B. de Medeiros, Ed.
M. Scurtescu
Google
P. Tarjan
Facebook

September 22, 2011

OAuth2 Multiple Response Type Encoding Practices

Abstract

This specification aims to provide guidance on proper encoding of responses to OAuth2 Authorization requests, where the request specifies a response type including space characters.

This specification also serves as the registration document for several specific new response types, in accordance with the stipulations of the OAuth Parameters Registry.

Table of Contents

---

1. Requirements Notation and Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in **[RFC2119]** .

Throughout this document, values are quoted to indicate that they are to be taken literally. When using these values in protocol messages, the quotes MUST NOT be used as part of the value.

---

2. Terminology

The following definitions define additional terminology used in this specification in addition to those defined in **OAuth 2.0** [OAuth.2.0].

Client and Server
: In the traditional client-server authentication model, the client requests an access restricted resource (protected resource) on the server by authenticating with the server using the resource owner's credentials.

Authorization Endpoint
: A web resource mantained by the server, and used to obtain authorization (grant) from the resource owner via user-agent redirection.

Response Type
: The client informs the authorization server of the desired authorization processing flow using the parameter response_type.

Authorization Endpoint Response Type Registry
: Process established by the OAuth2 specification for the registration of new response_type parameters.

Multiple-valued Response Types
: The OAuth2 specification allows for registration of space-separated response_type values. If a response type contains one of more space characters (%20), it is compared as a space-delimited list of values in which the order of values does not matter.

TOC

3. Encoding Multiple-valued Response Types

This specification does not provide guidance if, in a request, response_type includes a value that requires the server to return data partially encoded in the query string and partially encoded in the fragment.

Otherwise, if a multiple-valued response type is defined, then it is RECOMMENDED that the following encoding rules be applied for the issued response.

If, in a request, response_type includes only values that require the server to return data fully encoded within the query string then the returned data in the response for this multiple-valued response_type MUST be fully encoded within the query string. This recommendation applies to both success and error responses.

If, in a request, response_type includes any value that requires the server to return data fully encoded within the fragment then the returned data in the response for this multiple-valued response_type MUST be fully encoded within the fragment. This recommendation applies to both success and error responses.

Rationale: Whenever response_type values include fragment-encoded parts, a user-agent client component must be involved to complete processing of the response. If a new query parameter is added to the client URI, it will cause the user-agent to re-fetch the client URI, causing discontinuity of operation of the user-agent-based client components. If instead only fragment encoding is used, the user-agent will simply re-active the client component, at which time this component may process the fragment and also convey any parameters to a client host as necessary, e.g., via XmlHttpRequest. Therefore, full fragment encoding always results in lower latency for response processing.

4. Id Token Response Type

TOC

This section registers a new response type, the id_token, in accordance with the stipulations in the OAuth2 specification, Section 8.4. The intended purpose of the id_token is that it MUST provide an assertion of the identity of the resource owner as understood by the server. The assertion MUST specify a targeted audience, e.g. the requesting client. However, the specific semantics of the assertion and how it can be validated are not specified in this document.

id_token
: If supplied as the response_type parameter in an OAuth2 Authorization Request, a successful response should include the parameter id_token encoded in the fragment of the response URI.

Returning the id_token in a fragment reduces the likelihood that the id_token leaks during transport and mitigates the associated risks to the privacy of the user (resource owner).

5. None Response Type

TOC

This section registers the response type none, in accordance with the stipulations in the OAuth2 specification, Section 8.4. The intended purpose is to enable use cases where a party requests the server to register a grant of access to a protected resource on behalf of a client but requires no access credentials to be returned to the client at that time. The means by which the client eventually obtains the access credentials is left unspecified here.

One scenario is where a user wishes to purchase an application from a market, and desires to authorize application installation and grant the application access to protected resources in a single step. However, since the user is not presently interacting with the (not yet active) application, it is not appropriate to return access credentials simultaneously to the authorization step.

none
: When supplied as the response_type parameter in an OAuth2 authorization request, the Authorization server SHOULD NOT return an OAuth2 code nor a token in a succesful response to the grant request. If a redirect_uri is supplied, the user-agent SHOULD be redirected there after granting or denying access. The request MAY include a state parameter, and if so the server MUST echo its value by adding it to the redirect_uri when issuing a successful response. Any parameters added to the redirect_uri should be query encoded. This applies to both successful responses or error responses.

The response type none SHOULD NOT be combined with other response types.

6. Registration of Some Multiple-Valued Response Type Combinations

TOC

This section registers combinations of the values code, token, and id_token, which are

each individually registered response types.

code token
: When supplied as the value for the response_type parameter, a successful response MUST include both an access token and an authorization code as defined in the OAuth2 specification. Both successful and error responses SHOULD be fragment-encoded.

code id_token
: When supplied as the value for the response_type parameter, a successful response MUST include both an authorization code as well as an id_token. Both success and error responses SHOULD be fragment-encoded.

id_token token
: When supplied as the value for the response_type parameter, a successful response MUST include both an access token as well as an id_token. Both success and error responses SHOULD be fragment-encoded.

code id_token token
: When supplied as the value for the response_type parameter, a successful response MUST include an authorization code, an id_token, and an access token. Both success and error responses SHOULD be fragment-encoded.

A non-normative request/response example as issued/received by the user-agent:

```
GET https://server.example.com/authorize?
response_type=token%20id_token
&client_id=s6BhdRkqt3
&redirect_uri=https%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb
&state=af0ifjsldkj HTTP/1.1



HTTP/1.1 302 Found
Location: https://client.example.com/#
access_token=SlAV32hkKG&
token_type=bearer&
id_token=1234567.SlAV32hkKG.abcde1234&
expires_in=3600&
&state=af0ifjsldkj
```

---

7. Normative References

TOC

[OAuth.2.0] Hammer-Lahav, E., Ed., Recordon, D., and D. Hardt, "**OAuth 2.0 Authorization Protocol**," July 2011.

[RFC2119] **Bradner, S.**, "**Key words for use in RFCs to Indicate Requirement Levels**," BCP 14, RFC 2119, March 1997 (**TXT**, **HTML**, **XML**).

[RFC2616] **Fielding, R.**, **Gettys, J.**, **Mogul, J.**, **Frystyk, H.**, **Masinter, L.**, **Leach, P.**, and **T. Berners-Lee**, "**Hypertext Transfer Protocol -- HTTP/1.1**," RFC 2616, June 1999 (**TXT**, **PS**, **PDF**, **HTML**, **XML**).

[RFC3339] **Klyne, G., Ed.** and **C. Newman**, "**Date and Time on the Internet: Timestamps**," RFC 3339, July 2002 (**TXT**, **HTML**, **XML**).

[W3C.REC-html401-19991224] Raggett, D., Jacobs, I., and A. Hors, "**HTML 4.01 Specification**," World Wide Web Consortium Recommendation REC-html401-19991224, December 1999 (**HTML**).

Appendix A. Acknowledgements

TOC

The OpenID Community would like to thank the following people for the work they've done in the drafting and editing of this specification.

Naveen Agarwal (naa@google.com), Google

Breno de Medeiros (breno@google.com), Google

David Recordon (dr@fb.com), Facebook

Marius Scurtescu (mscurtescu@google.com), Google

Paul Tarjan (pt@fb.com), Facebook

Appendix B. Document History

TOC

[[To be removed from the final specification]]

-01

- Initial draft

Authors' Addresses

TOC

Breno (editor)
Google, Inc.
Email: breno@google.com

Marius
Google, Inc.
Email: mscurtescu@google.com

Paul
Facebook
Email: pt@fb.com