JWT Response for OAuth Token Introspection
draft-ietf-oauth-jwt-introspection-response-04

Abstract

This draft proposes an additional JSON Web Token (JWT) based response for OAuth 2.0 Token Introspection.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 23, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.
1. Introduction

OAuth 2.0 Token Introspection [RFC7662] specifies a method for a protected resource to query an OAuth 2.0 authorization server to determine the state of an access token and obtain data associated with the access token. This allows deployments to implement identifier-based access tokens in an interoperable way.

The introspection response, as specified in OAuth 2.0 Token Introspection [RFC7662], is a plain JSON object. However, there are use cases where the resource server requires stronger assurance that the authorization server issued the access token, including cases where the authorization server assumes liability for the token’s content. An example is a resource server using verified person data to create certificates, which in turn are used to create qualified electronic signatures.

In such use cases it may be useful or even required to return a signed JWT as the introspection response. This specification extends the token introspection endpoint with the capability to return responses as JWTs.
2. Requesting a JWT Response

A resource server requests to receive a JWT introspection response by including an Accept header with content type "application/jwt" in the introspection request.

The following is a non-normative example request:

```plaintext
POST /introspect HTTP/1.1
Host: server.example.com
Accept: application/jwt
Content-Type: application/x-www-form-urlencoded

token=2YotnFZFEjr1zCsicMWPAA
```

3. JWT Response

The introspection endpoint responds with a JWT, setting the Content-Type header to "application/jwt".

This JWT MUST contain the claims "iss" and "aud" in order to prevent misuse of the JWT as ID or access token (see Section 6.1).

This JWT MAY furthermore contain all claims defined in the "OAuth Token Introspection Response" registry established by [RFC7662].

The following is a non-normative example response (with line breaks for display purposes only):

```plaintext
HTTP/1.1 200 OK
Content-Type: application/jwt

eyJraWQiOiIxIiwiYWxnIjoiaXMyNTYifQ.eyJzdWIiOiJ2YotnFZFEjr1zCsicMWPAA.
```

The example response contains the following JSON document:
Depending on the specific resource server policy the JWT is either signed, or signed and encrypted. If the JWT is signed and encrypted it MUST be a Nested JWT, as defined in JWT [RFC7519].

Note: If the resource server policy requires a signed and encrypted response and the authorization server receives an unauthenticated request containing an Accept header with content type other than "application/jwt", it MUST refuse to serve the request and return an HTTP status code 400. This is done to prevent downgrading attacks to obtain token data intended for release to legitimate recipients only (see Section 6.2).

4. Client Metadata

The authorization server determines what algorithm to employ to secure the JWT for a particular introspection response. This decision can be based on registered metadata parameters for the resource server, supplied via dynamic client registration with the resource server posing as the client, as defined by this draft.

The parameter names follow the pattern established by OpenID Connect Dynamic Client Registration [OpenID.Registration] for configuring signing and encryption algorithms for JWT responses at the UserInfo endpoint.

The following client metadata parameters are introduced by this specification:

introspection_signed_response_alg OPTIONAL. JWS [RFC7515] algorithm ("alg" value) as defined in JWA [RFC7518] for signing introspection responses. If this is specified, the response will be signed using JWS and the configured algorithm. The default, if omitted, is "RS256".
introspection_encrypted_response_alg  OPTIONAL.  JWE [RFC7516]
algorithm ("alg" value) as defined in JWA [RFC7518] for
encrypting introspection responses. If this is specified,
the response will be encrypted using JWE and the configured
algorithm. The default, if omitted, is that no encryption is
performed. If both signing and encryption are requested, the
response will be signed then encrypted, with the result being
a Nested JWT, as defined in JWT [RFC7519].

introspection_encrypted_response_enc  OPTIONAL.  JWE [RFC7516]
algorithm ("enc" value) as defined in JWA [RFC7518] for
authenticated encryption of introspection responses. The
default, if omitted, is "A128CBC-HS256". Note: This
parameter MUST NOT be specified without setting
"introspection_encrypted_response_alg".

Resource servers may register their public encryption keys using the
"jwks_uri" or "jwks" metadata parameters.

5. Authorization Server Metadata

Authorization servers SHOULD publish the supported algorithms for
signing and encrypting the JWT of an introspection response by
utilizing OAuth 2.0 Authorization Server Metadata [RFC8414]
parameters.

The following parameters are introduced by this specification:

introspection_signing_alg_values_supported  OPTIONAL.  JSON array
containing a list of the JWS [RFC7515] signing algorithms
("alg" values) as defined in JWA [RFC7518] supported by the
introspection endpoint to sign the response.

introspection_encryption_alg_values_supported  OPTIONAL.  JSON array
containing a list of the JWE [RFC7516] encryption algorithms
("alg" values) as defined in JWA [RFC7518] supported by the
introspection endpoint to encrypt the response.

introspection_encryption_enc_values_supported  OPTIONAL.  JSON array
containing a list of the JWE [RFC7516] encryption algorithms
("enc" values) as defined in JWA [RFC7518] supported by the
introspection endpoint to encrypt the response.

6. Security Considerations
6.1. Cross-JWT Confusion

JWT introspection responses and OpenID Connect ID Tokens are syntactically similar. An attacker could therefore attempt to impersonate an end-user at a OpenID Connect relying party by passing the JWT as an ID token.

Such an attack can be prevented like any other token substitution attack. The authorization server MUST include the claims "iss" and "aud" in each JWT introspection response, with the "iss" value set to the authorization server’s issuer URL and the "aud" value set to the resource server’s identifier. This allows a correctly implemented OpenID Connect relying party to detect substitution by checking the "iss" and "aud" claims as described in Section 3.1.3.7. of [OpenID.Core]. Relying parties SHOULD also use and check the "nonce" parameter and claim to prevent token and code replay.

Resource servers utilizing JRTs to represent structured access tokens could be susceptible to replay attacks. Resource servers should therefore apply proper counter measures against replay as described in [I-D.ietf-oauth-security-topics], section 2.2.

JWT Confusion and other attacks involving JWTs are discussed in [I-D.ietf-oauth-jwt-bcp].

6.2. Token Data Leakage

The authorization server MUST use Transport Layer Security (TLS) 1.2 (or higher) in order to prevent token data leakage.

If the authorization server supports unauthenticated requests an attacker could potentially retrieve token data which must be kept confidential. This attack can be prevented by either authenticating any request to the token introspection endpoint or by setting up the respective recipient for encrypted responses.

In the latter case, confidentiality is ensured by the fact that only the legitimate recipient is able to decrypt the response. An attacker could try to circumvent this measure by requesting a plain JSON response, using an Accept header with the content type set to, for example, "application/json" instead of "application/jwt". To prevent this attack the authorization server MUST NOT serve requests with content type other than "application/jwt" if the resource server is set up to receive encrypted responses (see also Section 3).
6.3. Data Minimization

The authorisation server determines the token data a resource server is allowed to see based on the resource server’s client_id and suitable token data, e.g. the scope value.

7. Acknowledgements

We would like to thank Petteri Stenius, Neil Madden, Filip Skokan, and Tony Nadalin for their valuable feedback.

8. IANA Considerations

8.1. OAuth Dynamic Client Registration Metadata Registration

This specification requests registration of the following client metadata definitions in the IANA "OAuth Dynamic Client Registration Metadata" registry [IANA.OAuth.Parameters] established by [RFC7591]:

8.1.1. Registry Contents

- Client Metadata Name: "introspection_signed_response_alg"
  - Client Metadata Description: String value indicating the client’s desired introspection response signing algorithm.
  - Change Controller: IESG
  - Specification Document(s): Section 4 of [[ this specification ]]

- Client Metadata Name: "introspection_encrypted_response_alg"
  - Client Metadata Description: String value specifying the desired introspection response encryption algorithm (alg value).
  - Change Controller: IESG
  - Specification Document(s): Section 4 of [[ this specification ]]

- Client Metadata Name: "introspection_encrypted_response_enc"
  - Client Metadata Description: String value specifying the desired introspection response encryption algorithm (enc value).
  - Change Controller: IESG
  - Specification Document(s): Section 4 of [[ this specification ]]
8.2. OAuth Authorization Server Metadata Registration

This specification requests registration of the following values in the IANA "OAuth Authorization Server Metadata" registry [IANA.OAuth.Parameters] established by [RFC8414].

8.2.1. Registry Contents

- Metadata Name: "introspection_signing_alg_values_supported"
- Metadata Description: JSON array containing a list of algorithms supported by the authorization server for introspection response signing.
- Change Controller: IESG
- Specification Document(s): Section 5 of [[ this specification ]]

- Metadata Name: "introspection_encryption_alg_values_supported"
- Metadata Description: JSON array containing a list of algorithms supported by the authorization server for introspection response encryption (alg value).
- Change Controller: IESG
- Specification Document(s): Section 5 of [[ this specification ]]

- Metadata Name: "introspection_encryption_enc_values_supported"
- Metadata Description: JSON array containing a list of algorithms supported by the authorization server for introspection response encryption (enc value).
- Change Controller: IESG
- Specification Document(s): Section 5 of [[ this specification ]]

8.3. OAuth Token Introspection Response

This specification requests registration of the following claim values as defined in [OpenID.Core], Section 5.1, in the IANA "OAuth Token Introspection Response" registry. [IANA.OAuth.Parameters] established by [RFC7662].
8.3.1. Registry Contents

- Name: "name"
  - Description: End-User’s full name in displayable form including all name parts, possibly including titles and suffixes, ordered according to the End-User’s locale and preferences.
  - Change Controller: IESG
  - Specification Document(s): [OpenID.Core], Section 5.1

- Name: "given_name"
  - Description: Given name(s) or first name(s) of the End-User. Note that in some cultures, people can have multiple given names; all can be present, with the names being separated by space characters.
  - Change Controller: IESG
  - Specification Document(s): [OpenID.Core], Section 5.1

- Name: "family_name"
  - Description: Surname(s) or last name(s) of the End-User. Note that in some cultures, people can have multiple family names or no family name; all can be present, with the names being separated by space characters.
  - Change Controller: IESG
  - Specification Document(s): [OpenID.Core], Section 5.1

- Name: "middle_name"
  - Description: Middle name(s) of the End-User. Note that in some cultures, people can have multiple middle names; all can be present, with the names being separated by space characters. Also note that in some cultures, middle names are not used.
  - Change Controller: IESG
  - Specification Document(s): [OpenID.Core], Section 5.1

- Name: "nickname"
- **Description**: Casual name of the End-User that may or may not be the same as the `given_name`. For instance, a nickname value of Mike might be returned alongside a `given_name` value of Michael.

- **Change Controller**: IESG

- **Specification Document(s)**: [OpenID.Core], Section 5.1

- **Name**: "preferred_username"

- **Description**: Shorthand name by which the End-User wishes to be referred to at the RP, such as janedoe or j.doe. This value MAY be any valid JSON string including special characters such as @, /, or whitespace.

- **Change Controller**: IESG

- **Specification Document(s)**: [OpenID.Core], Section 5.1

- **Name**: "profile"

- **Description**: URL of the End-User’s profile page. The contents of this Web page SHOULD be about the End-User.

- **Change Controller**: IESG

- **Specification Document(s)**: [OpenID.Core], Section 5.1

- **Name**: "picture"

- **Description**: URL of the End-User’s profile picture. This URL MUST refer to an image file (for example, a PNG, JPEG, or GIF image file), rather than to a Web page containing an image. Note that this URL SHOULD specifically reference a profile photo of the End-User suitable for displaying when describing the End-User, rather than an arbitrary photo taken by the End-User.

- **Change Controller**: IESG

- **Specification Document(s)**: [OpenID.Core], Section 5.1

- **Name**: "website"

- **Description**: URL of the End-User’s Web page or blog. This Web page SHOULD contain information published by the End-User or an organization that the End-User is affiliated with.

- **Change Controller**: IESG
- Specification Document(s): [OpenID.Core], Section 5.1
  - Name: "email"
    - Description: End-User’s preferred e-mail address. Its value MUST conform to the RFC 5322 [RFC5322] addr-spec syntax.
  - Change Controller: IESG

- Specification Document(s): [OpenID.Core], Section 5.1
  - Name: "email_verified"
    - Description: True if the End-User’s e-mail address has been verified; otherwise false. When this Claim Value is true, this means that the OP took affirmative steps to ensure that this e-mail address was controlled by the End-User at the time the verification was performed. The means by which an e-mail address is verified is context-specific, and dependent upon the trust framework or contractual agreements within which the parties are operating.
  - Change Controller: IESG

- Specification Document(s): [OpenID.Core], Section 5.1
  - Name: "gender"
    - Description: End-User’s gender. Values defined by this specification are female and male. Other values MAY be used when neither of the defined values are applicable.
  - Change Controller: IESG

- Specification Document(s): [OpenID.Core], Section 5.1
  - Name: "birthdate"
    - Description: Time the End-User’s information was last updated. Its value is a JSON number representing the number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the date/time.
  - Change Controller: IESG

- Specification Document(s): [OpenID.Core], Section 5.1
  - Name: "zoneinfo"
- Description: String from zoneinfo [zoneinfo] time zone database representing the End-User’s time zone. For example, Europe/Paris or America/Los_Angeles.

- Change Controller: IESG

- Specification Document(s): [OpenID.Core], Section 5.1

- Name: "locale"

- Description: Time the End-User’s information was last updated. Its value is a JSON number representing the number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the date/time.

- Change Controller: IESG

- Specification Document(s): [OpenID.Core], Section 5.1

- Name: "phone_number"

- Description: End-User’s preferred telephone number. E.164 [E.164] is RECOMMENDED as the format of this Claim, for example, +1 (425) 555-1212 or +56 (2) 687 2400. If the phone number contains an extension, it is RECOMMENDED that the extension be represented using the RFC 3966 [RFC3966] extension syntax, for example, +1 (604) 555-1234;ext=5678.

- Change Controller: IESG

- Specification Document(s): [OpenID.Core], Section 5.1

- Name: "phone_number_verified"

- Description: True if the End-User’s phone number has been verified; otherwise false. When this Claim Value is true, this means that the OP took affirmative steps to ensure that this phone number was controlled by the End-User at the time the verification was performed. The means by which a phone number is verified is context-specific, and dependent upon the trust framework or contractual agreements within which the parties are operating. When true, the phone_number Claim MUST be in E.164 format and any extensions MUST be represented in RFC 3966 format.

- Change Controller: IESG

- Specification Document(s): [OpenID.Core], Section 5.1

- Name: "address"
Description: End-User’s preferred postal address. The value of the address member is a JSON [RFC4627] structure containing some or all of the members defined in [OpenID.Core], Section 5.1.1.

Change Controller: IESG

Specification Document(s): [OpenID.Core], Section 5.1

Name: "updated_at"

Description: Time the End-User’s information was last updated. Its value is a JSON number representing the number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the date/time.

Change Controller: IESG

Specification Document(s): [OpenID.Core], Section 5.1

9. References

9.1. Normative References

[I-D.ietf-oauth-jwt-bcp]

[I-D.ietf-oauth-security-topics]

[OpenID.Core]

[OpenID.Registration]


9.2. Informative References

Appendix A. Document History

[[ To be removed from the final specification ]]

-03

- added registration for OpenID Connect Standard Claims to OAuth Token Introspection Response registry

-02

- updated references

-01

- adapted wording to preclude any accept header except "application/jwt" if encrypted responses are required

- use registered alg value RS256 for default signing algorithm

- added text on claims in the token introspection response

-00

- initial version of the WG draft

- defined default signing algorithm

- changed behavior in case resource server is set up for encryption

- Added text on token data leakage prevention to the security considerations

- moved Security Considerations section forward

WG draft

-01

- fixed typos in client meta data field names

- added OAuth Server Metadata parameters to publish algorithms supported for signing and encrypting the introspection response

- added registration of new parameters for OAuth Server Metadata and Client Registration

- added explicit request for JWT introspection response
- made iss and aud claims mandatory in introspection response
- Stylistic and clarifying edits, updates references

initial version

Authors’ Addresses

Torsten Lodderstedt (editor)
yes.com AG
Email: torsten@lodderstedt.net

Vladimir Dzhuvinov
Connect2id Ltd.
Email: vladimir@connect2id.com