

# *OpenID Connect Conformance Profiles v1.0*

OpenID Connect Working Group, OpenID Foundation

March 29, 2015

## **1. Introduction**

This document defines the set of profiles of the OpenID Connect specifications used for certifying implementations conforming to those profiles. This document also lists the features that must be supported by implementations certified as conforming to each profile and lists the tests used to test those features.

Many but not all of the features are able to be tested using the self-certification test procedures established by the OpenID Connect working group and the OpenID Foundation. The testing procedures are described in the [Conformance Testing Procedures](#).

## **2. Overview of Conformance Profiles**

This section briefly describes each of the defined conformance profiles. In the published summaries of conformance self-certification results, these are the columns in the certification results table and implementations are the rows.

This section describes the conformance profiles included in the phase 1 launch of the OpenID Certification program in April 2015. While future phases of the OpenID Certification program will include Relying Party profiles, all of the phase 1 profiles are OpenID Provider profiles.

### **2.1 OpenID Provider Conformance Profiles**

#### **2.1.1 Basic OpenID Provider**

Basic OpenID Providers implement the features needed by Basic Relying Parties – essentially, those that use the features described in the [OpenID Connect Basic Client Implementer's Guide 1.0](#) (although the actual profile is based on [OpenID Connect Core 1.0](#)). These include the Mandatory to Implement Features for All OpenID Providers described in Section 15.1 of [OpenID Connect Core 1.0](#).

### 2.1.2 Implicit OpenID Provider

Implicit OpenID Providers implement the features needed by Implicit Relying Parties – those that use the features described in the [OpenID Connect Implicit Client Implementer’s Guide 1.0](#), excluding the Self-Issued OP features described in Section 4 (although the actual profile is based on [OpenID Connect Core 1.0](#)). These include the Mandatory to Implement Features for All OpenID Providers described in Section 15.1 of [OpenID Connect Core 1.0](#).

### 2.1.3 Hybrid OpenID Provider

Hybrid OpenID Providers implement the features needed by Hybrid Relying Parties – those that use the features described in Section 3.3 of [OpenID Connect Core 1.0](#). These include the Mandatory to Implement Features for All OpenID Providers described in Section 15.1 of [OpenID Connect Core 1.0](#).

### 2.1.4 OpenID Provider Publishing Configuration Information

OpenID Providers Publishing Configuration Information publish their discovery information at provider configuration endpoints, as described in Sections 3 and 4 of [OpenID Connect Discovery 1.0](#).

### 2.1.5 Dynamic OpenID Provider

Dynamic OpenID Providers implement the Mandatory to Implement Features for Dynamic OpenID Providers described in Section 15.2 of [OpenID Connect Core 1.0](#). Note that conforming to the Dynamic OpenID Provider profile also requires that the implementation conforms to the Basic OpenID Provider, Implicit OpenID Provider, and OpenID Provider Publishing Configuration Information profiles and implements the OP features of the [OpenID Connect Discovery 1.0](#) and [OpenID Connect Dynamic Client Registration 1.0](#) specifications.

## 3. Conformance Profile Definitions

### 3.1 OpenID Provider Conformance Profile Definitions

The following table specifies the protocol features included in the OpenID Provider conformance profiles defined above. It also names the tests in the OpenID Provider test suite at <http://op.certification.openid.net/> that are used to test those features.

Conformance Feature Information			OP Conformance Profiles				
Feature Name	Conformance Test Name	Test ID	Basic	Implicit	Hybrid	Config	Dynamic
Response Type & Response Mode							

Support code response_type	Request with response_type=code	OP-Response-code	y				
Support id_token response_type	Request with response_type=id_token	OP-Response-id_token		y			
Support id_token token response_type	Request with response_type=id_token token	OP-Response-id_token+token		y			
Support code id_token response_type	Request with response_type=code id_token	OP-Response-code+id_token			y		
Support code token response_type	Request with response_type=code token	OP-Response-code+token			y		
Support code id_token token response_type	Request with response_type=code id_token token	OP-Response-code+id_token+token			y		
Reject request without response_type	Authorization request missing the response_type parameter	OP-Response-Missing	y	y	y		
<b>ID Token</b>							
ID Token has iss claim		IdToken.verify()	y	y	y		
ID Token has sub claim		IdToken.verify()	y	y	y		
ID Token has aud claim		IdToken.verify()	y	y	y		
ID Token has iat claim		IdToken.verify()	y	y	y		
Does the OP sign the ID Token and with what	Does the OP sign the ID Token and with what	OP-IDToken-Signature	y	y	y		
Asymmetric ID Token signature with RS256	Asymmetric ID Token signature with RS256	OP-IDToken-RS256					y
ID Token has kid claim	IDToken has kid	OP-IDToken-kid	y	y	y		
Unsecured ID Token signature with none	Unsecured ID Token signature with none	OP-IDToken-none	y if uses none			y if uses none	y if uses none

ID Token has at_hash when ID Token and Access Token returned from Authorization Endpoint	ID Token has at_hash when ID Token and Access Token returned from Authorization Endpoint	OP-IDToken-at_hash		y	y		
ID Token has c_hash when ID Token and Authorization Code returned from Authorization Endpoint	ID Token has c_hash when ID Token and Authorization Code returned from Authorization Endpoint	OP-IDToken-c_hash			y		
<b>UserInfo Endpoint</b>							
Has UserInfo Endpoint	UserInfo Endpoint access with GET and bearer header	OP-UserInfo-Endpoint	y	y	y		
UserInfo Endpoint access with header method	UserInfo Endpoint access with POST and bearer header	OP-UserInfo-Header	y	y	y		
UserInfo Endpoint access with form-encoded body method	UserInfo Endpoint access with POST and bearer body	OP-UserInfo-Body	Warning if broken	Warning if broken	Warning if broken		
UserInfo has sub claim		OpenIDSchema.verify()	y	y	y		
Can provide signed UserInfo response with RS256	RP registers userinfo_signed_response_alg to signal that it wants signed UserInfo returned	OP-UserInfo-RS256					y
<b>nonce Request Parameter</b>							
Support requests without nonce when using the code flow	Login no nonce, code flow	OP-nonce-NoReq-code	y				
Reject requests without nonce unless using the code flow	Reject requests without nonce unless using the code flow	OP-nonce-NoReq-noncode		y	y		

ID Token has nonce when requested for code flow	ID Token has nonce when requested for code flow	OP-nonce-code	y				
ID Token has nonce when requested for non-code flows	Request with nonce, verifies it was returned in ID Token	OP-nonce-noncode		y	y		
<b>scope Request Parameter</b>							
Support openid scope	Does the OP sign the ID Token and with what	OP-IDToken-Signature	no err	no err	no err		
Support profile scope	Scope requesting profile claims	OP-scope-profile	no err	no err	no err		
Support email scope	Scope requesting email claims	OP-scope-email	no err	no err	no err		
Support address scope	Scope requesting address claims	OP-scope-address	no err	no err	no err		
Support phone scope	Scope requesting phone claims	OP-scope-phone	no err	no err	no err		
Support scope value requesting all basic claims	Scope requesting all claims	OP-scope-All	no err	no err	no err		
<b>display Request Parameter</b>							
Support display value page	Request with display=page	OP-display-page	no err	no err	no err		
Support display value popup	Request with display=popup	OP-display-popup	no err	no err	no err		
<b>prompt Request Parameter</b>							
Support prompt value login	Request with prompt=login	OP-prompt-login	y	y	y		
Support prompt value none	Request with prompt=none when not logged in	OP-prompt-none-NotLoggedIn	y	y	y		

Support prompt value none	Request with prompt=none when logged in	OP-prompt-none-LoggedIn	y	y	y		
<b>Misc Request Parameters</b>							
Support max_age request parameter	Requesting ID Token with max_age=1 seconds restriction	OP-Req-max_age=1	y	y	y		
ID Token has auth_time claim when max_age in request	Requesting ID Token with max_age=1 seconds restriction	OP-Req-max_age=1	y	y	y		
Support max_age request parameter when max age reached	Requesting ID Token with max_age=1 seconds restriction	OP-Req-max_age=1	Warning if no prompt	Warning if no prompt	Warning if no prompt		
Support max_age request parameter when max age not reached	Requesting ID Token with max_age=10000 seconds restriction	OP-Req-max_age=10000	y	y	y		
Ignores not understood query parameter in Authentication Request	Request with extra query component	OP-Req-NotUnderstood	y	y	y		
Support id_token_hint request parameter	Using prompt=none with user hint through id_token_hint	OP-Req-id_token_hint	SHOULD	SHOULD	SHOULD		
Support login_hint request parameter	Providing login_hint	OP-Req-login_hint	no err	no err	no err		
Support ui_locales request parameter	Providing ui_locales	OP-Req-ui_locales	no err	no err	no err		
Support claims_locales request parameter	Providing claims_locales	OP-Req-claims_locales	no err	no err	no err		
Support acr_values request parameter	Providing acr_values	OP-Req-acr_values	no err	no err	no err		
<b>OAuth Behaviors</b>							
OAuth state request value returned in response		VerifyState()	y	y	y		

Reject second use of Authorization Code	Trying to use authorization code twice should result in an error	OP-OAuth-2nd	Warning if under 30s		Warning if under 30s		
Reject second use of Authorization Code after 30 seconds	Trying to use authorization code twice with 30 seconds in between must result in an error	OP-OAuth-2nd-30s	OAuth MUST		OAuth MUST		
Second use of Authorization Code revokes previously issued Access Token	Trying to use authorization code twice should result in revoking previously issued access tokens	OP-OAuth-2nd-Revokes	OAuth SHOULD		OAuth SHOULD		
Reject second use of Authorization Code	Trying to use authorization code twice with 30 seconds in between must result in an error	OP-OAuth-2nd-30s	OAuth MUST		OAuth MUST		
<b>redirect_uri</b>							
Reject redirect_uri not matching a registered redirect_uri	Sent redirect_uri does not match a registered redirect_uri	OP-redirect_uri-NotReg	y	y	y		
Reject request without redirect_uri when multiple registered	Reject request without redirect_uri when multiple registered	OP-redirect_uri-Missing					y
Preserves query parameter in redirect_uri	Request with a redirect_uri with a query component when a redirect_uri with the same query component is registered	OP-redirect_uri-Query-OK					y

Preserves query parameter in registered redirect_uris	Request with a redirect_uri with a query component when a redirect_uri with the same query component is registered	OP-redirect_uri-Query-OK					y
Reject redirect_uri when query parameter does not match	Rejects redirect_uri when query parameter does not match what is registered	OP-redirect_uri-Query-Mismatch					y
Reject redirect_uri when query parameter added	Request with redirect_uri with query component when registered redirect_uri has no query component	OP-redirect_uri-Query-Added					y
Reject registration of redirect_uris with fragment	Registration where a redirect_uri has a fragment	OP-redirect_uri-RegFrag					y
<b>Client Authentication</b>							
Support client authentication to Token Endpoint using HTTP Basic with POST	Access token request with client_secret_basic authentication	OP-ClientAuth-Basic-Dynamic	y		y		
(same as above)	Access token request with client_secret_basic authentication	OP-ClientAuth-Basic-Static	y		y		
Support client authentication to Token Endpoint using form-encoded client credentials in POST body	Access token request with client_secret_post authentication	OP-ClientAuth-SecretPost-Dynamic	y		y		
(same as above)	Access token request with client_secret_post authentication	OP-ClientAuth-SecretPost-Static	y		y		



Discovery							
Publishes openid-configuration discovery information	Publishes openid-configuration discovery information	OP-Discovery-Config				y	y
Config has issuer		ProviderConfigurationResponse.verify()				y	y
Discovered issuer matches openid-configuration path prefix		ProviderConfigurationResponse.verify()				y	y
Discovered issuer matches ID Token iss value		IdToken.verify()				y	y
Config has authorization_endpoint		CheckEndpoint()				y	y
Config has token_endpoint		CheckEndpoint()				y unless only Implicit	y
Config has userinfo_endpoint		CheckEndpoint()				y unless self-issued	y
Config has jwks_uri	Verify that jwks_uri is published	OP-Discovery-jwks_uri				y unless only none	y
Keys in OP JWKS well formed	Keys in OP JWKS well formed	OP-Discovery-JWKS				y unless only none	y
Config has scopes_supported		CheckScopeSupport()				y	y
Config has response_types_supported		ProviderConfigurationResponse.verify()				y	y
Config has subject_types_supported		ProviderConfigurationResponse.verify()				y	y
Config has id_token_signing_alg_values_supported		ProviderConfigurationResponse.verify()				y unless only none	y

Config has claims_supported	Verify that claims_supported is published	OP-Discovery-claims_supported					y	y
All OP endpoints use https		VerifyOPEndpointsUseHTTPS()					y	y
Can Discover Identifiers using E-Mail Syntax	Can discover identifiers using e-mail syntax	OP-Discovery-WebFinger-Email						y
Support WebFinger discovery	Can discover identifiers using URL syntax	OP-Discovery-WebFinger						y
<b>Dynamic Client Registration</b>								
Config has registration_endpoint	Verify that registration_endpoint is published	OP-Registration-Endpoint						y
Enables dynamic registration	Client registration request	OP-Registration-Dynamic						y
Support using Sector Identifier for pairwise sub values								no err
Displays logo_uri in login page	Registration with logo_uri	OP-Registration-logo_uri						SHOULD
Displays policy_uri in login page	Registration with policy_uri	OP-Registration-policy_uri						SHOULD
Displays tos_uri in login page	Registration with tos_uri	OP-Registration-tos_uri						SHOULD
Uses keys registered with jwks value	Uses keys registered with jwks value	OP-Registration-jwks						y
Uses keys registered with jwks_uri value	Uses keys registered with jwks_uri value	OP-Registration-jwks_uri						y
Reject Sector Identifier not containing registered redirect_uri values	Incorrect registration of sector_identifier_uri	OP-Registration-Sector-Bad						y

Key Rotation								
Can rotate OP signing key	Can rotate OP signing keys	OP-Rotation-OP-Sig						y
Support RP signing key rotation	Request access token, change RSA signing key and request another access token	OP-Rotation-RP-Sig						y
request_uri Request Parameter								
Support request_uri request parameter	Support request_uri request parameter	OP-request_uri-Support						y
Support request_uri request parameter with unsecured request	Support request_uri request parameter with unsigned request	OP-request_uri-Unsigned	no err	no err	no err			
Support request_uri request parameter with unsecured request	Support request_uri request parameter with unsigned request	OP-request_uri-Unsigned-Dynamic						y
Support request_uri request parameter with signed request	Support request_uri request parameter with signed request	OP-request_uri-Sig						y
request Request Parameter								
Support request request parameter with unsecured request	Support request request parameter with unsigned request	OP-request-Unsigned	no err	no err	no err			
claims Request Parameter								
Support claims request parameter	Claims request with essential name claim	OP-claims-essential	no err	no err	no err			