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

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Draft | N. Sakimura | |  | NRI | |  | J. Bradley | |  | Ping Identity | |  | M. Jones | |  | Microsoft | |  | E. Jay | |  | Illumila | |  | July 2, 2013 | |

# OpenID Connect Discovery 1.0 - draft 17

### Abstract

OpenID Connect 1.0 is a simple identity layer on top of the OAuth 2.0 protocol. It allows 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 provides a mechanism for the OpenID Connect Client to discover the End-User's OpenID Provider as well as the necessary endpoints used by the OpenID Connect protocol suite.

### Table of Contents

[**1.**](#Introduction)  Introduction  
    [**1.1.**](#rnc)  Requirements Notation and Conventions  
    [**1.2.**](#terminology)  Terminology  
[**2.**](#ProviderDisc)  OpenID Provider Discovery  
    [**2.1.**](#IdentifierNormalization)  Identifier Normalization  
        [**2.1.1.**](#IdentifierTypes)  User Input Identifier Types  
        [**2.1.2.**](#NormalizationSteps)  Normalization Steps  
    [**2.2.**](#Examples)  Non-Normative Examples  
        [**2.2.1.**](#EmailSyntax)  User Input Using E-Mail Address Syntax  
        [**2.2.2.**](#URLSyntax)  User Input Using URL Syntax  
        [**2.2.3.**](#host.port.example)  User Input Using Hostname and Port Syntax  
        [**2.2.4.**](#AcctURISyntax)  User Input Using "acct" URI Syntax  
[**3.**](#ProviderMetadata)  OpenID Provider Metadata  
[**4.**](#ProviderConfig)  Obtaining OpenID Provider Configuration Information  
    [**4.1.**](#ProviderConfigurationRequest)  OpenID Provider Configuration Request  
    [**4.2.**](#ProviderConfigurationResponse)  OpenID Provider Configuration Response  
    [**4.3.**](#ProviderConfigurationValidation)  OpenID Provider Configuration Validation  
[**5.**](#stringops)  String Operations  
[**6.**](#ImplementationConsiderations)  Implementation Considerations  
[**7.**](#Security)  Security Considerations  
    [**7.1.**](#TLS_requirements)  TLS Requirements  
[**8.**](#IANA)  IANA Considerations  
    [**8.1.**](#WellKnownRegistry)  Well-Known URI Registry  
        [**8.1.1.**](#WellKnownContents)  Registry Contents  
[**9.**](#rfc.references1)  References  
    [**9.1.**](#rfc.references1)  Normative References  
    [**9.2.**](#rfc.references2)  Informative References  
[**Appendix A.**](#Acknowledgements)  Acknowledgements  
[**Appendix B.**](#Notices)  Notices  
[**Appendix C.**](#History)  Document History  
[**§**](#rfc.authors)  Authors' Addresses

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

### 1.  Introduction

In order for an OpenID Client to utilize OpenID Connect services for an End-User, the Client needs to know where the OpenID Provider is. OpenID Connect uses [WebFinger (Jones, P., Salgueiro, G., and J. Smarr, “WebFinger,” May 2013.)](#I-D.ietf-appsawg-webfinger) [I‑D.ietf‑appsawg‑webfinger] to locate the OpenID Provider for an End-User.

Once an OpenID Provider is identified, the endpoint and other configuration information for that OP is retrieved from a well-known location as a JSON document.

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

### 1.1.  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 [RFC 2119 (Bradner, S., “Key words for use in RFCs to Indicate Requirement Levels,” March 1997.)](#RFC2119) [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.

All uses of [JSON Web Signature (JWS) (Jones, M., Bradley, J., and N. Sakimura, “JSON Web Signature (JWS),” May 2013.)](#JWS) [JWS] and [JSON Web Encryption (JWE) (Jones, M., Rescorla, E., and J. Hildebrand, “JSON Web Encryption (JWE),” May 2013.)](#JWE) [JWE] data structures in this specification utilize the JWS Compact Serialization or the JWE Compact Serialization; the JWS JSON Serialization and the JWE JSON Serialization are not used.

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

### 1.2.  Terminology

This specification uses the terms "Access Token", "Refresh Token", "Authorization Code", "Authorization Grant", "Authorization Server", "Authorization Endpoint", "Client", "Client Identifier", "Client Secret", "Protected Resource", "Resource Owner", "Resource Server", and "Token Endpoint" defined by [OAuth 2.0 (Hardt, D., “The OAuth 2.0 Authorization Framework,” October 2012.)](#RFC6749) [RFC6749], and the terms defined by [OpenID Connect Messages 1.0 (Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., Mortimore, C., and E. Jay, “OpenID Connect Messages 1.0,” June 2013.)](#OpenID.Messages) [OpenID.Messages].

This specification also defines the following terms:

Resource

Entity that is the target of a request in WebFinger.

Host

Server where a WebFinger service is hosted.

Identifier

Value that uniquely characterizes an Entity in a specific context.

Note: this document defines various kinds of Identifiers, designed for use in different contexts. Examples include URLs using the https scheme and e-mail addresses.

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

### 2.  OpenID Provider Discovery

OpenID Provider discovery is OPTIONAL; if a Relying Party knows the OP information through an out-of-band mechanism, they can skip this step and proceed to [Section 4 (Obtaining OpenID Provider Configuration Information)](#ProviderConfig).

Provider discovery requires the following information to make a discovery request:

resource

Identifier of the target End-User that is the subject of the discovery request.

host

Server where a WebFinger service is hosted.

rel

URI identifying the type of service whose location is requested.

OpenID Connect uses the following discoverable rel value in [WebFinger (Jones, P., Salgueiro, G., and J. Smarr, “WebFinger,” May 2013.)](#I-D.ietf-appsawg-webfinger) [I‑D.ietf‑appsawg‑webfinger]:

|  |  |
| --- | --- |
| **Rel Type** | **URI** |
| OpenID Connect Issuer | http://openid.net/specs/connect/1.0/issuer |

To start discovery of OpenID endpoints, the End-User supplies an Identifier to the Client or Relying Party. The Client applies the normalization rules to the Identifier to determine the Resource and Host. Then it makes an HTTPS GET request to the Host's [WebFinger (Jones, P., Salgueiro, G., and J. Smarr, “WebFinger,” May 2013.)](#I-D.ietf-appsawg-webfinger) [I‑D.ietf‑appsawg‑webfinger] endpoint with the resource and rel parameters to obtain the location of the requested service.

The Issuer MUST be returned in the response. This includes a URI scheme (which MUST be https), a Host, and OPTIONALLY, a port.

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

### 2.1.  Identifier Normalization

The purpose of normalization is to determine a normalized Resource and Host from the user input Identifier. This is then used as input to WebFinger to discover the Issuer.

The user input Identifier SHOULD be a URL or URI relative reference defined in [RFC 3986 (Berners-Lee, T., Fielding, R., and L. Masinter, “Uniform Resource Identifier (URI): Generic Syntax,” January 2005.)](#RFC3986) [RFC3986]. The user input Identifier MUST include the authority component.

Note: A URI relative reference includes a string that looks like an e-mail address in the form of userinfo@host. This is a valid authority component of a URI but excludes various possible extra strings allowed in addr-spec syntax of [RFC 5322 (Resnick, P., Ed., “Internet Message Format,” October 2008.)](#RFC5322) [RFC5322].

The Identifier normalization rules MAY be extended by additional specifications to enable other identifier types such as telephone numbers or [XRIs (Reed, D. and D. McAlpin, “Extensible Resource Identifier (XRI) Syntax V2.0,” November 2005.)](#XRI_Syntax_2.0) [XRI\_Syntax\_2.0] to also be used.

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

### 2.1.1.  User Input Identifier Types

A user input Identifier can be categorized into the following types, which require different normalization processes:

1. User input Identifiers starting with the [XRI (Reed, D. and D. McAlpin, “Extensible Resource Identifier (XRI) Syntax V2.0,” November 2005.)](" \l "XRI_Syntax_2.0) [XRI\_Syntax\_2.0] global context symbols ('=','@', and '!') are RESERVED. Processing of these identifiers is out of scope for this specification.
2. All other user input Identifiers MUST be treated as a URI in one of the forms scheme "://" authority path-abempty [ "?" query ] [ "#" fragment ] or authority path-abempty [ "?" query ] [ "#" fragment ] or scheme ":" path-rootless per [RFC 3986 (Berners-Lee, T., Fielding, R., and L. Masinter, “Uniform Resource Identifier (URI): Generic Syntax,” January 2005.)](" \l "RFC3986) [RFC3986].

Note: The user input Identifier MAY be in the form of userinfo@host. For the End-User, this would normally be perceived as being an e-mail address. However, it is also a valid userpart "@" host section of an acct URI [[I‑D.ietf‑appsawg‑acct‑uri] (Saint-Andre, P., “The 'acct' URI Scheme,” June 2013.)](" \l "I-D.ietf-appsawg-acct-uri), and this specification treats it such as to exclude various extra strings allowed in addr-spec of [RFC 5322 (Resnick, P., Ed., “Internet Message Format,” October 2008.)](#RFC5322) [RFC5322].

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

### 2.1.2.  Normalization Steps

A string of any other type is interpreted as a URI in one of the forms scheme "://" authority path-abempty [ "?" query ] [ "#" fragment ] or authority path-abempty [ "?" query ] [ "#" fragment ] or scheme ":" path-rootless per [RFC 3986 (Berners-Lee, T., Fielding, R., and L. Masinter, “Uniform Resource Identifier (URI): Generic Syntax,” January 2005.)](#RFC3986) [RFC3986] and is normalized according to the following rules:

1. If the user input Identifier does not have an [RFC 3986 (Berners-Lee, T., Fielding, R., and L. Masinter, “Uniform Resource Identifier (URI): Generic Syntax,” January 2005.)](" \l "RFC3986) [RFC3986] scheme portion, the string is interpreted as [userinfo "@"] host [":" port] path-abempty [ "?" query ] [ "#" fragment ] per [RFC 3986 (Berners-Lee, T., Fielding, R., and L. Masinter, “Uniform Resource Identifier (URI): Generic Syntax,” January 2005.)](" \l "RFC3986) [RFC3986].
2. If the userinfo component is present and all of the scheme component, path component, query component, and port component are empty, the acct scheme is assumed. In this case, the normalized URI is formed by prefixing acct: to the string as the scheme. Per the ['acct' URI Scheme (Saint-Andre, P., “The 'acct' URI Scheme,” June 2013.)](" \l "I-D.ietf-appsawg-acct-uri) [I‑D.ietf‑appsawg‑acct‑uri], if there is an at-sign character ('@') in the userinfo component, it needs to be percent-encoded as described in [RFC 3986 (Berners-Lee, T., Fielding, R., and L. Masinter, “Uniform Resource Identifier (URI): Generic Syntax,” January 2005.)](" \l "RFC3986) [RFC3986].
3. For all other inputs without a scheme portion, the https scheme is assumed, and the normalized URI is formed by prefixing https:// to the string as the scheme.
4. When the input contains an explicit scheme such as acct that matches the RFC 3986 scheme ":" path-rootless syntax, no input normalization is performed.
5. If the resulting URI contains a fragment portion, it MUST be stripped off together with the fragment delimiter character "#".

The [WebFinger (Jones, P., Salgueiro, G., and J. Smarr, “WebFinger,” May 2013.)](#I-D.ietf-appsawg-webfinger) [I‑D.ietf‑appsawg‑webfinger] Resource in this case is the resulting URI, and the WebFinger Host is the authority component.

Note: Since the definition of authority in [RFC 3986 (Berners-Lee, T., Fielding, R., and L. Masinter, “Uniform Resource Identifier (URI): Generic Syntax,” January 2005.)](#RFC3986) [RFC3986] is [ userinfo "@" ] host [ ":" port ], it is legal to have a user input identifier like userinfo@host:port, e.g., alice@example.com:8080.

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

### 2.2.  Non-Normative Examples

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

### 2.2.1.  User Input Using E-Mail Address Syntax

To find the Issuer for the given user input in the form of an e-mail address joe@example.com, the WebFinger parameters are as follows:

|  |  |
| --- | --- |
| **WebFinger Parameter** | **Value** |
| resource | acct:joe@example.com |
| host | example.com |
| rel | http://openid.net/specs/connect/1.0/issuer |

Note that in this case, the acct: scheme [[I‑D.ietf‑appsawg‑acct‑uri] (Saint-Andre, P., “The 'acct' URI Scheme,” June 2013.)](#I-D.ietf-appsawg-acct-uri) is prepended to the Identifier.

Following the WebFinger specification, the Client would make the following request to get the discovery information (with line wraps within lines for display purposes only):

GET /.well-known/webfinger

?resource=acct%3Ajoe%40example.com

&rel=http%3A%2F%2Fopenid.net%2Fspecs%2Fconnect%2F1.0%2Fissuer

HTTP/1.1

Host: example.com

HTTP/1.1 200 OK

Content-Type: application/jrd+json

{

"subject": "acct:joe@example.com",

"links":

[

{

"rel": "http://openid.net/specs/connect/1.0/issuer",

"href": "https://server.example.com"

}

]

}

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

### 2.2.2.  User Input Using URL Syntax

To find the Issuer for the given URL, https://example.com/joe, the WebFinger parameters are as follows:

|  |  |
| --- | --- |
| **WebFinger Parameter** | **Value** |
| resource | https://example.com/joe |
| host | example.com |
| rel | http://openid.net/specs/connect/1.0/issuer |

Following the WebFinger specification, the Client would make the following request to get the discovery information (with line wraps within lines for display purposes only):

GET /.well-known/webfinger

?resource=https%3A%2F%2Fexample.com%2Fjoe

&rel=http%3A%2F%2Fopenid.net%2Fspecs%2Fconnect%2F1.0%2Fissuer

HTTP/1.1

Host: example.com

HTTP/1.1 200 OK

Content-Type: application/jrd+json

{

"subject": "https://example.com/joe",

"links":

[

{

"rel": "http://openid.net/specs/connect/1.0/issuer",

"href": "https://server.example.com"

}

]

}

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

### 2.2.3.  User Input Using Hostname and Port Syntax

If the user input is in the form of host:port, e.g., example.com:8080, then it is assumed as the authority component of the URL.

To find the Issuer for the given hostname, example.com:8080, the WebFinger parameters are as follows:

|  |  |
| --- | --- |
| **WebFinger Parameter** | **Value** |
| resource | https://example.com:8080/ |
| host | example.com:8080 |
| rel | http://openid.net/specs/connect/1.0/issuer |

Following the WebFinger specification, the Client would make the following request to get the discovery information (with line wraps within lines for display purposes only):

GET /.well-known/webfinger

?resource=https%3A%2F%2Fexample.com%3A8080%2F

&rel=http%3A%2F%2Fopenid.net%2Fspecs%2Fconnect%2F1.0%2Fissuer

HTTP/1.1

Host: example.com:8080

HTTP/1.1 200 OK

Content-Type: application/jrd+json

{

"subject": "https://example.com:8080/",

"links":

[

{

"rel": "http://openid.net/specs/connect/1.0/issuer",

"href": "https://server.example.com"

}

]

}

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

### 2.2.4.  User Input Using "acct" URI Syntax

To find the Issuer for the given user input in the form of an account URI acct:juliet%40capulet.example@shoppingsite.example.com, the WebFinger parameters are as follows:

|  |  |
| --- | --- |
| **WebFinger Parameter** | **Value** |
| resource | acct:juliet%40capulet.example@shoppingsite.example.com |
| host | shoppingsite.example.com |
| rel | http://openid.net/specs/connect/1.0/issuer |

Following the WebFinger specification, the Client would make the following request to get the discovery information (with line wraps within lines for display purposes only):

GET /.well-known/webfinger

?resource=acct%3Ajuliet%2540capulet.example%40shoppingsite.example.com

&rel=http%3A%2F%2Fopenid.net%2Fspecs%2Fconnect%2F1.0%2Fissuer

HTTP/1.1

Host: shoppingsite.example.com

HTTP/1.1 200 OK

Content-Type: application/jrd+json

{

"subject": "acct:juliet%40capulet.example@shoppingsite.example.com",

"links":

[

{

"rel": "http://openid.net/specs/connect/1.0/issuer",

"href": "https://server.example.com"

}

]

}

Note: It is common for sites to use e-mail addresses as local identifiers for accounts at those sites, even though the domain in the e-mail address one controlled by the site. For instance, the site example.org might have a local account named joe@example.com. As of the time of this writing, a discussion is ongoing among WebFinger contributors about the syntax that should be used when discovering information about such accounts with WebFinger. The current thinking seems to be that such accounts would be represented by quoting the '@' character in the userinfo portion of the account identifier when constructing the acct: URI representing the account. Such an example is acct:joe%40example.com@example.org. In a future version of this specification, it is possible that normalization rules will be defined allowing End-Users to input values like joe@example.com@example.org to initiate discovery on such accounts.

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

### 3.  OpenID Provider Metadata

OpenID Providers have metadata describing their configuration. The OpenID Provider Metadata values used by OpenID Connect are:

issuer

REQUIRED. URL using the https scheme with no query or fragment component that the OP asserts as its Issuer Identifier.

authorization\_endpoint

OPTIONAL. URL of the OP's Authentication and Authorization Endpoint [[OpenID.Messages] (Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., Mortimore, C., and E. Jay, “OpenID Connect Messages 1.0,” June 2013.)](#OpenID.Messages).

token\_endpoint

OPTIONAL. URL of the OP's OAuth 2.0 Token Endpoint [[OpenID.Messages] (Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., Mortimore, C., and E. Jay, “OpenID Connect Messages 1.0,” June 2013.)](#OpenID.Messages).

userinfo\_endpoint

RECOMMENDED. URL of the OP's UserInfo Endpoint [[OpenID.Messages] (Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., Mortimore, C., and E. Jay, “OpenID Connect Messages 1.0,” June 2013.)](#OpenID.Messages). This URL MUST use the https scheme and MAY contain port, path, and query parameter components.

check\_session\_iframe

OPTIONAL. URL of an OP endpoint that provides a page to support cross-origin communications for session state information with the RP Client, using the HTML5 postMessage API. The page is loaded from an invisible iframe embedded in an RP page so that it can run in the OP's security context. See [[OpenID.Session] (Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., and N. Agarwal, “OpenID Connect Session Management 1.0,” June 2013.)](#OpenID.Session).

end\_session\_endpoint

OPTIONAL. URL of the OP's endpoint that initiates logging out the End-User. See [[OpenID.Session] (Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., and N. Agarwal, “OpenID Connect Session Management 1.0,” June 2013.)](#OpenID.Session).

jwks\_uri

REQUIRED. URL of the OP's JSON Web Key Set [[JWK] (Jones, M., “JSON Web Key (JWK),” May 2013.)](#JWK) document. This contains the signing key(s) the Client uses to validate signatures from the OP. The JWK Set MAY also contain the Server's encryption key(s), which are used by Clients to encrypt requests to the Server. When both signing and encryption keys are made available, a use (Key Use) parameter value is REQUIRED for all keys in the document to indicate each key's intended usage.

registration\_endpoint

RECOMMENDED. URL of the OP's Dynamic Client Registration Endpoint [[OpenID.Registration] (Sakimura, N., Bradley, J., and M. Jones, “OpenID Connect Dynamic Client Registration 1.0,” June 2013.)](#OpenID.Registration).

scopes\_supported

RECOMMENDED. JSON array containing a list of the [OAuth 2.0 (Hardt, D., “The OAuth 2.0 Authorization Framework,” October 2012.)](#RFC6749) [RFC6749] scope values that this server supports. The server MUST support the openid scope value.

response\_types\_supported

REQUIRED. JSON array containing a list of the OAuth 2.0 response\_type values that this server supports. The server MUST support the code, id\_token, and the token id\_token response type values.

grant\_types\_supported

OPTIONAL. JSON array containing a list of the OAuth 2.0 grant type values that this server supports. The server MUST support the authorization\_code and implicit grant type values and MAY support the urn:ietf:params:oauth:grant-type:jwt-bearer grant type defined in [OAuth JWT Bearer Token Profiles (Jones, M., Campbell, B., and C. Mortimore, “JSON Web Token (JWT) Bearer Token Profiles for OAuth 2.0,” March 2013.)](#OAuth.JWT) [OAuth.JWT]. If omitted, the default value is ["authorization\_code", "implicit"].

acr\_values\_supported

OPTIONAL. JSON array containing a list of the Authentication Context Class References that this server supports.

subject\_types\_supported

REQUIRED. JSON array containing a list of the subject identifier types that this server supports. Valid types include pairwise and public.

userinfo\_signing\_alg\_values\_supported

OPTIONAL. JSON array containing a list of the JWS [[JWS] (Jones, M., Bradley, J., and N. Sakimura, “JSON Web Signature (JWS),” May 2013.)](#JWS) signing algorithms (alg values) [[JWA] (Jones, M., “JSON Web Algorithms (JWA),” May 2013.)](#JWA) supported by the UserInfo Endpoint to encode the Claims in a JWT [[JWT] (Jones, M., Bradley, J., and N. Sakimura, “JSON Web Token (JWT),” May 2013.)](#JWT).

userinfo\_encryption\_alg\_values\_supported

OPTIONAL. JSON array containing a list of the JWE [[JWE] (Jones, M., Rescorla, E., and J. Hildebrand, “JSON Web Encryption (JWE),” May 2013.)](#JWE) encryption algorithms (alg values) [[JWA] (Jones, M., “JSON Web Algorithms (JWA),” May 2013.)](#JWA) supported by the UserInfo Endpoint to encode the Claims in a JWT [[JWT] (Jones, M., Bradley, J., and N. Sakimura, “JSON Web Token (JWT),” May 2013.)](#JWT).

userinfo\_encryption\_enc\_values\_supported

OPTIONAL. JSON array containing a list of the JWE encryption algorithms (enc values) [[JWA] (Jones, M., “JSON Web Algorithms (JWA),” May 2013.)](#JWA) supported by the UserInfo Endpoint to encode the Claims in a JWT [[JWT] (Jones, M., Bradley, J., and N. Sakimura, “JSON Web Token (JWT),” May 2013.)](#JWT).

id\_token\_signing\_alg\_values\_supported

REQUIRED. JSON array containing a list of the JWS signing algorithms (alg values) supported by the Authorization Server for the ID Token to encode the Claims in a JWT [[JWT] (Jones, M., Bradley, J., and N. Sakimura, “JSON Web Token (JWT),” May 2013.)](#JWT).

id\_token\_encryption\_alg\_values\_supported

OPTIONAL. JSON array containing a list of the JWE encryption algorithms (alg values) supported by the Authorization Server for the ID Token to encode the Claims in a JWT [[JWT] (Jones, M., Bradley, J., and N. Sakimura, “JSON Web Token (JWT),” May 2013.)](#JWT).

id\_token\_encryption\_enc\_values\_supported

OPTIONAL. JSON array containing a list of the JWE encryption algorithms (enc values) supported by the Authorization Server for the ID Token to encode the Claims in a JWT [[JWT] (Jones, M., Bradley, J., and N. Sakimura, “JSON Web Token (JWT),” May 2013.)](#JWT).

request\_object\_signing\_alg\_values\_supported

OPTIONAL. JSON array containing a list of the JWS signing algorithms (alg values) supported by the Authorization Server for the Request Object described in Section 2.9 of [OpenID Connect Messages 1.0 (Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., Mortimore, C., and E. Jay, “OpenID Connect Messages 1.0,” June 2013.)](#OpenID.Messages) [OpenID.Messages]. These algorithms are used both when the Request Object is passed by value (using the request parameter) and when it is passed by reference (using the request\_uri parameter). Servers SHOULD support none and RS256.

request\_object\_encryption\_alg\_values\_supported

OPTIONAL. JSON array containing a list of the JWE encryption algorithms (alg values) supported by the Authorization Server for the Request Object described in Section 2.9 of [OpenID Connect Messages 1.0 (Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., Mortimore, C., and E. Jay, “OpenID Connect Messages 1.0,” June 2013.)](#OpenID.Messages) [OpenID.Messages]. These algorithms are used both when the Request Object is passed by value and when it is passed by reference.

request\_object\_encryption\_enc\_values\_supported

OPTIONAL. JSON array containing a list of the JWE encryption algorithms (enc values) supported by the Authorization Server for the Request Object described in Section 2.9 of [OpenID Connect Messages 1.0 (Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., Mortimore, C., and E. Jay, “OpenID Connect Messages 1.0,” June 2013.)](#OpenID.Messages) [OpenID.Messages]. These algorithms are used both when the Request Object is passed by value and when it is passed by reference.

token\_endpoint\_auth\_methods\_supported

OPTIONAL. JSON array containing a list of authentication methods supported by this Token Endpoint. The options are client\_secret\_post, client\_secret\_basic, client\_secret\_jwt, and private\_key\_jwt, as described in Section 2.2.1 of [OpenID Connect Messages 1.0 (Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., Mortimore, C., and E. Jay, “OpenID Connect Messages 1.0,” June 2013.)](#OpenID.Messages) [OpenID.Messages]. Other authentication methods MAY be defined by extensions. If omitted, the default is client\_secret\_basic -- the HTTP Basic Authentication Scheme as specified in Section 2.3.1 of [OAuth 2.0 (Hardt, D., “The OAuth 2.0 Authorization Framework,” October 2012.)](#RFC6749) [RFC6749].

token\_endpoint\_auth\_signing\_alg\_values\_supported

OPTIONAL. JSON array containing a list of the JWS signing algorithms (alg values) supported by the Token Endpoint for the private\_key\_jwt and client\_secret\_jwt methods to encode the JWT [[JWT] (Jones, M., Bradley, J., and N. Sakimura, “JSON Web Token (JWT),” May 2013.)](#JWT). Servers SHOULD support RS256.

display\_values\_supported

OPTIONAL. JSON array containing a list of the display parameter values that the OpenID Provider supports. These values are described in Section 2.1.1 of [OpenID Connect Messages 1.0 (Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., Mortimore, C., and E. Jay, “OpenID Connect Messages 1.0,” June 2013.)](#OpenID.Messages) [OpenID.Messages].

claim\_types\_supported

OPTIONAL. JSON array containing a list of the Claim Types that the OpenID Provider supports. These Claim Types are described in Section 2.6 of [OpenID Connect Messages 1.0 (Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., Mortimore, C., and E. Jay, “OpenID Connect Messages 1.0,” June 2013.)](#OpenID.Messages) [OpenID.Messages]. Values defined by this specification are normal, aggregated, and distributed. If not specified, the implementation supports only normal Claims.

claims\_supported

RECOMMENDED. JSON array containing a list of the Claim Names of the Claims that the OpenID Provider MAY be able to supply values for. Note that for privacy or other reasons, this might not be an exhaustive list.

service\_documentation

OPTIONAL. URL of a page containing human-readable information that developers might want or need to know when using the OpenID Provider. In particular, if the OpenID Provider does not support Dynamic Client Registration, then information on how to register Clients needs to be provided in this documentation.

claims\_locales\_supported

OPTIONAL. Languages and scripts supported for values in Claims being returned, represented as a JSON array of [BCP47 (Phillips, A. and M. Davis, “Tags for Identifying Languages,” September 2009.)](#RFC5646) [RFC5646] language tag values. Not all languages and scripts are necessarily supported for all Claim values.

ui\_locales\_supported

OPTIONAL. Languages and scripts supported for the user interface, represented as a JSON array of [BCP47 (Phillips, A. and M. Davis, “Tags for Identifying Languages,” September 2009.)](#RFC5646) [RFC5646] language tag values.

claims\_parameter\_supported

OPTIONAL. Boolean value specifying whether the OP supports use of the claims parameter, with true indicating support. If omitted, the default value is false.

request\_parameter\_supported

OPTIONAL. Boolean value specifying whether the OP supports use of the request parameter, with true indicating support. If omitted, the default value is false.

request\_uri\_parameter\_supported

OPTIONAL. Boolean value specifying whether the OP supports use of the request\_uri parameter, with true indicating support. If omitted, the default value is true.

require\_request\_uri\_registration

OPTIONAL. Boolean value specifying whether the OP requires any request\_uri values used to be pre-registered using the request\_uris registration parameter. Pre-registration is REQUIRED when the value is true. If omitted, the default value is false.

op\_policy\_uri

OPTIONAL. URL that the OpenID Provider provides to the person registering the Client to read about the OP's requirements on how the Relying Party can use the data provided by the OP. The registration process SHOULD display this URL to the person registering the Client if it is given.

op\_tos\_uri

OPTIONAL. URL that the OpenID Provider provides to the person registering the Client to read about OpenID Provider's terms of service. The registration process SHOULD display this URL to the person registering the Client if it is given.

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

### 4.  Obtaining OpenID Provider Configuration Information

This step is OPTIONAL. The OpenID Provider endpoints and configuration information MAY be obtained out-of-band.

Using the Issuer discovered in [Section 2 (OpenID Provider Discovery)](#ProviderDisc) or through direct configuration, the OpenID Provider's configuration can be retrieved.

OpenID Providers MUST make a JSON document available at the path formed by concatenating the string /.well-known/openid-configuration to the Issuer. The syntax and semantics of .well-known are defined in [RFC 5785 (Nottingham, M. and E. Hammer-Lahav, “Defining Well-Known Uniform Resource Identifiers (URIs),” April 2010.)](#RFC5785) [RFC5785] and apply to the Issuer value when it contains no path component. openid-configuration MUST point to a JSON document compliant with this specification and MUST be returned using the application/json content type.

OpenID Providers supporting discovery MUST support receiving WebFinger requests via TLS. See [Section 7.1 (TLS Requirements)](#TLS_requirements) for more information on using TLS.

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

### 4.1.  OpenID Provider Configuration Request

An OpenID Provider Configuration Document MUST be queried using an HTTPS GET request at the previously specified path.

The Client would make the following request to the Issuer to get the Configuration information, if the Issuer contains no path component.

GET /.well-known/openid-configuration HTTP/1.1

Host: example.com

If the Issuer value contains a path component, any terminating / MUST be removed before appending /.well-known/openid-configuration. The Client would make the following request to the Issuer to get the Configuration information, if the Issuer string were https://example.com/issuer1

GET /issuer1/.well-known/openid-configuration HTTP/1.1

Host: example.com

Path components are allowed to support multiple issuers per host. This is required in some multi-tenant hosting configurations. This use of .well-known is for supporting multiple issuers per host, and unlike its use in [RFC 5785 (Nottingham, M. and E. Hammer-Lahav, “Defining Well-Known Uniform Resource Identifiers (URIs),” April 2010.)](#RFC5785) [RFC5785], it does not provide general information about the host.

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

### 4.2.  OpenID Provider Configuration Response

The response is a set of Claims about the OpenID Provider's configuration, including all necessary endpoints, supported scopes, and public key location information. The response MUST return the 200 OK response code and a plain text JSON object using the application/json content type that contains a set of Claims as its members that are a subset of the Metadata values defined in [Section 3 (OpenID Provider Metadata)](#ProviderMetadata). Other Claims MAY also be returned.

Claims that return multiple values are represented as JSON arrays. Claims with zero elements MUST be omitted from the response.

The following is a non-normative example response:

HTTP/1.1 200 OK

Content-Type: application/json

{

"issuer":

"https://server.example.com",

"authorization\_endpoint":

"https://server.example.com/connect/authorize",

"token\_endpoint":

"https://server.example.com/connect/token",

"token\_endpoint\_auth\_methods\_supported":

["client\_secret\_basic", "private\_key\_jwt"],

"token\_endpoint\_auth\_signing\_alg\_values\_supported":

["RS256", "ES256"],

"userinfo\_endpoint":

"https://server.example.com/connect/userinfo",

"check\_session\_iframe":

"https://server.example.com/connect/check\_session",

"end\_session\_endpoint":

"https://server.example.com/connect/end\_session",

"jwks\_uri":

"https://server.example.com/jwks.json",

"registration\_endpoint":

"https://server.example.com/connect/register",

"scopes\_supported":

["openid", "profile", "email", "address",

"phone", "offline\_access"],

"response\_types\_supported":

["code", "code id\_token", "id\_token", "token id\_token"],

"acr\_values\_supported":

["urn:mace:incommon:iap:silver",

"urn:mace:incommon:iap:bronze"],

"subject\_types\_supported":

["public", "pairwise"],

"userinfo\_signing\_alg\_values\_supported":

["RS256", "ES256", "HS256"],

"userinfo\_encryption\_alg\_values\_supported":

["RSA1\_5", "A128KW"],

"userinfo\_encryption\_enc\_values\_supported":

["A128CBC-HS256", "A128GCM"],

"id\_token\_signing\_alg\_values\_supported":

["RS256", "ES256", "HS256"],

"id\_token\_encryption\_alg\_values\_supported":

["RSA1\_5", "A128KW"],

"id\_token\_encryption\_enc\_values\_supported":

["A128CBC-HS256", "A128GCM"],

"request\_object\_signing\_alg\_values\_supported":

["none", "RS256", "ES256"],

"display\_values\_supported":

["page", "popup"],

"claim\_types\_supported":

["normal", "distributed"],

"claims\_supported":

["sub", "iss", "auth\_time", "acr",

"name", "given\_name", "family\_name", "nickname",

"profile", "picture", "website",

"email", "email\_verified", "locale", "zoneinfo",

"http://example.info/claims/groups"],

"claims\_parameter\_supported":

true,

"service\_documentation":

"http://server.example.com/connect/service\_documentation.html",

"ui\_locales\_supported":

["en-US", "en-GB", "en-CA", "fr-FR", "fr-CA"]

}

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

### 4.3.  OpenID Provider Configuration Validation

If any of the validation procedures defined in this specification fail, any operations requiring the information that failed to correctly validate MUST be aborted and the information that failed to validate MUST NOT be used.

If the configuration response contains the Issuer element, the value MUST exactly match the Issuer for the URL that was directly used to retrieve the configuration. Since the discovery process allows for multiple levels of redirection, this Issuer URL MAY be different from the one originally used to begin the discovery process.

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

### 5.  String Operations

Processing some OpenID Connect messages requires comparing values in the messages to known values. For example, the member names in the provider configuration response might be compared to specific member names such as issuer. Comparing Unicode strings, however, has significant security implications.

Therefore, comparisons between JSON strings and other Unicode strings MUST be performed as specified below:

1. Remove any JSON applied escaping to produce an array of Unicode code points.
2. [Unicode Normalization (Davis, M., Whistler, K., and M. Dürst, “Unicode Normalization Forms,” 09 2009.)](" \l "USA15) [USA15] MUST NOT be applied at any point to either the JSON string or to the string it is to be compared against.
3. Comparisons between the two strings MUST be performed as a Unicode code point to code point equality comparison.

In several places, this specification uses space delimited lists of strings. In all such cases, only the ASCII space character (0x20) MAY be used for this purpose.

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

### 6.  Implementation Considerations

This specification defines features used by both Relying Parties and OpenID Providers that choose to implement Discovery. 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 Discovery are defined by this specification.

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

### 7.  Security Considerations

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

### 7.1.  TLS Requirements

Implementations MUST support TLS. Which version(s) ought to be implemented will vary over time, and depend on the widespread deployment and known security vulnerabilities at the time of implementation. At the time of this writing, TLS version 1.2 [[RFC5246] (Dierks, T. and E. Rescorla, “The Transport Layer Security (TLS) Protocol Version 1.2,” August 2008.)](#RFC5246) is the most recent version, but has very limited actual deployment, and might not be readily available in implementation toolkits. TLS version 1.0 [[RFC2246] (Dierks, T. and C. Allen, “The TLS Protocol Version 1.0,” January 1999.)](#RFC2246) is the most widely deployed version, and will give the broadest interoperability.

To protect against information disclosure and tampering, confidentiality protection MUST be applied using TLS with a ciphersuite that provides confidentiality and integrity protection.

Whenever TLS is used, a TLS server certificate check MUST be performed, per [RFC 6125 (Saint-Andre, P. and J. Hodges, “Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS),” March 2011.)](#RFC6125) [RFC6125].

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

### 8.  IANA Considerations

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

### 8.1.  Well-Known URI Registry

This specification registers the well-known URI defined in [Section 4 (Obtaining OpenID Provider Configuration Information)](#ProviderConfig) in the IANA Well-Known URI registry defined in [RFC 5785 (Nottingham, M. and E. Hammer-Lahav, “Defining Well-Known Uniform Resource Identifiers (URIs),” April 2010.)](#RFC5785) [RFC5785].

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

### 8.1.1.  Registry Contents

* URI suffix: openid-configuration
* Change controller: OpenID Foundation Artifact Binding Working Group - openid-specs-ab@lists.openid.net
* Specification document: [Section 4 (Obtaining OpenID Provider Configuration Information)](" \l "ProviderConfig) of this document
* Related information: (none)

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

### 9.  References

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

### 9.1. Normative References

|  |  |
| --- | --- |
| **[I-D.ietf-appsawg-acct-uri]** | Saint-Andre, P., “[The 'acct' URI Scheme](http://tools.ietf.org/html/draft-ietf-appsawg-acct-uri-05),” draft-ietf-appsawg-acct-uri-05 (work in progress), June 2013 ([TXT](http://www.ietf.org/internet-drafts/draft-ietf-appsawg-acct-uri-05.txt)). |
| **[I-D.ietf-appsawg-webfinger]** | Jones, P., Salgueiro, G., and J. Smarr, “[WebFinger](http://tools.ietf.org/html/draft-ietf-appsawg-webfinger-14),” draft-ietf-appsawg-webfinger-14 (work in progress), May 2013 ([TXT](http://www.ietf.org/internet-drafts/draft-ietf-appsawg-webfinger-14.txt)). |
| **[JWA]** | Jones, M., “[JSON Web Algorithms (JWA)](http://tools.ietf.org/html/draft-ietf-jose-json-web-algorithms),” draft-ietf-jose-json-web-algorithms (work in progress), May 2013 ([HTML](http://tools.ietf.org/html/draft-ietf-jose-json-web-algorithms-11)). |
| **[JWE]** | Jones, M., Rescorla, E., and J. Hildebrand, “[JSON Web Encryption (JWE)](http://tools.ietf.org/html/draft-ietf-jose-json-web-encryption),” draft-ietf-jose-json-web-encryption (work in progress), May 2013 ([HTML](http://tools.ietf.org/html/draft-ietf-jose-json-web-encryption-11)). |
| **[JWK]** | Jones, M., “[JSON Web Key (JWK)](http://tools.ietf.org/html/draft-ietf-jose-json-web-key),” draft-ietf-jose-json-web-key (work in progress), May 2013 ([HTML](http://tools.ietf.org/html/draft-ietf-jose-json-web-key-11)). |
| **[JWS]** | Jones, M., Bradley, J., and N. Sakimura, “[JSON Web Signature (JWS)](http://tools.ietf.org/html/draft-ietf-jose-json-web-signature),” draft-ietf-jose-json-web-signature (work in progress), May 2013 ([HTML](http://tools.ietf.org/html/draft-ietf-jose-json-web-signature-11)). |
| **[JWT]** | Jones, M., Bradley, J., and N. Sakimura, “[JSON Web Token (JWT)](http://tools.ietf.org/html/draft-ietf-oauth-json-web-token),” draft-ietf-oauth-json-web-token (work in progress), May 2013 ([HTML](http://tools.ietf.org/html/draft-ietf-oauth-json-web-token-08)). |
| **[OAuth.JWT]** | Jones, M., Campbell, B., and C. Mortimore, “[JSON Web Token (JWT) Bearer Token Profiles for OAuth 2.0](http://tools.ietf.org/html/draft-ietf-oauth-jwt-bearer),” draft-ietf-oauth-jwt-bearer (work in progress), March 2013 ([HTML](http://tools.ietf.org/html/draft-ietf-oauth-jwt-bearer-05)). |
| **[OpenID.Messages]** | Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., Mortimore, C., and E. Jay, “[OpenID Connect Messages 1.0](http://openid.net/specs/openid-connect-messages-1_0-20.html),” June 2013. |
| **[OpenID.Registration]** | Sakimura, N., Bradley, J., and M. Jones, “[OpenID Connect Dynamic Client Registration 1.0](http://openid.net/specs/openid-connect-registration-1_0-19.html),” June 2013. |
| **[OpenID.Session]** | Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., and N. Agarwal, “[OpenID Connect Session Management 1.0](http://openid.net/specs/openid-connect-session-1_0-15.html),” June 2013. |
| **[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)). |
| **[RFC2246]** | [Dierks, T.](mailto:tdierks@certicom.com) and [C. Allen](mailto:callen@certicom.com), “[The TLS Protocol Version 1.0](http://tools.ietf.org/html/rfc2246),” RFC 2246, January 1999 ([TXT](http://www.rfc-editor.org/rfc/rfc2246.txt)). |
| **[RFC3986]** | [Berners-Lee, T.](mailto:timbl@w3.org), [Fielding, R.](mailto:fielding@gbiv.com), and [L. Masinter](mailto:LMM@acm.org), “[Uniform Resource Identifier (URI): Generic Syntax](http://tools.ietf.org/html/rfc3986),” STD 66, RFC 3986, January 2005 ([TXT](http://www.rfc-editor.org/rfc/rfc3986.txt), [HTML](http://xml.resource.org/public/rfc/html/rfc3986.html), [XML](http://xml.resource.org/public/rfc/xml/rfc3986.xml)). |
| **[RFC5246]** | Dierks, T. and E. Rescorla, “[The Transport Layer Security (TLS) Protocol Version 1.2](http://tools.ietf.org/html/rfc5246),” RFC 5246, August 2008 ([TXT](http://www.rfc-editor.org/rfc/rfc5246.txt)). |
| **[RFC5322]** | [Resnick, P., Ed.](mailto:presnick@qualcomm.com), “[Internet Message Format](http://tools.ietf.org/html/rfc5322),” RFC 5322, October 2008 ([TXT](http://www.rfc-editor.org/rfc/rfc5322.txt), [HTML](http://xml.resource.org/public/rfc/html/rfc5322.html), [XML](http://xml.resource.org/public/rfc/xml/rfc5322.xml)). |
| **[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)). |
| **[RFC5785]** | Nottingham, M. and E. Hammer-Lahav, “[Defining Well-Known Uniform Resource Identifiers (URIs)](http://tools.ietf.org/html/rfc5785),” RFC 5785, April 2010 ([TXT](http://www.rfc-editor.org/rfc/rfc5785.txt)). |
| **[RFC6125]** | Saint-Andre, P. and J. Hodges, “[Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)](http://tools.ietf.org/html/rfc6125),” RFC 6125, March 2011 ([TXT](http://www.rfc-editor.org/rfc/rfc6125.txt)). |
| **[RFC6749]** | Hardt, D., “[The OAuth 2.0 Authorization Framework](http://tools.ietf.org/html/rfc6749),” RFC 6749, October 2012 ([TXT](http://www.rfc-editor.org/rfc/rfc6749.txt)). |
| **[USA15]** | [Davis, M.](mailto:markdavis@google.com), [Whistler, K.](mailto:ken@unicode.org), and M. Dürst, “Unicode Normalization Forms,” Unicode Standard Annex 15, 09 2009. |

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

### 9.2. Informative References

|  |  |
| --- | --- |
| **[XRI\_Syntax\_2.0]** | Reed, D. and D. McAlpin, “Extensible Resource Identifier (XRI) Syntax V2.0,” November 2005 ([HTML](http://www.oasis-open.org/committees/download.php/15376/xri-syntax-V2.0-cs.html), [PDF](http://www.oasis-open.org/committees/download.php/15377/xri-syntax-V2.0-cs.pdf)). |

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

### Appendix A.  Acknowledgements

This specification is the work of the OpenID AB/Connect Working Group, which includes dozens of active and dedicated participants. In particular, the following individuals contributed ideas, feedback, and wording that influenced this specification:

Andrew Arnott, Dirk Balfanz, Casper Biering, John Bradley, Johnny Bufu, Brian Campbell, Blaine Cook, Pamela Dingle, Vladimir Dzhuvinov, George Fletcher, Dick Hardt, Roland Hedberg, Edmund Jay, Michael B. Jones, Torsten Lodderstedt, Nov Matake, Breno de Medeiros, Chuck Mortimore, Anthony Nadalin, Axel Nennker, John Panzer, Justin Richer, Nat Sakimura, Owen Shepherd, Andreas Solberg, and Kick Willemse.

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

### Appendix B.  Notices

Copyright (c) 2013 The OpenID Foundation.

The OpenID Foundation (OIDF) grants to any Contributor, developer, implementer, or other interested party a non-exclusive, royalty free, worldwide copyright license to reproduce, prepare derivative works from, distribute, perform and display, this Implementers Draft or Final Specification solely for the purposes of (i) developing specifications, and (ii) implementing Implementers Drafts and Final Specifications based on such documents, provided that attribution be made to the OIDF as the source of the material, but that such attribution does not indicate an endorsement by the OIDF.

The technology described in this specification was made available from contributions from various sources, including members of the OpenID Foundation and others. Although the OpenID Foundation has taken steps to help ensure that the technology is available for distribution, it takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this specification or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any independent effort to identify any such rights. The OpenID Foundation and the contributors to this specification make no (and hereby expressly disclaim any) warranties (express, implied, or otherwise), including implied warranties of merchantability, non-infringement, fitness for a particular purpose, or title, related to this specification, and the entire risk as to implementing this specification is assumed by the implementer. The OpenID Intellectual Property Rights policy requires contributors to offer a patent promise not to assert certain patent claims against other contributors and against implementers. The OpenID Foundation invites any interested party to bring to its attention any copyrights, patents, patent applications, or other proprietary rights that may cover technology that may be required to practice this specification.

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

### Appendix C.  Document History

[[ To be removed from the final specification ]]

-17

* Required that the OpenID Provider configuration information be returned using the application/json content type.
* Updated the acct: URI reference to draft-ietf-appsawg-acct-uri-05.
* Updated normative text so that e-mail addresses use the acct: scheme.
* Added an example for the acct: scheme.
* Fixed #856 - Updated normative text to clarify that no input normalization is performed when the input contains an explicit scheme such as acct that matches the RFC 3986 scheme ":" path-rootless syntax.

-16

* Removed the version discovery element.
* Added a note about the future possibility of acct: URIs like acct:joe%40example.com@example.org when e-mail addresses are used as local account identifiers at sites.
* Stated that the JWS Compact Serialization and the JWE Compact Serialization are always used for JWS and JWE data structures.

-15

* Fixed #820 - Removed assumption that Clients that want encrypted responses also sign requests.

-14

* Fixed #801 - Removed schema and id parameters to UserInfo Endpoint.

-13

* Added Security Considerations section about TLS version requirements and usage.
* Removed language about supporting other transport-layer mechanisms with equivalent security to TLS.
* State that when any validations fail, any operations requiring the information that failed to correctly validate MUST be aborted and the information that failed to validate MUST NOT be used.
* Change from Content-Type application/json to application/jrd+json, tracking the change made in WebFinger.
* Fixed #768 - Added required version value to example response.
* Fixed #771 - Added required x509\_url value to example response.
* Fixed #769 - Added Claim Type identifiers and definition.
* Fixed #770 - Added claims\_locales\_supported and ui\_locales\_supported.
* Fixed #781 - Added require\_request\_uri\_registration discovery parameter.
* Fixed #772 - Added op\_policy\_url and op\_tos\_url.
* Fixed #782 - Changed uses of "\_url" in identifiers to "\_uri".
* Fixed #703 - Added the PKIX JWK key type for X.509 certificates and consolidated the x509\_uri, x509\_encryption\_uri, and jwk\_encryption\_uri parameters into a combined jwk\_uri parameter.
* Fixed #786 - Changed the name of jwk\_uri to jwks\_uri.
* Moved OP metadata list to its own section.
* Added the grant\_types\_supported discovery parameter.
* Added the claims\_parameter\_supported, request\_parameter\_supported, and request\_uri\_parameter\_supported discovery parameters.
* Fixed #788 - Renamed "OpenID Request Object" to "Request Object".
* Use legal acr values in examples.

-12

* Made the OpenID Foundation Artifact Binding Working Group the change controller for the values registered with IANA.
* Added display\_values\_supported, claim\_types\_supported, and claims\_supported discovery elements, fixing issue #656.
* Added Implementation Considerations section.
* Fixed #656 - Changed token\_endpoint\_auth\_type to token\_endpoint\_auth\_method and token\_endpoint\_auth\_types\_supported to token\_endpoint\_auth\_methods\_supported.
* Fixed #697 - Added service\_documentation to enable OPs not supporting dynamic registration to say how to register clients.
* Fixed #698 - Inconsistent use of articles.
* Fixed #628 - Defined REQUIRED, RECOMMENDED, and OPTIONAL discovery elements.
* Naming consistency changes. Renamed check\_session\_iframe\_url to check\_session\_iframe and end\_session\_endpoint\_url back to end\_session\_endpoint.
* Fixed #705 - Switched from using Simple Web Discovery to [WebFinger (Jones, P., Salgueiro, G., and J. Smarr, “WebFinger,” May 2013.)](" \l "I-D.ietf-appsawg-webfinger) [I‑D.ietf‑appsawg‑webfinger]. This also means that Identifiers using e-mail address syntax are prefixed by the acct: scheme when passed as resource parameters to WebFinger.

-11

* Fixed #687 - Inconsistency between user\_id and prn claims. The fix changed these names: user\_id -> sub, user\_id\_types\_supported -> subject\_types\_supported, user\_id\_type -> subject\_type, and prn -> sub.
* Renamed acrs\_supported to acr\_values\_supported for naming consistency.
* Fixed #676 Allow port number to be specified for e-mail syntax identifiers.
* Improved the fix for #625 Scheme extraction.
* Clarified that jwk\_url and jwk\_encryption\_url refer to documents containing JWK Sets - not single JWK keys.

-10

* Fixed #621 Changed Identifier definition
* Fixed #625 Scheme extraction
* Fixed #652 Identifier normalization
* Fixed #640 Added check\_session\_endpoint and end\_session\_endpoint
* Fixed #627 Configuration response must be 200 OK
* updated OAuth reference
* Clarify the use of .well-known as part of a path for multi-tenant
* Fixes #665 Add client\_secret\_jwt to token\_endpoint\_auth\_algs\_supported
* Fixed #614 - Discovery - 3.2 Distinguishing between signature and integrity parameters for HMAC algorithms. This fix tracks the parameter changes made to the JWE spec in draft-ietf-jose-json-web-encryption-06. It deletes the parameters {userinfo,id\_token}\_encrypted\_response\_int. It replaces the parameters {userinfo,id\_token,request\_object,token\_endpoint}\_algs\_supported with {userinfo,id\_token,request\_object,token\_endpoint}\_signing\_alg\_values\_supported and {userinfo,id\_token,request\_object,token\_endpoint}\_encryption\_{alg,enc}\_values\_supported.
* Fixed #666 - JWS signature validation vs. verification.
* Removed section on Redirection, since it was removed from Simple Web Discovery in favor of the "simple-web-discovery" domain prefix.
* Referenced OAuth 2.0 RFC -- RFC 6749.

-09

* Removed Check ID Endpoint, per issue #570
* Added PAPE Reference to the Informative References, per issue #574
* Added "id\_token" response type as being MTI for OpenID Providers
* Changed default OpenID Request Object signing algorithm to RS256, per issue #571
* Use standards track version of JSON Web Token spec (draft-ietf-oauth-json-web-token)

-08

* Remove the no path component restriction from issuer, per issue #513
* Updated Notices
* Updated References

-07

* Rename iso29115\_supported to acrs\_supported
* Rename jwk\_document to jwk\_url
* specify full email address to be used for the principal parameter
* Added token\_endpoint\_auth\_types\_supported for list of Token Endpoint authentication types
* Added token\_endpoint\_auth\_algs\_supported for Token Endpoint supported authentication algorithms
* Added 'pairwise' and 'public' to supported identifier types
* Added valid signature and encryption algorithms for OpenID Request Object
* Added URLs for JWK and X509 encryption keys
* Use RFC 6125 to verify TLS endpoints
* Removed fallback mechanism when discovery endpoint is unreachable
* Removed Account URI scheme
* Changed 'contact' to 'contacts', 'redirect\_uri' to 'redirect\_uris'
* Added section about string comparison rules needed
* Allows extensions to identifier normalization via specifications
* Clarifies the host in a URL
* Update John Bradley email and affiliation for Implementer's Draft
* Change flows\_supported to response\_types\_supported
* Register openid-configuration .well-known path in IANA Considerations
* Corrected instances of x509\_url\_encryption to x509\_encryption\_url and jwk\_url\_encryption to jwk\_encryption\_url

-06

* Changed Check Session Endpoint to Check ID Endpoint to match Basic.
* Changed certs\_url to x509\_url to match registration and JWE format.

-05

* Ticket #46 Added text to 3.3
* Deleted duplicate check session endpoint from 4.2
* Ticket #40 Added clarification of issuer url to 4.2
* Ticket #39 Cleaned up issuer examples and added verification text.

-04

* Changes associated with renaming "Lite" to "Basic Client" and replacing "Core" and "Framework" with "Messages" and "Standard".
* Numerous cleanups, including updating references.

-03

* Corrected examples.

-02

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

-01

* Incorporate working group decisions from 5-Jul-11 spec call.
* Consistency and cleanup pass, including removing unused references.

-00

* Initial version based upon former openid-connect-swd-1\_0 spec.

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

### Authors' Addresses

|  |  |
| --- | --- |
|  | Nat Sakimura |
|  | Nomura Research Institute, Ltd. |
| **Email:** | [n-sakimura@nri.co.jp](mailto:n-sakimura@nri.co.jp) |
|  |  |
|  | John Bradley |
|  | Ping Identity |
| **Email:** | [ve7jtb@ve7jtb.com](mailto:ve7jtb@ve7jtb.com) |
|  |  |
|  | Michael B. Jones |
|  | Microsoft |
| **Email:** | [mbj@microsoft.com](mailto:mbj@microsoft.com) |
|  |  |
|  | Edmund Jay |
|  | Illumila |
| **Email:** | [ejay@mgi1.com](mailto:ejay@mgi1.com) |