MITRE

Enterprise Mission Tailored OpenID Connect (OIDC) Profile

The views, opinions and/or findings contained in this report are those of The MITRE Corporation and should not be construed as an official government position, policy, or decision, unless designated by other documentation.

Approved for Public Release; Distribution Unlimited. Public Release Case Number 19-3213

©2020 The MITRE Corporation.
All rights reserved.

Bedford, MA

Beth Abramowitz
Kelley Burgin
Tommy Farinelli
Neil McNab
Michael Peck
Mark Russell
Roger Westman

February 2020

This page intentionally left blank.

Table of Contents

1 Introduction

OpenID Connect, standardized by the OpenID Foundation [OIDC-Core], provides relying parties (RP) with the ability to delegate user authentication to an identity provider (IdP). Users authenticate to an IdP, and the IdP provides the RP with an assertion of the successful authentication.

This document profiles OpenID Connect for use in enterprise environments. This profile is derived from the International Government Assurance Profile (iGov) for OpenID Connect 1.0 [iGov-OIDC] produced by the OpenID Foundation.

OpenID Connect itself is a profile of the OAuth 2.0 web authorization framework [RFC6749]. This profile builds upon requirements found in the Enterprise OAuth 2.0 Profile. In OpenID Connect, the OAuth client is known as a Relying Party (RP), and the OAuth authorization server is known as an Identity Provider (IdP).

1.1 Requirements Notation and Convention

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

All uses of JSON Web Signature (JWS) and JSON Web Encryption (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.

1.2 Conformance

This specification defines requirements for the following components:

- OpenID Connect 1.0 relying parties (also known as OpenID Clients)
- OpenID Connect 1.0 identity providers (also known as OpenID Providers)

The requirements include details of interactions between these components:

- Relying party to identity provider

When a profile-compliant component is interacting with other profile-compliant components, in any valid combination, all components MUST fully conform to the features and requirements of this specification. All interaction with non-profile-compliant components is outside the scope of this specification.

A profile-compliant OpenID Connect IdP MUST support and utilize certain features as described in section 3 of this profile.

Since OpenID Connect builds upon the OAuth 2.0 specification, a profile-compliant OpenID Connect IdP MUST comply with all authorization server requirements in the Enterprise OAuth 2.0 Profile, with the exception that if it does not provide general OAuth 2.0 authorization server services, then functionality related to interaction between the authorization server and protected resources is OPTIONAL.

A profile-compliant OpenID Connect relying party MUST support and utilize certain features as described in section 2 of this profile.

Since OpenID Connect builds upon the OAuth 2.0 specification, a profile-compliant OpenID Connect relying party MUST comply with all client requirements in the Enterprise OAuth 2.0 Profile.

1.3 Environment Overview

This profile is intended for use in enterprise environments, not consumer-facing environments. Enterprise environments have different privacy and security considerations. For example, the base OpenID Connect specification includes optional privacy considerations to prevent relying parties from correlating user identities, while in enterprise environments relying parties generally need the ability to strongly identify users.

The enterprise is assumed to have a deployed public key infrastructure (PKI). The PKI issues each end user a certificate attesting to the user's identity. The PKI also issues non-person entity (NPE) certificates to relying parties and identity providers.

Users have attributes associated with them representing what types of data the user is permitted to access. Relying parties similarly have attributes associated with them. In environments where attributes are highly sensitive, relying parties can be restricted to obtain only attributes about the user that are shared with the relying party, i.e. the intersection of both entities' attributes.

1.4 Use Cases

This profile is oriented around one primary use case: user authentication to a web application / server.

This use case section is non-normative, and is intended to provide examples to set the stage for the rest of the profile document.

Authentication to native applications is another potential use case, but is not addressed at this time. Typically, users are not actually authenticating to a native application, but rather are authorizing the native application to access resources on behalf of the user. This use case is already addressed by the Enterprise OAuth Profile.

OAuth and OpenID Connect may be combined in different ways as part of an overall authentication and authorization workflow. A single authorization server may perform both OAuth and OpenID Connect functions. In that case, the requirements of the Enterprise OAuth 2.0 and OpenID Connect 1.0 profiles would apply to the interactions between the client and authorization server (known as relying party and identity provider respectively in OpenID Connect terminology).

In other cases, an OAuth authorization server might act as an OpenID Connect relying party for the purpose of authenticating users, relying upon a separate OpenID Connect identity provider for authentication. In the context of this profile, this use case is functionally identical to the User Authentication to a Web Application use case described below, with the OAuth authorization server acting in the role of the relying party web application.

1.4.1 User Authentication to a Web Application

In this use case, a web application (relying party) needs to authenticate a user. In many current enterprise environments, relying parties authenticate users through Transport Layer Security (TLS) client certificate authentication between the user's web browser and the relying party web server. As part of the TLS handshake, users prove possession of a private key associated with a public key infrastructure (PKI) certificate that uniquely identifies and authenticates the user. Although this method provides strong authentication, allowing OpenID Connect-based authentication to web servers brings potential advantages by offloading authentication complexities to an identity provider.

Using OpenID Connect can simplify the configuration of relying party web servers. Currently, each relying party web server must be configured with trusted certificates from the certification authorities (CA) that it trusts certificates from. These often include not only the relying party organization's CA but also other CAs belonging to partners, such as other agencies, foreign governments, and industry. With so many partners, these CA certificates may need to be frequently updated, placing a burden on the web server administrators. If OpenID Connect were instead used, the web server would be configured to trust assertions from its home organization's identity provider. The identity provider would handle the complexities of enabling authentication from multiple partners, rather than requiring it to be handled at each individual relying party.

Using OpenID Connect enables authentication method flexibility. There may be cases where TLS client certificate authentication is not appropriate or is not sufficient, making use of other authentication methods desired. TLS client certificate authentication of the user to the identity provider can of course still be used. It would be impractical for every relying party web server to be configured to handle alternative authentication methods, but it would become practical if that configuration only needed to occur at the identity provider.

For example, the "zero trust" security model advocates strongly authenticating both the user's identity and the identity and security properties of the user's endpoint computing system, in order to decrease reliance on enterprise network boundaries for security. The logic for analyzing endpoint system security properties as part of an authentication decision could be placed at the identity provider, but would be impractical to place at every relying party.

It may be necessary to authenticate users who do not possess a PKI certificate or have temporarily lost access to their private key. It may be desirable to require additional authentication methods in conjunction with TLS client certificate authentication, for example during an elevated threat condition, or to perform particularly sensitive operations. Examples of other potential authenticators include the Fast Identity Online (FIDO) standards (either using an external token such as a YubiKey or using a cryptographic store built into the endpoint computing device) and RSA SecurID.

Additionally, web browser-based TLS client certificate authentication is not widely used outside government environments. Some commercial-off-the-shelf (COTS) products acting in the relying party role may not directly support user authentication using TLS client certificates but may support OpenID Connect.

Figure 1 provides a high-level protocol overview of this use case.

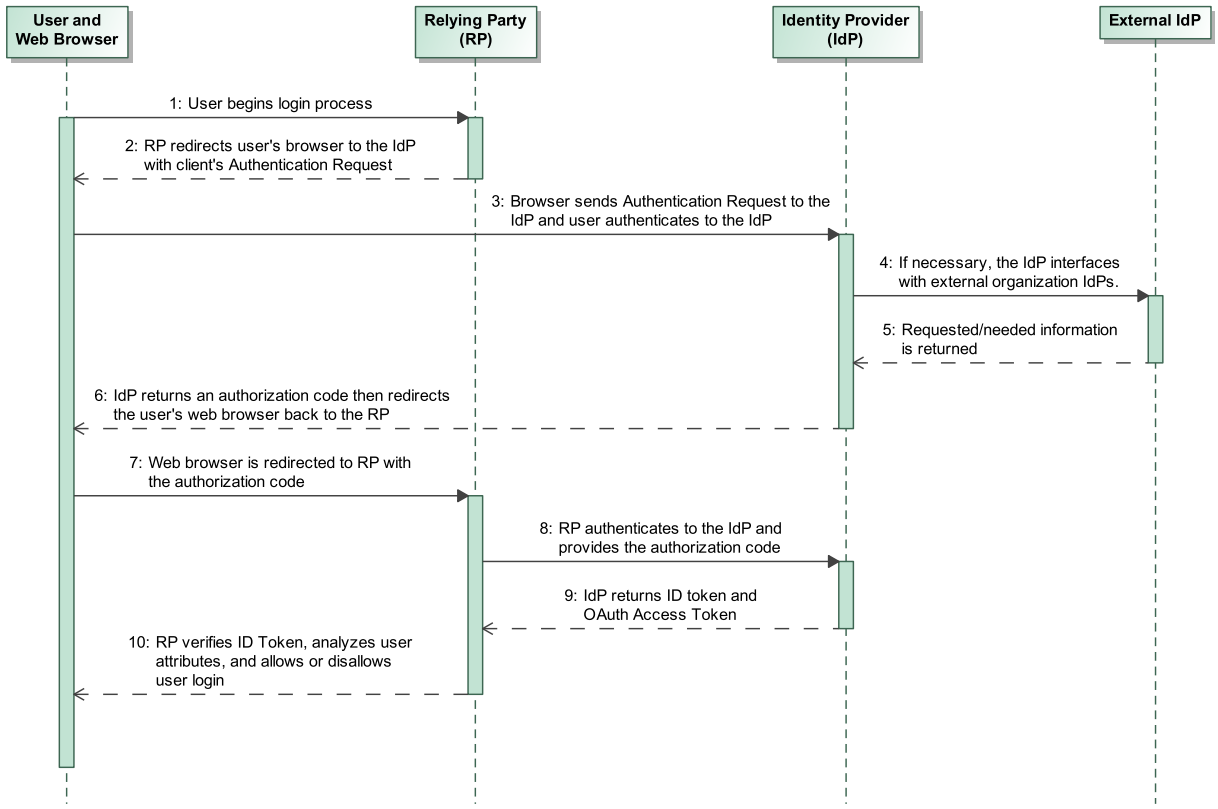

Figure 1. Figure 2 - Overview of OpenID Connect authentication

Figure 2 provides a high-level protocol overview of this use case including a non-exhaustive overview of this profile's requirements and recommendations.

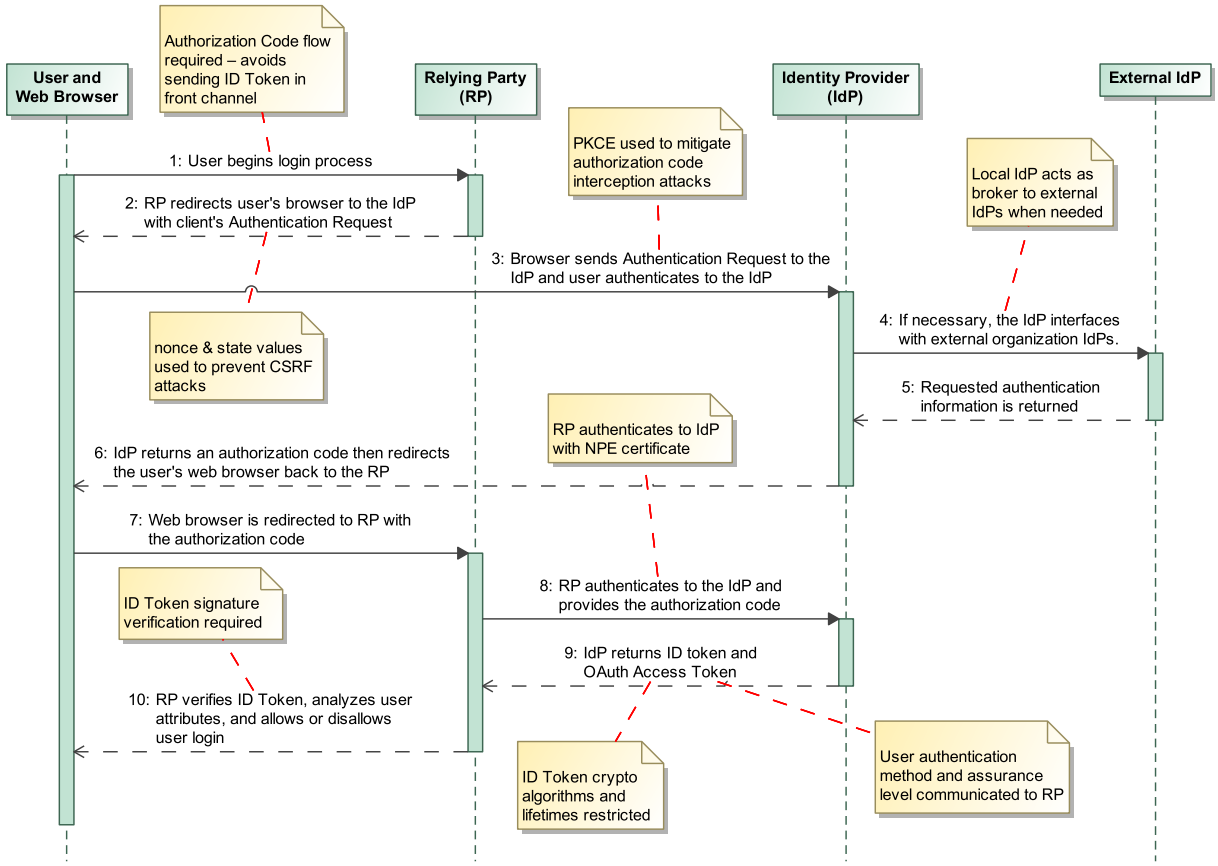

Figure 3: Overview of OpenID Connect authentication using profile requirements (non-exhaustive)

2 Relying Party Profile

This section profiles the expected OpenID Connect behavior of relying parties. Relying parties act in the role of OAuth client and are expected to conform with the Client Profiles section of the Enterprise OAuth Profile.

This profile assumes that OpenID Connect relying parties are OAuth confidential clients. Requirements for relying parties acting as OAuth public clients are out-of-scope and would need to be specified separately.

Each relying party MUST trust a single IdP. If interactions with multiple identity providers is required, the relying party's local identity provider can act as a broker to other identity providers.

2.1 Requests to the Authorization Endpoint (Authentication Request)

The Enterprise OAuth Profile specifies requirements for requests to Authorization Endpoints – for example, when to use the PKCE parameters to secure token exchange.

In addition to the requirements specified in Section 2.2.2 of the Enterprise OAuth Profile, the following describes the supported OpenID Connect Authorization Code Flow parameters for use with profile-compatible IdPs. See Section 3.1.2.1 of [OIDC-Core].

Request Parameters:

client_id	REQUIRED	The RP's OAuth 2.0 Client Identifier valid at the Identity Provider/Authorization Server
response_type	REQUIRED	MUST be set to code; the hybrid flows are not permitted under this profile
scope	REQUIRED	Indicates the attributes being requested. (See Section 4.2)
redirect_uri	REQUIRED	Indicates a valid endpoint where the client will receive the authentication response.
state	REQUIRED	Unguessable random string generated by the RP, used to protect against CSRF attacks. Must contain a sufficient amount of entropy to avoid guessing. Returned to the RP in the authentication response.
nonce	REQUIRED	Unguessable random string generated by the RP, used to protect against CSRF attacks. Must contain a sufficient amount of entropy to avoid guessing. Returned to the RP in the ID Token.
vtr	OPTIONAL	MUST be set to a value as described in Section 6.1 of Vectors of Trust [RFC8485]. vtr takes precedence over acr_values.
acr_values	OPTIONAL	Lists the acceptable LoAs for this authentication. See Section 3.1. MUST not be set if vtr is specified.
code_challenge and code_challenge_method	REQUIRED	If the PKCE protocol is being used by the RP. See Enterprise OAuth Profile.

A sample request may look like:

```
https://idp.government.gov/oidc/authorization?
    response_type=code
    &client_id=827937609728-m2mvqffo9bsefh4di90saus4n0diar2h
    &scope=openid
    &redirect_uri=https%3A%2F%2Frp.fed1.gov%2Foidc%2Flogin
Response
    &state=2ca3359dfbfd0
    &nonce=71d7b7e582067
    &code_challenge=2mjy65K8_lh9XlDiOQItYyYhArgzebK-Xx6K8lltE6A
    &code_challenge_method=S256
    &acr_values=http%3A%2F%2Fidmanagement.gov%2Fns%2F
assurance%2Floa%2F1
    +http%3A%2F%2Fidmanagement.gov%2Fns%2Fassurance%2Floa%2F2
    +http%3A%2F%2Fidmanagement.gov%2Fns%2Fassurance%2Floa%2F3
    +http%3A%2F%2Fidmanagement.gov%2Fns%2Fa
```

2.2 Requests to the Token Endpoint

Requirements for the request to the Token Endpoint are identical to the requirements specified in Section 2.2.3 of the Enterprise OAuth Profile.

2.3 ID Tokens

All relying parties MUST validate the signature of an ID Token before accepting it using the public key of the issuing server. The IdP's public signing keys MUST be made available in the jwks_uri claim in the IdP's discovery document, and MAY be made available in the form of NPE certificates issued to the IdP. The jwks_uri endpoint MUST be served over HTTPS. ID Tokens MAY be encrypted using the appropriate key of the requesting relying party.

Relying parties MUST verify the following in received ID tokens:

iss	The "issuer" field is the Uniform Resource Locater (URL) of the expected issuer
aud	The "audience" field contains the client ID of the RP
nonce	Must match the nonce value submitted in the authentication request
exp	Expiration timestamp for the token is a date (integer number of seconds since from 19700101T00:00:00Z UTC)
iat	Issued at timestamp for the token is a date (integer number of seconds since from 19700101T00:00:00Z UTC)

2.4 Request Objects

RPs MAY optionally send requests to the authorization endpoint using the request parameter as defined by OpenID Connect. RPs MAY send requests to the authorization endpoint by reference using the request_uri parameter.

Request objects MUST either be signed by a key corresponding to an X.509 certificate issued to the RP or by a key corresponding to a public key registered with the IdP. Request objects MAY be encrypted to the IdP's public key.

2.5 Discovery

RPs SHOULD cache OpenID Provider metadata once an IdP has been discovered and used by the RP. If HTTP cache headers are supplied by the IdP, metadata MUST NOT be re-requested before indicated by the headers. Metadata SHOULD NOT be re-requested from the IdP sooner than 24 hours after the most recent successful request. In the case of an unsuccessful request and cached metadata, re-request SHOULD NOT be made for at least 60 minutes.

Cached metadata MUST expire and after that time MUST be discarded. Cached metadata SHOULD be discarded when 30 days have passed since the most recent successful request, but MAY be discarded sooner.

3 Identity Provider Profile

This section profiles the expected OpenID Connect behavior of identity providers. Identity providers act in the role of OAuth authorization server and are expected to conform with the Authorization Server Profile section of the Enterprise OAuth Profile, with the exception that the Enterprise OAuth Profile's protected resource requirements are only required if the identity provider / authorization server provides general OAuth authorization server functionality.

As stated in section 2, each relying party MUST trust a single IdP. In the common enterprise use case with PKI authentication, a local IdP can directly authenticate users from partner organizations and obtain their attributes from an attribute service. In some cases, interactions with other IdPs may be necessary (for example, for interacting with a partner organization that does not use PKI or whose user attributes are not available through an attribute service). In these cases, the IdP may act as a broker by redirecting the user to another IdP. In these cases, the IdP acting as a broker may be considered both an IdP in relation to the application being accessed and a relying party in relation to the other IdP.

3.1 ID Tokens

All ID Tokens MUST be signed by the IdP's private signature key. ID Tokens MAY be encrypted using the appropriate key of the requesting RP. IdPs MUST support the RS256 signature method (the Rivest, Shamir, and Adleman (RSA) signature algorithm with at least a 256 bit hash) and MAY also use the following signature algorithms: RS384, RS512, ES256, ES384, ES512, PS256, PS384, PS512.

The ID Token MUST expire and SHOULD have an active lifetime no longer than five minutes. Since the ID token is consumed by the RP and not presented to remote systems, much shorter expiration times are RECOMMENDED where possible.

The token response includes an access token (which can be used to make a UserInfo request) and ID token (a signed and optionally encrypted JSON Web Token). ID Token values have the following meanings:

iss	REQUIRED	The "issuer" field is the Uniform Resource Locater (URL) of the expected issuer.
aud	REQUIRED	The "audience" field contains the client ID of the RP.
sub	REQUIRED	A value that uniquely identifies the user. For example, the full Distinguished Name (DN) from the user's client certificate (if available).
vot	OPTIONAL	The vector value as specified in Vectors of Trust [RFC8485]. See Section 3.4 for more details. vot takes precedence over acr.
vtm	REQUIRED if vot is provided.	The trustmark URI as specified in Vectors of Trust. See Section 3.4 for more details.
acr	REQUIRED	The authentication class with which the user authenticated. MUST be a member of the acr_values list from the authentication

		request. Values for this field may correspond to NIST Authenticator Assurance Levels (AALs); other values may be defined for use in a specific community. The IdP MAY include this claim in addition to "vot" for clients that do not support vot. See Authentication Context for more details. .
amr	REQUIRED	The user's authentication method to the IdP. See below for sample values for this field.
nonce	REQUIRED	MUST match the nonce value that was provided in the authentication request.
jti	REQUIRED	A unique identifier for the token, which can be used to prevent reuse of the token.
auth_time	REQUIRED	This MUST be included if the provider can assert an end user's authentication intent was demonstrated. For example, a login event where the user took some action to authenticate.
exp	REQUIRED	The expiration time (integer number of seconds since from 1970-01-01T00:00:00Z UTC), after which the token MUST be considered invalid
iat	REQUIRED	Issued at timestamp
at_hash	REQUIRED	Access token hash value (see section 3.1.3.6 of OpenID Connect Core for details on generating this field)

Authentication Context Class Reference (acr): A string specifying a defined Authentication Context Class Reference. The following URLs defined in the Federal Identity, Credential, and Access Management (FICAM) MAY be used to convey assurance levels defined in NIST SP 800-63-2:

- http://idmanagement.gov/ns/assurance/loa/1
- http://idmanagement.gov/ns/assurance/loa/2
- http://idmanagement.gov/ns/assurance/loa/3
- http://idmanagement.gov/ns/assurance/loa/4

These values may be superseded by a future specification of standard values to convey AAL, IAL, and FAL. IdPs and RPs MAY define additional acr values that have agreed-upon definitions for a given user community or mission area.

Authentication Methods Reference (amr): a JSON array of strings indicating authentication methods used to authenticate the user to the IdP. May have multiple values when mutli-factor authentication is used. [RFC 8176] provides a set of standard amr values. However, community discussion and agreement is needed to determine the applicability of a given authentication mechanism and the specific definitions of amr values. The definition and adoption of specific amr values is out of scope for this profile.

3.2 UserInfo Endpoint

IdPs MUST support the UserInfo Endpoint and, at a minimum, the sub (subject) claim.

Support for a UserInfo Endpoint is important for maximum relying party implementation interoperability even if no additional user information is returned. Relying parties are not required to call the UserInfo Endpoint, but should not receive an error if they do.

In an example transaction, the relying party sends a request to the UserInfo Endpoint like the following:

```
GET /userinfo HTTP/1.1
Authorization: Bearer
eyJhbGciOiJSUzI1NiJ9.eyJleHAiOjE0MTg3MDI0MTIsImF1ZCI6WyJjMWJjODR
lNC00N2VlLTRiNjQtYmI1Mi01Y2RhNmM4MWY3ODgiXSwiaXNzIjoiaHR0cHM6XC9
cL2lkcC1wLmV4YW1wbGUuY29tXC8iLCJqdGkiOiJkM2Y3YjQ4Zi1iYzgxLTQwZWM
tYTE0MC05NzRhZjc0YzRkZTMiLCJpYXQiOjE0MTg2OTg4MTJ9i.HMz_tzZ90_b0Q
ZS-AXtQtvclZ7M4uDAs1WxCFxpgBfBanolW37X8h1ECrUJexbXMD6rrj_uuWEqPD
738oWRo0rOnoKJAgbF1GhXPAYnN5pZRygWSD1a6RcmN85SxUig0H0e7drmdmRkPQ
gbl2wMhu-6h2Oqw-ize4dKmykN9UX_2drXrooSxpRZqFVYX8PkCvCCBuFy2O-
HPRov_SwtJMk5qjUWMyn2I4Nu2s-R20aCA-7T5dunr0iWCkLQnVnaXMfA22RlRiU
87nl21zappYb1_EHF9ePyq3Q353cDUY7vje8m2kKXYTgc_bUAYuW-W3SMSw5UlKa
HtSZ6PQICoA
Accept: text/plain, application/json, application/*+json, */*
Host: idp-p.example.com
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.2.3 (java 1.5)
```

And receives a document in response like the following:

```
HTTP/1.1 200 OK
Date: Tue, 16 Dec 2014 03:00:12 GMT
Access-Control-Allow-Origin: *
Content-Type: application/json;charset=ISO-8859-1
Content-Language: en-US
Content-Length: 333
Connection: close
{
     "sub": "6WZQPpnQxV",
     "iss": "https://idp-p.example.com"
     "given_name": "Stephen",
     "family_name": "Emeritus",
}
```

IdPs MUST support the generation of JWT encoded responses from the UserInfo Endpoint in addition to unsigned JSON objects. Signed responses MUST be signed by the IdP's key, and encrypted responses MUST be encrypted with the authorized RP's public key. Hashing and signature algorithm requirements for UserInfo responses are the same as those described in Section 3.1 regarding ID Tokens.

IdPs MAY provide different sets of user claims in the ID Token and UserInfo endpoint. For example, an IdP that provides a large number of user claims could provide a baseline set of claims in the ID Token and enable RPs to request additional claims as needed from the UserInfo endpoint.

3.3 Request Objects

IdPs MUST accept requests containing a request object signed by the RP's private key. IdPs MUST validate the signature on such requests against either an X.509 certificate belonging to the RP (whose Distinguished Name is associated with the RP's registration on the IdP) or a public key registered to the RP by the IdP. IdPs SHOULD accept request objects encrypted with the IdP's public key (this would require the IdP to publish a public key suitable for key agreement or key establishment).

IdPs MAY accept request objects by reference using the request_uri parameter. If request_uri is used, its value MUST be an HTTPS URL.

Both of these methods allow for RPs to create a request that is protected from tampering through the browser, allowing for a higher security mode of operation for RPs that require it. RPs are not required to use request objects, but IdPs are required to support requests using them.

3.4 Vectors of Trust

As vectors of trust is an emerging concept, use of the vtr value and vot field is OPTIONAL. If the vtr (Vectors of Trust Request) value is present in the authorization request as defined in the Vectors of Trust standard, the IdP SHOULD respond with a valid vot value as defined in Section 3.1. Both the vtr and vot MUST contain values in accordance with the Vectors of Trust standard. These values MAY be those defined in the Vectors of Trust standard directly or MAY be from a compatible standard. The IdP MAY require the user to reauthenticate, provide a second factor, or perform another action in order to fulfill the state requested in the vtr.

For backwards compatibility RPs MAY send an acr_values parameter. If both the vtr and acr_values are in the request, the vtr MUST take precedence and the acr_values MUST be ignored.

It is out of the scope of this document to determine how an organization maps their digital identity practices to valid VOT component values.

3.5 Authentication Context

IdPs MUST provide acr (authentication context class reference, equivalent to the Security Assertion Markup Language (SAML) element of the same name) and MUST provide amr (authentication methods reference) values in ID tokens.

The acr and amr are defined in Section 3.1.

3.6 Discovery

OpenID Connect Discovery provides a standard, programmatic way for RPs to obtain configuration details for communicating with IdPs. Exposing a Discovery endpoint does NOT inherently put the IdP at risk to attack. Endpoints and parameters specified in the Discovery document should be considered public information regardless of the existence of the Discovery document. IdPs MUST provide a Discovery endpoint at the standard well-known URL specified in [OIDC-Discovery].

Access to the Discovery document MAY be protected by requiring client TLS authentication. Endpoints described in the Discovery document MUST use HTTPS and MAY have additional controls the IdP wishes to support.

All IdPs are uniquely identified by a URL known as the issuer. This URL serves as the prefix of a service discovery endpoint as specified in the OpenID Connect Discovery standard. The discovery document MUST contain at minimum the following fields:

issuer	REQUIRED	The fully qualified issuer URL of the OpenID Provider.
authorization_endpoint	REQUIRED	The fully qualified URL of the IdP's authorization endpoint defined by [RFC6749].
token_endpoint	REQUIRED	The fully qualified URL of the server's token endpoint defined by [RFC6749].
introspection_endpoint	OPTIONAL	The fully qualified URL of the server's introspection endpoint defined by OAuth Token Introspection.
revocation_endpoint	OPTIONAL	The fully qualified URL of the server's revocation endpoint defined by OAuth Token Revocation.
jwks_uri	REQUIRED	The fully qualified URI of the IdP’s public key in JWK Set format. For verifying the signatures on the id_token.
scopes_supported	REQUIRED	The list of scopes the server supports.
claims_supported	REQUIRED	The list of claims available in the supported scopes. See below.
vot	OPTIONAL	The vectors supported.
acr_values	OPTIONAL	The acrs supported.

The following example shows the JSON document found at a discovery endpoint for an identity provider:

```
{
    "request_parameter_supported": true,
    "id_token_encryption_alg_values_supported": [
      "RSA-OAEP", "RSA1_5", "RSA-OAEP-256"
    ],
```

```
    "registration_endpoint": "https://idp-
p.example.com/register",
    "userinfo_signing_alg_values_supported": [
      "RS256", "RS384", "RS512"
    ],
    "token_endpoint": "https://idp-p.example.com/token",
    "request_uri_parameter_supported": false,
    "request_object_encryption_enc_values_supported": [
      "A192CBC-HS384", "A192GCM", "A256CBC+HS512",
      "A128CBC+HS256", "A256CBC-HS512",
      "A128CBC-HS256", "A128GCM", "A256GCM"
    ],
    "token_endpoint_auth_methods_supported": [
      "tls_client_auth"
    ],
    "userinfo_encryption_alg_values_supported": [
      "RSA-OAEP", "RSA1_5",
      "RSA-OAEP-256"
    ],
    "subject_types_supported": [
      "public"
    ],
    "id_token_encryption_enc_values_supported": [
      "A192CBC-HS384", "A192GCM", "A256CBC+HS512",
      "A128CBC+HS256", "A256CBC-HS512", "A128CBC-HS256",
      "A128GCM", "A256GCM"
    ],
    "claims_parameter_supported": false,
    "jwks_uri": "https://idp-p.example.com/jwk",
    "id_token_signing_alg_values_supported": [
      "RS256", "RS384", "RS512", "none"
    ],
    "authorization_endpoint": "https://idp-
p.example.com/authorize",
    "require_request_uri_registration": false,
    "introspection_endpoint": "https://idp-
p.example.com/introspect",
    "request_object_encryption_alg_values_supported": [
      "RSA-OAEP", RSA1_5", "RSA-OAEP-256"
    ],
    "service_documentation": "https://idp-p.example.com/about",
    "response_types_supported": [
      "code", "token"
    ],
    "token_endpoint_auth_signing_alg_values_supported": [
      "RS256", "RS384", "RS512"
    ],
    "revocation_endpoint": "https://idp-p.example.com/revoke",
```

```
    "request_object_signing_alg_values_supported": [
      "RS256", "RS384", "RS512"
    ],
    "claim_types_supported": [
      "normal"
    ],
    "grant_types_supported": [
      "authorization_code",
    ],
    "scopes_supported": [
      "profile", "openid", "doc"
    ],
    "userinfo_endpoint": "https://idp-p.example.com/userinfo",
    "userinfo_encryption_enc_values_supported": [
      "A192CBC-HS384", "A192GCM",
"A256CBC+HS512","A128CBC+HS256",
      "A256CBC-HS512", "A128CBC-HS256", "A128GCM", "A256GCM"
    ],
    "op_tos_uri": "https://idp-p.example.com/about",
    "issuer": "https://idp-p.example.com/",
    "op_policy_uri": "https://idp-p.example.com/about",
    "claims_supported": [
      "sub", "name", "vot", "acr"
    ],
    "vot": "???"
    "acr_values": [
      "http://idmanagement.gov/ns/assurance/loa/2",
      "http://idmanagement.gov/ns/assurance/loa/3",
      "http://idmanagement.gov/ns/assurance/loa/4",
    ]
}
```

It is RECOMMENDED that IdPs provide cache information through standard HTTP caching headers such as Cache-Control with max-age or Expires. HTTP caching headers SHOULD be set to a minimum of 24 hours.

The IdP MAY provide its public key in JWK Set format, such as the following 2048-bit RSA key:

```
{
"keys": [
    {
        "alg": "RS256",
        "e": "AQAB",
        "n":
"o80vbR0ZfMhjZWfqwPUGNkcIeUcweFyzB2S2T-hje83IOVct8gVg9FxvHPK1ReE
W3-p7-A8GNcLAuFP_8jPhiL6LyJC3F10aV9KPQFF-w6Eq6VtpEgYSfzvFegNiPtp
```

```
MWd7C43EDwjQ-GrXMVCLrBYxZC-P1ShyxVBOzeR_5MTC0JGiDTecr_2YT6o_3aE2
SIJu4iNPgGh9MnyxdBo0Uf0TmrqEIabquXA1-V8iUihwfI8qjf3EujkYi7gXXelI
o4_gipQYNjr4DBNlE0__RI0kDU-27mb6esswnP2WgHZQPsk779fTcNDBIcYgyLuj
lcUATEqfCaPDNp00J6AbY6w",
        "kty": "RSA",
        "kid": "rsa1"
      }
] }
```

4 User Info

The availability, quality, and reliability of an individual's identity attributes will vary greatly across jurisdictions and IdP systems. The following recommendations ensure maximum cross jurisdictional interoperability, while setting RP expectations on the type of data they may acquire.

4.1 Claims Supported

Discovery mandates the inclusion of the claims_supported field that defines the claims an RP MAY expect to receive for the supported scope values. IdPs MUST return claims on a best effort basis. However, an IdP asserting it can provide a user claim does not imply that this data is available for all its users: RPs MUST be prepared to receive partial data. Providers MAY return claims outside of the claims_supported list, but they MUST still ensure that the extra claims do not violate the policies set out by the federation, which may include filtering the returned attributes based on the relying party's attributes.

This profile does not specify claim names or values. The specific claims to be used in a given environment will be addressed in that environment's claims management specification or dictionary. It is hoped that claim names and values will be harmonized as much as practical across different mission enterprises.

4.2 Scope Profiles

In OpenID Connect, scopes are generally used by relying parties to request that specific sets of claims about the user be returned in the ID Token and/or from the UserInfo endpoint. The OpenID Connect Core specification defines the following standard scopes. IdPs MUST recognize these standard scopes, though they are not required to return all corresponding claims to all relying parties.

profile	OPTIONAL	This scope value requests access to the End-User's default profile Claims, which are: name, family_name, given_name, middle_name, nickname, preferred_username, profile, picture, website, gender, birthdate, zoneinfo, locale, and updated_at.
email	OPTIONAL	This scope value requests access to the email and email_verified Claims.

address	OPTIONAL	This scope value requests access to the address Claim.
phone	OPTIONAL	This scope value requests access to the phone_number and phone_number_verified Claims

IdPs MAY support additional scope values and corresponding claim sets as needed to support mission needs.

4.3 Claims Request

OpenID.Core section 5.5 defines a method for a RP to request specific claims in the UserInfo object. IdPs SHOULD support this claims parameter in the interest of data minimization; that is, the IdP only returns information on the subject the RP specifically asks for, and does not volunteer additional information about the subject.

RPs requesting the profile scope MAY provide a claims request parameter. If the claims request is omitted, the IdP SHOULD provide a default claims set that it has available for the subject, in accordance with any policies set out by the trust framework the IdP supports.

4.4 Claims Response

Response to a UserInfo request MUST match the scope and claims requested to avoid having an IdP overexpose a user's identity information.

Claims response MAY also make use of the aggregated and/or distributed claims structure to refer to the original source of the subject's claims.

4.5 Claims Metadata

Claims Metadata (such as locale or the confidence level the IdP has in the claim for the user) can be expressed as attributes within the UserInfo object, but are outside the scope of this document. These types of claims are best described by the trust framework the RPs and IdPs operate within.

5 Privacy Considerations

Data minimization is an essential concept in trust frameworks and federations exchanging user identity information for government applications. The design of this specification takes into consideration mechanisms to protect the user's government identity information and activity from unintentional exposure. Values for sensitive user attributes need to be limited to only those applications and services with a verified need to know.

Request claims SHOULD be supported by IdPs to ensure that only the data the RP explicitly requests is provided in the UserInfo response. This prevents situations where an RP may only require a partial set of claims, but receives (and is therefore exposed to) a full set of claims.

For example, System A is accredited to operate up to the SECRET level. User B has a TOP SECRET clearance the IdP knows of. System A registers with the OpenID Provider that it needs to know the clearance level of the users connecting to the system.

Using a traditional attribute sharing scheme, when User B logs into System A with OpenID Connect, the UserInfo response indicates User B is cleared up to the TOP SECRET level. This is not desired as it unnecessarily discloses to System A the fact that User B has a TOP SECRET clearance.

The desired approach is that the IdP also knows the accreditation level of System A (or can query a data source for this information) and filters the information provided to System A accordingly. When User B logs into System A with OpenID Connect, the UserInfo response indicates User B is cleared up to the SECRET level. Even though User B is cleared to TOP SECRET, this is not disclosed to System A because it has no need to know, it does not process information at the TOP SECRET level. User B is still able to access all information he is entitled to in System A as the initial scenario.

6 Security Considerations

All transactions MUST be protected in transit by TLS as described in BCP195.

All implementations MUST conform to applicable recommendations found in the Security Considerations sections of [RFC6749] and those found in the OAuth 2.0 Threat Model and Security Considerations document.

7 Normative References

[OIDC-Core] OpenID Foundation. "OpenID Connect Core 1.0 incorporating errata set 1", November 2014, <https://openid.net/specs/openid-connect-core-1_0.html>.

[iGov-OIDC] M. Varley and P. Grassi. "International Government Assurance Profile (iGov) for OpenID Connect 1.0 - Draft 03," October 2018, <https://openid.net/specs/openid-igov-openid-connect-1_0-03.html>.

[RFC6749] Hardt, D., Ed. "The OAuth 2.0 Authorization Framework", RFC 6749, DOI 10.17487/RFC6749, October 2012, <http://www.rfc-editor.org/info/rfc6749>.

[RFC8485] Richer, J. and Johansson, L. "Vectors of Trust," October 2018, <https://tools.ietf.org/html/rfc8485>.

[OIDC-Discovery] OpenID Foundation. "OpenID Connect Discovery 1.0 incorporating errata set 1", November 2014, <https://openid.net/specs/openid-connect-discovery-1_0.html>.

8 Informative References

[RFC4211] Schaad, J. “Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF)”, September 2005, <https://tools.ietf.org/html/rfc4211>.

[NIST.800-63-2] National Institute of Standards and Technology (NIST), "Electronic Authentication Guideline", NIST Special Publication 800-63-2, August 2013, <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63-2.pdf >.

[RFC4226] M'Raihi, D., Bellare, M., Hoornaert, F., Naccache, D., and O. Ranen, "HOTP: An HMAC-Based One-Time Password Algorithm", RFC 4226, December 2005. <http://www.ietf.org/rfc/rfc4226.txt>.

[RFC6238] M'Raihi, D., Machani, S., Pei, M., and J. Rydell, "TOTP: Time-Based One-Time Password Algorithm", RFC 6238, May 2011. <http://www.ietf.org/rfc/rfc6238.txt>.

[MSDN Microsoft, "Integrated Windows Authentication with Negotiate", September 2011, <https://blogs.msdn.com/b/benjaminperkins/archive/2011/09/14/iis-integrated-windows-authentication-with-negotiate.aspx>.

Appendix A Acronyms

acr	authentication context class reference
amr	authentication methods reference
iGov	International Government Assurance Profile
JSON	JavaScript Object Notation
JWA	JSON Web Algorithms
JWT	JSON Web Token
OIDC	OpenID Connect
SAML	Security Assertion Markup Language
URL	Uniform Resource Locator
vot	Vector of Trust
vtr	Vectors of Trust Request