OpenID Connect Conformance Profiles

OpenID Connect Working Group, OpenID Foundation

September 3, 2014

# Introduction

This document defines a set of profiles of the OpenID Connect specifications to be used for certifying implementations conforming to those profiles. In a future version of this document, each profile definition will list the features that must be supported by implementations certified as conforming to that profile.

Many but not all of the features will be able to be tested using self-certification test procedures to be established by the OpenID Connect working group and the OpenID Foundation. The testing procedures for these features will be described in a different document that is yet to be written. Defining how to achieve certification for these profiles is beyond the scope of this document.

# Overview of Conformance Profiles

This section briefly describes each of the defined conformance profiles. When we publish summaries of conformance self-certification results, these will be the columns in the certification results table and implementations will be the rows.

For the initial certification rollout, we will probably want to focus on only a subset of these profiles. Which profiles we want to launch with is discussed in the [Open Issues](#_Open_Issues) section of this document.

## 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](http://openid.net/specs/openid-connect-basic-1_0.html) (although the actual profile will be based on [OpenID Connect Core 1.0](http://openid.net/specs/openid-connect-core-1_0.html)). These features include the Mandatory to Implement Features for All OpenID Providers described in Section 15.1 of [OpenID Connect Core 1.0](http://openid.net/specs/openid-connect-core-1_0.html).

## Basic Relying Party

Basic Relying Parties implement the features described in the [OpenID Connect Basic Client Implementer’s Guide 1.0](http://openid.net/specs/openid-connect-basic-1_0.html) (although the actual profile will be based on [OpenID Connect Core 1.0](http://openid.net/specs/openid-connect-core-1_0.html)).

## 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](http://openid.net/specs/openid-connect-discovery-1_0.html). They also rotate their signing keys in the manner described in Section 10.1 of [OpenID Connect Core 1.0](http://openid.net/specs/openid-connect-core-1_0.html).

## Relying Party Using Configuration Information

OpenID Relying Parties Using Configuration Information obtain info about the OpenID Providers that they use from provider configuration endpoints, as described in Sections 3 and 4 of [OpenID Connect Discovery 1.0](http://openid.net/specs/openid-connect-discovery-1_0.html). They also support OP signing key rotation in the manner described in Section 10.1 of [OpenID Connect Core 1.0](http://openid.net/specs/openid-connect-core-1_0.html).

## 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](http://openid.net/specs/openid-connect-implicit-1_0.html), excluding the Self-Issued OP features described in Section 4 (although the actual profile will be based on [OpenID Connect Core 1.0](http://openid.net/specs/openid-connect-core-1_0.html)). These features include the Mandatory to Implement Features for All OpenID Providers described in Section 15.1 of [OpenID Connect Core 1.0](http://openid.net/specs/openid-connect-core-1_0.html).

## Implicit Relying Party

Implicit Relying Parties implement the features described in the [OpenID Connect Implicit Client Implementer’s Guide 1.0](http://openid.net/specs/openid-connect-implicit-1_0.html), excluding the Self-Issued OP features described in Section 4 (although the actual profile will be based on [OpenID Connect Core 1.0](http://openid.net/specs/openid-connect-core-1_0.html)).

## Self-Issued OpenID Provider

Self-Issued OpenID Providers implement the OP features described in Section 7 of [OpenID Connect Core 1.0](http://openid.net/specs/openid-connect-core-1_0.html). These OPs must also implement the Mandatory to Implement Features for All OpenID Providers described in Section 15.1 of [OpenID Connect Core 1.0](http://openid.net/specs/openid-connect-core-1_0.html).

## Self-Issued Relying Party

Self-Issued Relying Parties implement the RP features described in Section 7 of [OpenID Connect Core 1.0](http://openid.net/specs/openid-connect-core-1_0.html).

## Full OpenID Provider

Full OpenID Providers implement all six of the response\_type values specified in Section 3 of [OpenID Connect Core 1.0](http://openid.net/specs/openid-connect-core-1_0.html). They implement the “request”, “request\_uri”, and “claims” request parameters. They support encrypted requests and encrypted responses. They support rotation of RP and OP singing and encryption keys. They support both public and pairwise subject identifiers. They support offline access. They support all the client authentication methods defined in Section 9. These OPs must also implement the Mandatory to Implement Features for All OpenID Providers described in Section 15.1 of [OpenID Connect Core 1.0](http://openid.net/specs/openid-connect-core-1_0.html).

## Full Relying Party

Full Relying Parties implement all six of the response\_type values specified in Section 3 of [OpenID Connect Core 1.0](http://openid.net/specs/openid-connect-core-1_0.html). They implement the “request”, “request\_uri”, and “claims” request parameters. They support encrypted requests and encrypted responses. They support rotation of RP and OP singing and encryption keys. They can request offline access.

## 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](http://openid.net/specs/openid-connect-core-1_0.html). Note that conforming to the Dynamic OpenID Provider profile also means that the implementation will conform to the Basic OpenID Provider, Implicit OpenID Provider, and OpenID Provider Publishing Configuration Information profiles and implement the OP features of the [OpenID Connect Discovery 1.0](http://openid.net/specs/openid-connect-discovery-1_0.html) and [OpenID Connect Dynamic Client Registration 1.0](http://openid.net/specs/openid-connect-registration-1_0.html) specifications.

## Dynamic Relying Party

Dynamic Relying Parties implement the features of the Basic Relying Party, Implicit Relying Party, and Relying Party Using Configuration Information profiles. In addition to this, they implement the RP features of the [OpenID Connect Discovery 1.0](http://openid.net/specs/openid-connect-discovery-1_0.html) and [OpenID Connect Dynamic Client Registration 1.0](http://openid.net/specs/openid-connect-registration-1_0.html) specifications. It is recommended that Dynamic Relying Parties also seek certification as Relying Parties with Self-Issued OpenID Provider Support.

# Features Required for Conformance Profiles

TBD – For each profile, list the features that it contains. These features will be given semantic identifiers. When the feature is testable, also call this out.

These features already largely exist at <http://osis.idcommons.net/wiki/Category:OC5_Features> and in Roland Hedberg’s test suite, although will need to be given short semantic names for certification purposes.

# Open Issues

* This document probably defines too many profiles to start certification testing with. A proposal would be to initially roll out certification testing for these profiles:
  + Basic OpenID Provider
  + Basic Relying Party
  + OpenID Provider Publishing Configuration Information
  + Relying Party Using Configuration Information
  + Dynamic OpenID Provider
  + Dynamic Relying Party
* Should we define experimental profiles for certifying Session Management implementations?
* Should we define experimental profiles for certifying Form Post Response Mode implementations?
* What client authentication methods should the different profiles support?