Stream: Internet Engineering Task Force (IETF)
RFC: 9701
Category: Standards Track
Published: January 2025
ISSN: 2070-1721
Authors: T. Lodderstedt, Ed. *yes.com AG* V. Dzhuvinov *Connect2id Ltd.*

RFC 9701
JSON Web Token (JWT) Response for OAuth Token Introspection

Abstract

This specification proposes an additional response secured by JSON Web Token (JWT) for OAuth 2.0 Token Introspection.

Status of This Memo

Copyright Notice

Table of Contents

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 enables deployments to implement opaque 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 token introspection response for an access token, including cases where the authorization server assumes liability for the content of the token introspection response. An example is a resource server using verified personal 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 [RFC7519] as the introspection response. This specification extends the token introspection endpoint with the capability to return responses as JWTs.

2. Requirements Notation

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Resource Server Management

The authorization server (AS) and the resource server (RS) maintain a strong, two-way trust relationship. The resource server relies on the authorization server to obtain authorization, user, and other data as input to its access control decisions and service delivery. The authorization server relies on the resource server to handle the provided data appropriately.

In the context of this specification, the token introspection endpoint is used to convey such security data and potentially also privacy-sensitive data related to an access token.

In order to process the introspection requests in a secure and privacy-preserving manner, the authorization server **MUST** be able to identify, authenticate, and authorize resource servers.

The AS **MAY** additionally encrypt the token introspection response JWTs. If encryption is used, the AS is provisioned with encryption keys and algorithms for the RS.

The AS **MUST** be able to determine whether an RS is the audience for a particular access token and what data it is entitled to receive; otherwise, the RS is not authorized to obtain data for the access token. The AS has the discretion of how to fulfill this requirement. The AS could, for example, maintain a mapping between scope values and RSs.

The requirements given above imply that the AS maintains credentials and other configuration data for each RS.

One way is by utilizing dynamic client registration [RFC7591] and treating every RS as an OAuth client. In this case, the AS is assumed to at least maintain a "client_id" and a "token_endpoint_auth_method" with complementary authentication method metadata, such as "jwks" or "client_secret". In cases where the AS needs to acquire consent to transmit data to an RS, the following client metadata fields are recommended: "client_name", "client_uri", "contacts", "tos_uri", and "policy_uri".

The AS **MUST** restrict the use of client credentials by an RS to the calls it requires, e.g., the AS **MAY** restrict such a client to call the token introspection endpoint only. How the AS implements this restriction is beyond the scope of this specification.

This specification further introduces client metadata to manage the configuration options required to sign and encrypt token introspection response JWTs.

4. Requesting a JWT Response

An RS requests a JWT introspection response by sending an introspection request with an `Accept` HTTP header field set to "application/token-introspection+jwt".

The AS **MUST** authenticate the caller at the token introspection endpoint. Authentication can utilize client authentication methods or a separate access token that is issued to the RS and identifies the RS as the subject.

The following is a non-normative example request, with the RS authenticating with a private key JWT:

```
POST /introspect HTTP/1.1
Host: as.example.com
Accept: application/token-introspection+jwt
Content-Type: application/x-www-form-urlencoded

token=2YotnFZFEjr1zCsicMWpAA&
client_assertion_type=
 urn%3Aietf%3Aparams%3Aoauth%3Aclient-assertion-type%3Ajwt-bearer&
 client_assertion=PHNhbWxwOl[...omitted for brevity...]ZT
```

5. JWT Response

The introspection endpoint responds with a JWT, setting the `Content-Type` HTTP header field to "application/token-introspection+jwt" and the JWT `typ` ("type") header parameter to "token-introspection+jwt".

The JWT **MUST** include the following top-level claims:

iss
: **MUST** be set to the issuer URL of the authorization server.

aud
: **MUST** identify the resource server receiving the token introspection response.

iat
: **MUST** be set to the time when the introspection response was created by the authorization server

token_introspection
: A JSON object containing the members of the token introspection response, as specified in [RFC7662], Section 2.2. The separation of the introspection response members into a dedicated JSON object containing a JWT claim is intended to prevent conflict and confusion with top-level JWT claims that may bear the same name.

  If the access token is invalid, expired, revoked, or not intended for the calling resource server (audience), the authorization server **MUST** set the value of the `active` member in the `token_introspection` claim to `false` and **MUST NOT** include other members. Otherwise, the `active` member is set to `true`.

  The AS **SHOULD** narrow down the `scope` value to the scopes relevant to the particular RS.

  As specified in Section 2.2 of [RFC7662], implementations **MAY** extend the token introspection response with service-specific claims. In the context of this specification, such claims will be added as top-level members of the `token_introspection` claim.

  Token introspection response parameter names intended to be used across domains **MUST** be registered in the "OAuth Token Introspection Response" registry [IANA.OAuth.Token.Introspection] defined by [RFC7662].

  When the AS acts as a provider of resource owner identity claims to the RS, the AS determines, based on its RS-specific policy, what identity claims to return in the token introspection response. The AS **MUST** ensure the release of any privacy-sensitive data is legally based (see Section 9).

  Further content of the introspection response is determined by the RS-specific policy at the AS.

The JWT **MAY** include other claims, including those from the "JSON Web Token Claims" registry established by [RFC7519]. The JWT **SHOULD NOT** include the `sub` and `exp` claims, as an additional measure to prevent misuse of the JWT as an access token (see Section 8.1).

Note: Although the JWT format is widely used as an access token format, the JWT returned in the introspection response is not an alternative representation of the introspected access token and is not intended to be used as an access token.

This specification registers the "application/token-introspection+jwt" media type, which is used as the value of the `typ` ("type") header parameter of the JWT to indicate that the payload is a token introspection response.

The JWT is cryptographically secured as specified in [RFC7519].

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: An AS compliant with this specification **MUST** refuse to serve introspection requests that don't authenticate the caller and return an HTTP status code 400. This is done to ensure token data is released to legitimate recipients only and prevent downgrading to [RFC7662] behavior (see Section 8.2).

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

```
HTTP/1.1 200 OK
Content-Type: application/token-introspection+jwt

eyJraWQiOiJ3RzZEIiwidHlwIjoidG9rZW4taW50cm9zcGVjdGlvbitqd3QiLCJhbGc
iOiJSUzI1NiJ9.eyJpc3MiOiJodHRwczovL2FzLmV4YW1wbGUuY29tLyIsImF1ZCI6I
mh0dHBzOi8vcnMuZXhhbXBsZS5jb20vcmVzb3VyY2UiLCJpYXQiOjE1MTQ3OTc4OTIs
InRva2VuX2ludHJvc3BlY3Rpb24iOnsiYWN0aXZlIjp0cnVlLCJpc3MiOiJodHRwczo
vL2FzLmV4YW1wbGUuY29tLyIsImF1ZCI6Imh0dHBzOi8vcnMuZXhhbXBsZS5jb20vcm
Vzb3VyY2UiLCJpYXQiOjE1MTQ3OTc4MjIsImV4cCI6MTUxNDc5Nzk0MiwiY2xpZW50X
2lkIjoicGFpQjJnb28wYSIsInNjb3BlIjoicmVhZCB3cml0ZSBkb2xwaGluIiwic3Vi
IjoiWjVPM3VwUEM4OFFyQWp4MDBkaXMiLCJiaXJ0aGRhdGUiOiIxOTgyLTAyLTAxIiw
iZ2l2ZW5fbmFtZSI6IkpvaG4iLCJmYW1pbHlfbmFtZSI6IkRvZSIsImp0aSI6InQxRm
9DQ2FaZDRYdjRPUkpVV1ZVZVRaZnNLaFczMENRQ3JXRERqd1h5NncifX0.przJMU5Gh
mNzvwtt1Sr-xa9xTkpiAg5IshbQsRiRVP_7eGR1GHYrNwQh84kxOkHCyje2g5WSRcYo
sGEVIiC-eoPJJ-qBwqwSlgx9JEeCDw2W5DjrblOI_N0Jvsq_dUeOyoWVMqlOydOBhKN
Y0smBrI4NZvEExucOm9WUJXMuJtvq1gBes-0go5j4TEv9sOP9uu81gqWTr_LOo6pgT0
tFFyZfWC4kbXPXiQ2YT6mxCiQRRNM-l9cBdF6Jx6IOrsfFhBuYdYQ_mlL19HgDDOFal
eyqmru6lKlASOsaE8dmLSeKcX91FbG79FKN8un24iwIDCbKT9xlUFl54xWVShNDFA
```

The example response JWT header contains the following JSON document:

```
{
  "typ": "token-introspection+jwt",
  "alg": "RS256",
  "kid": "wG6D"
}
```

The example response JWT payload contains the following JSON document:

```
{
  "iss":"https://as.example.com/",
  "aud":"https://rs.example.com/resource",
  "iat":1514797892,
  "token_introspection":
     {
        "active":true,
        "iss":"https://as.example.com/",
        "aud":"https://rs.example.com/resource",
        "iat":1514797822,
        "exp":1514797942,
        "client_id":"paiB2goo0a",
        "scope":"read write dolphin",
        "sub":"Z5O3upPC88QrAjx00dis",
        "birthdate":"1982-02-01",
        "given_name":"John",
        "family_name":"Doe",
        "jti":"t1FoCCaZd4Xv4ORJUWVUeTZfsKhW30CQCrWDDjwXy6w"
     }
}
```

6. Client Metadata

The authorization server determines the algorithm 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 [RFC7591] with the resource server acting as a client, as specified below.

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**. "JSON Web Signature (JWS)" [RFC7515] algorithm (`alg` value), as defined in "JSON Web Algorithms (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**. "JSON Web Encryption (JWE)" [RFC7516] algorithm (`alg` value), as defined in JWA [RFC7518], for content key encryption. If this is specified, the response will be encrypted using JWE and the configured content encryption algorithm (`introspection_encrypted_response_enc`). 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 content 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.

7. 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. Resource servers use this data to parametrize their client registration requests.

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 content encryption key for introspection responses (content key encryption).

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 (content encryption).

8. Security Considerations

8.1. Cross-JWT Confusion

The `iss` and potentially the `aud` claim of a token introspection JWT can resemble those of a JWT-encoded access token. An attacker could try to exploit this and pass a JWT token introspection response as an access token to the resource server. The `typ` ("type") JWT header "token-introspection+jwt" and the encapsulation of the token introspection members, such as `sub` and `scope` in the `token_introspection` claim, are intended to prevent such substitution attacks. Resource servers **MUST** therefore check the `typ` JWT header value of received JWT-encoded access tokens and ensure all minimally required claims for a valid access token are present.

Resource servers **MUST** additionally apply the countermeasures against access token replay, as described in [RFC9700].

JWT confusion and other attacks involving JWTs are discussed in [RFC8725].

8.2. Token Data Leakage

The authorization server **MUST** use Transport Layer Security (TLS) 1.2 (or higher), per BCP 195 [RFC9325], in order to prevent token data leakage.

Section 2.1 of [RFC7662] permits requests to the introspection endpoint to be authorized with an access token that doesn't identify the caller. To prevent introspection of tokens by parties that are not the intended consumer, the authorization server **MUST** require all requests to the token introspection endpoint to be authenticated.

9. Privacy Considerations

The token introspection response can be used to transfer personal identifiable information (PII) from the AS to the RS. The AS **MUST** conform to legal and jurisdictional constraints for the data transfer before any data is released to a particular RS. The details and determining of these constraints vary by jurisdiction and are outside the scope of this document.

A commonly found way to establish the legal basis for releasing PII is by explicit user consent gathered from the resource owner by the AS during the authorization flow.

It is also possible that the legal basis is established out of band, for example, in an explicit contract or by the client gathering the resource owner's consent.

If the AS and the RS belong to the same legal entity (1st party scenario), there is potentially no need for an explicit user consent, but the terms of service and policy of the respective service provider **MUST** be enforced at all times.

In any case, the AS **MUST** ensure that the scope of the legal basis is enforced throughout the whole process. The AS **MUST** retain the scope of the legal basis with the access token, e.g., in the scope value, it **MUST** authenticate the RS, and the AS **MUST** determine the data an RS is allowed to receive based on the RS's identity and suitable token data, e.g., the scope value.

Implementers should be aware that a token introspection request lets the AS know when the client (and potentially the user) is accessing the RS, which is also an indication of when the user is using the client. If this implication is not acceptable, implementers **MUST** use other means to relay access token data, for example, by directly transferring the data needed by the RS within the access token.

10. IANA Considerations

10.1. OAuth Dynamic Client Registration Metadata Registration

The following client metadata definitions have been registered in the IANA "OAuth Dynamic Client Registration Metadata" registry [IANA.OAuth.Parameters] established by [RFC7591]:

10.1.1. 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: IETF
Reference: Section 6 of RFC 9701

Client Metadata Name: `introspection_encrypted_response_alg`
Client Metadata Description: String value specifying the desired introspection response content key encryption algorithm (alg value)
Change Controller: IETF
Reference: Section 6 of RFC 9701

Client Metadata Name: `introspection_encrypted_response_enc`
Client Metadata Description: String value specifying the desired introspection response content encryption algorithm (enc value)
Change Controller: IETF
Reference: Section 6 of RFC 9701

10.2. OAuth Authorization Server Metadata Registration

The following values have been registered in the IANA "OAuth Authorization Server Metadata" registry [IANA.OAuth.Parameters] established by [RFC8414].

10.2.1. 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: IETF
Reference: Section 7 of RFC 9701

Metadata Name: `introspection_encryption_alg_values_supported`
Metadata Description: JSON array containing a list of algorithms supported by the authorization server for introspection response content key encryption (alg value)
Change Controller: IETF
Reference: Section 7 of RFC 9701

Metadata Name: `introspection_encryption_enc_values_supported`
Metadata Description: JSON array containing a list of algorithms supported by the authorization server for introspection response content encryption (enc value)
Change Controller: IETF

Reference: Section 7 of RFC 9701

10.3. Media Type Registration

The "application/token-introspection+jwt" media type has been registered in the "Media Types" registry [IANA.MediaTypes] in the manner described in [RFC6838]. It can be used to indicate that the content is a token introspection response in JWT format.

10.3.1. Registry Contents

Type name: application

Subtype name: token-introspection+jwt

Required parameters: N/A

Optional parameters: N/A

Encoding considerations: binary. A token introspection response is a JWT; JWT values are encoded as a series of base64url-encoded values (with trailing '=' characters removed), some of which may be the empty string, separated by period ('.') characters.

Security considerations: See Section 8 of RFC 9701

Interoperability considerations: N/A

Published specification: Section 4 of RFC 9701

Applications that use this media type: Applications that produce and consume OAuth Token Introspection Responses in JWT format

Fragment identifier considerations: N/A

Additional information:
Magic number(s): N/A
File extension(s): N/A
Macintosh file type code(s): N/A

Person & email address to contact for further information:
Torsten Lodderstedt (torsten@lodderstedt.net)

Intended usage: COMMON

Restrictions on usage: none

Author: Torsten Lodderstedt (torsten@lodderstedt.net)

Change controller: IETF

10.4. JWT Claim Registration

The "token_introspection" claim has been registered in the "JSON Web Token (JWT)" registry [IANA.JWT] in the manner described in [RFC7519].

10.4.1. Registry Contents

Claim Name: token_introspection
Claim Description: Token introspection response
Change Controller: IETF
Reference: Section 5 of RFC 9701

11. References

11.1. Normative References

[IANA.JWT] IANA, "JSON Web Token (JWT) Claims", <https://www.iana.org/assignments/jwt>.

[IANA.MediaTypes] IANA, "Media Types", <http://www.iana.org/assignments/media-types>.

[IANA.OAuth.Token.Introspection] IANA, "OAuth Token Introspection Response", <https://www.iana.org/assignments/oauth-parameters>.

[OpenID.Registration] Sakimura, N., Bradley, J., and M. Jones, "OpenID Connect Dynamic Client Registration 1.0 incorporating errata set 1", November 2014, <https://openid.net/specs/openid-connect-registration-1_0.html>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, <https://www.rfc-editor.org/info/rfc6838>.

[RFC7515] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Signature (JWS)", RFC 7515, DOI 10.17487/RFC7515, May 2015, <https://www.rfc-editor.org/info/rfc7515>.

[RFC7516] Jones, M. and J. Hildebrand, "JSON Web Encryption (JWE)", RFC 7516, DOI 10.17487/RFC7516, May 2015, <https://www.rfc-editor.org/info/rfc7516>.

[RFC7518] Jones, M., "JSON Web Algorithms (JWA)", RFC 7518, DOI 10.17487/RFC7518, May 2015, <https://www.rfc-editor.org/info/rfc7518>.

[RFC7519] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", RFC 7519, DOI 10.17487/RFC7519, May 2015, <https://www.rfc-editor.org/info/rfc7519>.

[RFC7591] Richer, J., Ed., Jones, M., Bradley, J., Machulak, M., and P. Hunt, "OAuth 2.0 Dynamic Client Registration Protocol", RFC 7591, DOI 10.17487/RFC7591, July 2015, <https://www.rfc-editor.org/info/rfc7591>.

[RFC7662] Richer, J., Ed., "OAuth 2.0 Token Introspection", RFC 7662, DOI 10.17487/RFC7662, October 2015, <https://www.rfc-editor.org/info/rfc7662>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8414] Jones, M., Sakimura, N., and J. Bradley, "OAuth 2.0 Authorization Server Metadata", RFC 8414, DOI 10.17487/RFC8414, June 2018, <https://www.rfc-editor.org/info/rfc8414>.

[RFC8725] Sheffer, Y., Hardt, D., and M. Jones, "JSON Web Token Best Current Practices", BCP 225, RFC 8725, DOI 10.17487/RFC8725, February 2020, <https://www.rfc-editor.org/info/rfc8725>.

[RFC9325] Sheffer, Y., Saint-Andre, P., and T. Fossati, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 9325, DOI 10.17487/RFC9325, November 2022, <https://www.rfc-editor.org/info/rfc9325>.

[RFC9700] Lodderstedt, T., Bradley, J., Labunets, A., and D. Fett, "Best Current Practice for OAuth 2.0 Security", BCP 240, RFC 9700, DOI 10.17487/RFC9700, January 2025, <https://www.rfc-editor.org/info/rfc9700>.

11.2. Informative References

[IANA.OAuth.Parameters] IANA, "OAuth Parameters", <http://www.iana.org/assignments/oauth-parameters>.

Acknowledgements

We would like to thank Petteri Stenius, Neil Madden, Filip Skokan, Tony Nadalin, Remco Schaar, Justin Richer, Takahiko Kawasaki, Benjamin Kaduk, Robert Wilton, and Roman Danyliw for their valuable feedback.

Authors' Addresses

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

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