED. The schema in which the data is to be returned. The only defined value is openid. If the value of this parameter is omitted, or not openid, the response may be a proprietary schema to support backwards compatibility. A URL MAY be passed to define custom schemes not specified by short names. Custom scheme names and responses are out of scope for this specification.

id

This identifier is reserved for backwards compatibility. It MUST be ignored by the endpoint if the openid schema is used.

The following is a non-normative example. Line wraps are for display purpose only:

GET /userinfo HTTP/1.1

Host: server.example.com

Authorization: Bearer vF9dft4qmT

|  |
| --- |
| [**TOC**](#toc) |

### 4.2.  Client Receives UserInfo Response

If the requested schema is openid, the response MUST return a plain text [JSON (Crockford, D., “The application/json Media Type for JavaScript Object Notation (JSON),” July 2006.)](#RFC4627) [RFC4627] structure that contains a set of members that are a subset of those defined below. Additional members (not specified below) MAY also be returned.

The members may be represented in multiple languages and scripts. To specify the languages and scripts, [BCP47 (Phillips, A. and M. Davis, “Tags for Identifying Languages,” September 2009.)](#RFC5646) [RFC5646] language tags MUST be added to each member names delimited by a #, e.g., familyName#ja-Kana-JP for expressing Family Name in Katakana in Japanese, which is commonly used to index and represent the phonetics of the Kanji representation of the same represented as familyName#ja-Hani-JP.

|  |  |  |
| --- | --- | --- |
| **Member** | **Type** | **Description** |
| user\_id | string | Identifier for the user at the issuer. |
| name | string | User's full name in displayable form including all name parts, ordered according to user's locale and preferences. |
| given\_name | string | Given name or first name of the user. |
| family\_name | string | Surname or last name of the user. |
| middle\_name | string | Middle name of the user. |
| nickname | string | Casual name of the user that may or may not be the same as the given\_name. For instance, a nickname value of Mike might be returned alongside a given\_name value of Michael. |
| profile | string | URL of user's profile page. |
| picture | string | URL of the user's profile picture. |
| website | string | URL of user's web page or blog. |
| email | string | The user's preferred e-mail address. |
| verified | boolean | True if the user's e-mail address has been verified; otherwise false. |
| gender | string | The user's gender: female or male. |
| birthday | string | The user's birthday, represented as a date string in MM/DD/YYYY format. The year MAY be 0000, indicating that it is omitted. |
| zoneinfo | string | String from zoneinfo [[zoneinfo] (Public Domain, “The tz database,” June 2011.)](#zoneinfo) timezone database. For example, Europe/Paris or America/Los\_Angeles. |
| locale | string | The user's locale, represented as an [RFC 5646 (Phillips, A. and M. Davis, “Tags for Identifying Languages,” September 2009.)](" \l "RFC5646) [RFC5646] language tag. This is typically an [ISO 639-1 Alpha-2 (International Organization for Standardization, “ISO 639-1:2002. Codes for the representation of names of languages -- Part 1: Alpha-2 code,” 2002.)](#ISO639-1) [ISO639‑1] language code in lowercase and an [ISO 3166-1 Alpha-2 (International Organization for Standardization, “ISO 3166-1:1997. Codes for the representation of names of countries and their subdivisions -- Part 1: Country codes,” 1997.)](#ISO3166-1) [ISO3166‑1] country code in uppercase, separated by a dash. For example, en-US or fr-CA. As a compatibility note, some implementations have used an underscore as the separator rather than a dash, for example, en\_US; Implementations MAY choose to accept this locale syntax as well. |
| phone\_number | string | The user's preferred telephone number. For example, +1 (425) 555-1212 or +56 (2) 687 2400. |
| address | JSON object | The user's preferred address. The value of the address member is a [JSON (Crockford, D., “The application/json Media Type for JavaScript Object Notation (JSON),” July 2006.)](#RFC4627) [RFC4627] structure containing some or all of these string-valued fields: formatted, street\_address, locality, region, postal\_code, and country. The street\_address field MAY contain multiple lines if the address representation requires it. Implementations MAY return only a subset of the fields of an address, depending upon the information available and the user's privacy preferences. For example, the country and region might be returned without returning more fine-grained address information. |
| updated\_time | string | Time the user's information was last updated, represented as a [RFC 3339 (Klyne, G., Ed. and C. Newman, “Date and Time on the Internet: Timestamps,” July 2002.)](#RFC3339) [RFC3339] datetime. For example, 2011-01-03T23:58:42+0000. |

|  |
| --- |
| **Table 1: Reserved Member Definitions** |

Following is a non-normative example of such response:

{

"name": "Jane Doe"

"given\_name": "Jane",

"family\_name": "Doe",

"email": "janedoe@example.com",

"picture": "http://example.com/janedoe/me.jpg"

}

|  |
| --- |
| [**TOC**](#toc) |

### 4.2.1.  Error Response

When some error condition arises, the UserInfo Endpoint returns the Error Response. In addition to the standard [OAuth 2.0 (Hammer-Lahav, E., Ed., Recordon, D., and D. Hardt, “OAuth 2.0 Authorization Protocol,” September 2011.)](#OAuth.2.0) [OAuth.2.0] errors, the UserInfo Endpoint may return the following errors:

unsupported\_schema

The requested schema is unsupported.

|  |
| --- |
| [**TOC**](#toc) |

### 5.  Discovery and Registration

Some OpenID Connect installations can use a pre-configured set of OpenID Providers and/or Relying Parties. In those cases, it may not be necessary to support dynamic discovery of information about identities or services or dynamic registration of clients.

However, if installations choose to support unanticipated interactions between Relying Parties and OpenID Providers that do not have pre-configured relationships, they SHOULD accomplish this by implementing the facilities defined in the [OpenID Connect Discovery 1.0 (Sakimura, N., Bradley, J., Jones, M., and E. Jay, “OpenID Connect Discovery 1.0,” September 2011.)](#OpenID.Discovery) [OpenID.Discovery] and [OpenID Connect Dynamic Client Registration 1.0 (Sakimura, N., Bradley, J., Ed., and M. Jones, “OpenID Connect Dynamic Client Registration 1.0,” September 2011.)](#OpenID.Registration) [OpenID.Registration] specifications.

|  |
| --- |
| [**TOC**](#toc) |

### 6.  Query String Serialization

In order to serialize the parameters using the query string serialization, the client constructs the string by adding the parameters and values to the query component using the application/x-www-form-urlencoded format as defined by [[W3C.REC‑html401‑19991224] (Raggett, D., Hors, A., and I. Jacobs, “HTML 4.01 Specification,” December 1999.)](#W3C.REC-html401-19991224).

Following is a non-normative example of such serialization:

GET /authorize?scope=openid&response\_type=code

&client\_id=s6BhdRkqt3

&redirect\_uri=https%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb HTTP/1.1

Host: server.example.com

|  |
| --- |
| [**TOC**](#toc) |

### 7.  Security Considerations

In addition to the Security Considerations in [OAuth 2.0 (Hammer-Lahav, E., Ed., Recordon, D., and D. Hardt, “OAuth 2.0 Authorization Protocol,” September 2011.)](#OAuth.2.0) [OAuth.2.0], the following are the list of threats and remedies that were considered for this specification.

|  |
| --- |
| [**TOC**](#toc) |

### 7.1.  Assertion Manufacture/Modification

An assertion is the result of the authentication performed by the Authorization Server that was provided to the client. The assertion is used to pass information about the user or the authentication process from the Authorization Server to the client.

To mitigate this attack, the assertion may be sent over a protected channel such as TLS/SSL. In order to protect the integrity of assertions from malicious attack, the authorization server MUST be authenticated. In this specification, the assertion is always sent over TLS/SSL protected channel.

For details of the threat, see [[SP800‑63] (National Institute of Standards and Technology, “NIST SP800-63rev.1: Electronic Authentication Guideline,” .)](#SP800-63).

|  |
| --- |
| [**TOC**](#toc) |

### 7.2.  Assertion Disclosure

This profile is subject to assertion disclosure in the user's browser, if it is compromised. Other OpenID Connect profiles should be used if this threat needs to be mitigated.

For details of the threat, see [[SP800‑63] (National Institute of Standards and Technology, “NIST SP800-63rev.1: Electronic Authentication Guideline,” .)](#SP800-63).

|  |
| --- |
| [**TOC**](#toc) |

### 7.3.  Assertion Redirect

To mitigate this threat, the assertion includes the identity of the client for whom it was generated as client\_id. The client verifies that incoming assertions include its identity as the recipient of the assertion.

For details of the threat, see [[SP800‑63] (National Institute of Standards and Technology, “NIST SP800-63rev.1: Electronic Authentication Guideline,” .)](#SP800-63).

|  |
| --- |
| [**TOC**](#toc) |

### 7.4.  Assertion Reuse

The assertion includes a timestamp and a short lifetime of validity. The client checks the timestamp and lifetime values to ensure that the assertion is currently valid.

The use of a nonce in the request is REQUIRED. The response from the Check ID Endpoint contains the nonce sent in the authorization request. This SHOULD be checked against a list of already received ID assertions to check for replays.

For details of the threat, see [[SP800‑63] (National Institute of Standards and Technology, “NIST SP800-63rev.1: Electronic Authentication Guideline,” .)](#SP800-63).

|  |
| --- |
| [**TOC**](#toc) |

### 7.5.  Assertion Substitution

Responses to assertion requests is bound to the corresponding requests by message order in HTTP, as both assertions and requests are protected by TLS that can detect and disallow malicious reordering of packets.

For details of the threat, see [[SP800‑63] (National Institute of Standards and Technology, “NIST SP800-63rev.1: Electronic Authentication Guideline,” .)](#SP800-63).

|  |
| --- |
| [**TOC**](#toc) |

### 7.6.  Authentication Request Disclosure

Since the authentication request is sent over a protected channel, the disclosure may only happen at the User-Agent where the information is decrypted.

For details of the threat, see [[SP800‑63] (National Institute of Standards and Technology, “NIST SP800-63rev.1: Electronic Authentication Guideline,” .)](#SP800-63).

|  |
| --- |
| [**TOC**](#toc) |

### 7.7.  Authentication Process Threats

In the category of Authentication Process Threats, the following threats exist:

* Online Guessing
* Phishing
* Pharming
* Eavesdropping
* Replay
* Session Hijacking
* Man-in-the-Middle

The authentication process, per se, as described in [[SP800‑63] (National Institute of Standards and Technology, “NIST SP800-63rev.1: Electronic Authentication Guideline,” .)](#SP800-63) is out of scope for this protocol, but care SHOULD be taken to achieve appropriate protection.

For details of the threat, see [[SP800‑63] (National Institute of Standards and Technology, “NIST SP800-63rev.1: Electronic Authentication Guideline,” .)](#SP800-63).

|  |
| --- |
| [**TOC**](#toc) |

### 7.8.  Implicit Flow Threats

In the implicit flow, the access token is returned in the fragment part of the client's redirect\_uri through HTTPS. Thus it is protected between the Authorization Server and the User-Agent, and User-Agent and the Client. The only the place it can be captured is the User-Agent where the TLS session is terminated, and is possible if the User-Agent is infested by malware.

|  |
| --- |
| [**TOC**](#toc) |

### 7.9.  Availability

When the Authorization Server is down, users will likely be unable to access it. To mitigate this risk, the client SHOULD allow users to associate multiple Authorization Servers.

|  |
| --- |
| [**TOC**](#toc) |

### 8.  Privacy Considerations

The UserInfo response typically contains Personally Identifiable Information. As such, user consent for the release of the information for the specified purpose SHOULD be obtained at or prior to the authorization time in accordance with relevant regulations. The purpose of use is typically registered in association with the redirect\_uri.

Only necessary UserInfo data should be stored at the client and the client SHOULD associate the received data with the purpose of use statement.

The Resource Server SHOULD make the UserInfo access log available to the user so that the user can monitor who accessed his data.

To protect the user from a possible correlation among clients, the use of a Pairwise Pseudonymous Identifier (PPID) as the user\_id SHOULD be considered.

|  |
| --- |
| [**TOC**](#toc) |

### 9.  IANA Considerations

This document makes no request of IANA.

|  |
| --- |
| [**TOC**](#toc) |

### 10. Normative References

|  |  |
| --- | --- |
| **[ISO29115]** | McCallister, E., “ITU-T Recommendation X.eaa | ISO/IEC 2nd CD 29115 -- Information technology - Security techniques - Entity authentication assurance framework,” ISO/IEC 29115. |
| **[ISO3166-1]** | International Organization for Standardization, “[ISO 3166-1:1997. Codes for the representation of names of countries and their subdivisions -- Part 1: Country codes](http://www.w3.org/WAI/ER/IG/ert/iso639.htm),” 1997. |
| **[ISO639-1]** | International Organization for Standardization, “ISO 639-1:2002. Codes for the representation of names of languages -- Part 1: Alpha-2 code,” 2002. |
| **[OAuth.2.0]** | Hammer-Lahav, E., Ed., Recordon, D., and D. Hardt, “[OAuth 2.0 Authorization Protocol](http://tools.ietf.org/html/draft-ietf-oauth-v2),” September 2011. |
| **[OAuth.2.0.Bearer]** | Jones, M., Ed., Recordon, D., and D. Hardt, “[The OAuth 2.0 Protocol: Bearer Tokens](http://tools.ietf.org/html/draft-ietf-oauth-v2-bearer),” September 2011. |
| **[OpenID.Discovery]** | Sakimura, N., Bradley, J., Jones, M., and E. Jay, “[OpenID Connect Discovery 1.0](http://openid.net/specs/openid-connect-discovery-1_0.html),” September 2011. |
| **[OpenID.Registration]** | Sakimura, N., Bradley, J., Ed., and M. Jones, “[OpenID Connect Dynamic Client Registration 1.0](http://openid.net/specs/openid-connect-registration-1_0.html),” September 2011. |
| **[OpenID.Session]** | [de Medeiros, B.](mailto:breno@google.com), “[OpenID Connect Session Management 1.0](http://openid.net/specs/openid-connect-session-1_0.html),” September 2011. |
| **[RFC2119]** | [Bradner, S.](mailto:sob@harvard.edu), “[Key words for use in RFCs to Indicate Requirement Levels](http://tools.ietf.org/html/rfc2119),” BCP 14, RFC 2119, March 1997 ([TXT](http://www.rfc-editor.org/rfc/rfc2119.txt), [HTML](http://xml.resource.org/public/rfc/html/rfc2119.html), [XML](http://xml.resource.org/public/rfc/xml/rfc2119.xml)). |
| **[RFC2616]** | [Fielding, R.](mailto:fielding@ics.uci.edu), [Gettys, J.](mailto:jg@w3.org), [Mogul, J.](mailto:mogul@wrl.dec.com), [Frystyk, H.](mailto:frystyk@w3.org), [Masinter, L.](mailto:masinter@parc.xerox.com), [Leach, P.](mailto:paulle@microsoft.com), and [T. Berners-Lee](mailto:timbl@w3.org), “[Hypertext Transfer Protocol -- HTTP/1.1](http://tools.ietf.org/html/rfc2616),” RFC 2616, June 1999 ([TXT](http://www.rfc-editor.org/rfc/rfc2616.txt), [PS](http://www.rfc-editor.org/rfc/rfc2616.ps), [PDF](http://www.rfc-editor.org/rfc/rfc2616.pdf), [HTML](http://xml.resource.org/public/rfc/html/rfc2616.html), [XML](http://xml.resource.org/public/rfc/xml/rfc2616.xml)). |
| **[RFC3339]** | [Klyne, G., Ed.](mailto:GK@ACM.ORG) and [C. Newman](mailto:chris.newman@sun.com), “[Date and Time on the Internet: Timestamps](http://tools.ietf.org/html/rfc3339),” RFC 3339, July 2002 ([TXT](http://www.rfc-editor.org/rfc/rfc3339.txt), [HTML](http://xml.resource.org/public/rfc/html/rfc3339.html), [XML](http://xml.resource.org/public/rfc/xml/rfc3339.xml)). |
| **[RFC4627]** | Crockford, D., “[The application/json Media Type for JavaScript Object Notation (JSON)](http://tools.ietf.org/html/rfc4627),” RFC 4627, July 2006 ([TXT](http://www.rfc-editor.org/rfc/rfc4627.txt)). |
| **[RFC5646]** | Phillips, A. and M. Davis, “[Tags for Identifying Languages](http://tools.ietf.org/html/rfc5646),” BCP 47, RFC 5646, September 2009 ([TXT](http://www.rfc-editor.org/rfc/rfc5646.txt)). |
| **[SP800-63]** | National Institute of Standards and Technology, “[NIST SP800-63rev.1: Electronic Authentication Guideline](http://csrc.nist.gov/publications/drafts/800-63-rev1/SP800-63-Rev1-Draft3_June2011.pdf),” NIST SP800-63. |
| **[W3C.REC-html401-19991224]** | Raggett, D., Hors, A., and I. Jacobs, “[HTML 4.01 Specification](http://www.w3.org/TR/1999/REC-html401-19991224),” World Wide Web Consortium Recommendation REC-html401-19991224, December 1999 ([HTML](http://www.w3.org/TR/1999/REC-html401-19991224)). |
| **[zoneinfo]** | Public Domain, “[The tz database](http://www.twinsun.com/tz/tz-link.htm),” June 2011. |

|  |
| --- |
| [**TOC**](#toc) |

### Appendix A.  Acknowledgements

The OpenID Community would like to thank the following people for the work they've done in the drafting and editing of this specification.

Axel Nennker (axel.nennker@telekom.de), Deutsche Telekom

Casper Biering (cb@peercraft.com), Peercraft

John Bradley (jbradely@mac.com), Protiviti Government Services

Breno de Medeiros (breno@gmail.com), Google

George Fletcher (gffletch@aol.com), AOL

Edmund Jay (ejay@mgi1.com), MGI1

Michael B. Jones (mbj@microsoft.com), Microsoft

Chuck Mortimore (cmortimore@salesforce.com), Salesforce

Hideki Nara (hideki.nara@gmail.com), Takt Communications

Nat Sakimura (n-sakimura@nri.co.jp)), Nomura Research Institute, Ltd.

Ryo Itou (ritou@yahoo-corp.jp), Yahoo! Japan

|  |
| --- |
| [**TOC**](#toc) |

### Appendix B.  Document History

[[ To be removed from the final specification ]]

-14

* Changed section 3.2.1 to refer to access\_token ticket #134.
* Bumped version + date.
* Changed 7.4 in security considerations to show none is REQUIRED.
* Changed 3.2.4.1 User Info to UserInfo per Ticket #137.
* Changed formatting of 7.1 per ticket #140.

-13

* Changed check\_session to check\_id.
* schema=openid now required when requesting UserInfo.
* Removed issued\_to, since not well defined.
* Removed display values popup, touch, and mobile, since not well defined.

-12

* Ticket #48 Changed Check Session to take the id\_token as a parameter.

-11

* Renamed from "Lite" to "Basic Client".
* Numerous cleanups, including updating references.

-10

* Add back id\_token to the response type per issue 27.
* Changed endpoint name in example from id\_token to check\_session.
* Added token\_type to the response and explanations of the optional parameters.

-09

* Clean up typos.
* Clean up scope explanation.
* Fix 3.2.4.1 to include id\_token in response.

-08

* Added note about OP needing to read the full spec.
* Reverted back to GET for introspection based on Google feedback.
* Changed scopes to openid, profile, address, and email to make them additive.
* Changed introspection to Check Session Endpoint to be consistent with session management.
* Changed validation rules, the Check session endpoint will return an error for expired or invalid tokens, so the client doesn't need to check expiration.
* Added explanation of why an id\_token is used to verify identity rather than the user\_info access token.

-07

* Changed introspection to post
* Changed user\_info from ide to user\_ide to be consistent with introspection endpoint.
* Fixed introspection example to use id\_token rather than access token.
* Removed asking for id\_token in response type.
* Fixed Sec 3 to be clear it is client secret that is maintained between the client and the OP.

-06

* Only require the token flow in Lite. Removed code flow.
* Make id\_token required. The id\_token is treated as opaque.
* Rearranged sections for readability.
* Dropped the schema parameter to the Introspection endpoint, which was formerly a string with the value user\_id. This is unnecessary since the id\_token parameter already can be used to disambiguate the intended uses(s) of the endpoint.
* Dropped the requested audience from the Lite spec, which was formerly the identifier of the target audience of the response. This could be part of the Standard spec, but is an advanced scenario, and so not appropriate for Lite.
* Reference the Discovery and Registration specs, since they're needed for interaction between non-pre-configured parties (so that OpenID Connect installations can be Open).

-05

* Corrected issues raised by Casper Biering.
* Created the OpenID Connect Lite specification.

-04

* Correct issues raised by Pam Dingle and discussed on the mailing list after the 7-Jul-11 working group call.
* Adopted long\_names.

-03

* Correct issues raised by Johnny Bufu and discussed on the 7-Jul-11 working group call.

-02

* Consistency and cleanup pass, including removing unused references.

-01

* Initial draft

|  |
| --- |
| [**TOC**](#toc) |

### Authors' Addresses

|  |  |
| --- | --- |
|  | Nat Sakimura (editor) |
|  | Nomura Research Institute, Ltd. |
| **Email:** | [n-sakimura@nri.co.jp](mailto:n-sakimura@nri.co.jp) |
|  |  |
|  | John Bradley (editor) |
|  | Independent |
| **Email:** | [jbradley@mac.com](mailto:jbradley@mac.com) |
|  |  |
|  | Breno de Medeiros |
|  | Google |
| **Email:** | [breno@google.com](mailto:breno@google.com) |
|  |  |
|  | Michael B. Jones |
|  | Microsoft Corporation |
| **Email:** | [mbj@microsoft.com](mailto:mbj@microsoft.com) |
|  |  |
|  | Edmund Jay |
|  | MGI1 |
| **Email:** | [ejay@mgi1.com](mailto:ejay@mgi1.com) |
|  |  |
|  | Chuck Mortimore |
|  | Salesforce |
| **Email:** | [cmortimore@salesforce.com](mailto:cmortimore@salesforce.com) |