OAuth Response Metadata
draft-sakimura-oauth-meta-07

Abstract

This specification defines an extensible metadata that may be inserted into the OAuth 2.0 responses to assist the clients to process those responses. It is expressed either as a link header, or query parameters. It will allow the client to learn where the members in the response could be used, what is the characteristics of the payload is, how it should be processed, and so on. Since they are just additional response header/query parameters, any client that does not understand this extension should not break and work normally while supporting clients can utilize the metadata to take the advantage of the extension.

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 http://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 August 15, 2016.

Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.
1. Introduction

Although OAuth 2.0 [RFC6749] has been known for its REST friendliness, OAuth itself is not RESTful, as it heavily relies on out-of-band information to drive the interactions. This situation can be eased by hypertext-enabling the endpoint responses through the introduction of data structure that represents such hypertext and other metadata.

Hyper-text enabling the OAuth responses has many advantages. For example,

- The protected resource can tell which authorization servers it supports.
- Permissioned resource discovery: It is possible to tell the client which resource endpoint it should use. This has a privacy

advantage. The location of the resource by itself may be a sensitive information as its location may reveal information about the resource owner. Therefore, it may be sensible to tell the location only after the user consent.

- It is possible to give a hint on the processing of the payload.
- It will be resistant to IdP Mix-up attack.
- It will be resistant to Code Phishing Attack.

This specification defines methods to represent such metadata in the authorization and token responses.

1.1. 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 RFC 2119 [RFC2119].

1.2. terminology


2. Resource Endpoint Response

Resource Endpoints that implement this specification returns the following link relation (rel) and the corresponding URI value as defined in [RFC5988] in the response header. The response header can be returned in response to HEAD, GET, or POST request to the endpoint.

duri  The URI of the corresponding authorization server’s discovery document, from which the client can learn the server capabilities and endpoints.

auri  The URI of the corresponding Authorization Endpoint URI.

A typical example of the use of these header values are in the case of the Client accessing the protected resource without a proper credential. For example, in the case of an [RFC6750] protected resource, the unauthorized access may result in a response header that includes both WWW-Authenticate header as well as the Web Linking
header indicating either the Authorization Endpoint URI or the discovery document URI.

There is no cardinality restriction on relations put in place by [RFC5988]. Therefore, the resource can respond with multiple Authorization Endpoint URI or Discovery Document URIs from which the Client may choose the appropriate one.

HTTP/1.1 401 Unauthorized
WWW-Authenticate: Bearer realm="example"
    Link: <https://example.com/.well-known/openid-configuration>; rel="duri",
    <https://example.net/.well-known/openid-configuration>; rel="duri",
    <https://example.com/authz/>; rel="auri",
    <https://example.net/az/>; rel="auri",
    <https://example.com/payment-upon-trial-expiry>; rel="payments"

3. Token Endpoint Response

Token Endpoints that implements this specification returns the following link relation (rel) and the corresponding URI value as defined in [RFC5988] in the Access Token Response defined in [RFC6749].

ruri  Resource URI. The value of this parameter is the URI of the Resource Endpoint that the Access Token is supposed to be used at. If this value is present, the client MUST NOT send the Access Token to any other URI.

turi  Token Endpoint URI. The value of this parameter is the URI of the Token Endpoint that the Refresh Token can be sent to obtain a new Access Token. If this value is present, then the client MUST NOT send the refresh token to any other places.

Any other rels that are registered in Link Relation Type Registry defined in [RFC5988] registry can be used.

There is no cardinality restriction on relations put in place by [RFC5988]. Therefore, the Token Endpoint can respond with multiple Resource Endpoint URI or Discovery Document URIs from which the Client may choose the appropriate one.

Following is an example of an HTTPS response.
HTTP/1.1 200 OK
Link: <https://example.com/userinfo>; rel="ruri",
     <https://example.net/photostream/>; rel="ruri",
     <https://example.com/payment-upon-trial-expiry>; rel="payments"
Content-Type: application/JSON; charset=utf-8
{
    "access_token":"aCeSsToKen"
}

4. Authorization Endpoint HEAD response

Authorization Endpoints that implements this specification returns the following link relation (rel) and the corresponding URI value as defined in [RFC5988] in the response to the HEAD request.

turi  Token Endpoint URI. The value of this parameter is the URI of the Token Endpoint that the Authorization Code can be sent to obtain the Access Token.

duri  The URI of discovery document, from which the client can learn the server capabilities and endpoints.

ruri  Resource URI. The value of this parameter is the URI of the Resource Endpoint that the Access Token can be used at. If this parameter is specified, the client MUST NOT send the Access Token to any other URIs than the value of this parameter.

There is no cardinality restriction on relations put in place by [RFC5988]. Therefore, the Authorization Endpoint can respond with multiple Endpoint URIs with a same relation type from which the Client may choose the appropriate one.

Following is an example of an HTTPS response.

HTTP/1.1 200 OK
Link: <https://example.com/.well-known/openid-configuration>; rel="duri",
     <https://example.net/.well-known/openid-configuration>; rel="duri",
     <https://example.com/payment-upon-trial-expiry>; rel="payments"
Content-Type: application/JSON; charset=utf-8

5. Authorization Response

While [RFC5988] defines a useful way of conveying link relations, it cannot be utilized for a redirect based communication such as the authorization response of OAuth 2.0. This section defines a way to
return a limited set of those link relations as query parameters so that it can be conveyed over the redirection.

The authorization response of the implementation of this specification may return the following query parameter in the redirect URI.

turi  Token Endpoint URI. The value of this parameter is the URI of the Token Endpoint that the Authorization Code can be sent to obtain the Access Token. If this parameter is specified, the client MUST check that the value of turi matches exactly with the pre-registered token endpoint URI of the Authorization Server that the session recovered from the state variable points to. The client MUST NOT send the code to any other URIs than the value of this parameter.

ruri  Resource URI. The value of this parameter is the URI of the Resource Endpoint that the Access Token can be used at. If this parameter is specified, the client MUST NOT send the Access Token to any other URIs than the value of this parameter.

As long as the link relation type string does not collide with the underlying protocol parameters, they can also be specified as a query parameter. The value MUST be encoded in application/x-www-form-urlencoded.

The following is an example of such response. Line breaks are for display purposes only.

HTTP/1.1 302 Found
Location: https://client.example.com/cb?code=SplxlOBeZQQYbYS6WxSbIA&turi=https%3A%2F%2Fexample.com%2Ftoken&state=xyz

6.  IANA Considerations

6.1.  Link Type Registration

Pursuant to [RFC5988], the following link type registrations [[will be]] registered by mail to link-relations@ietf.org.

- Relation Name: turi
- Description: An OAuth 2.0 Token Endpoint specified in section 3.2 of [RFC6749].
- Reference: This specification
7. Security Considerations

7.1. Authorization Response Query Parameter Tampering by a Bad User

The query response parameters may be tampered by the man-in-the-browser. It can also be tampered by a malicious user. In general, anything that comes via the browser/user-agent can be tainted and untrusted.

This specification mandates the turi check so that tampering of turi by the malicious user will be detected. It does not mandate ruri check as the user can get the Access Token and send it to anywhere he wants anyways when it is returned to the browser.

However, other parameters are not protected. The Client MUST treat them tainted and implement its own check rules for each parameters.

To solve this "Tampering by bad user", either HMAC(concat(params)) need to be sent with them or have all of them inside the JWS.

8. Acknowledgements


9. Document History

-07

o Added note that there is no cardinality requirements so that multiple endpoints can be returned by repeating a same rel.

o Added resource endpoint response.

-06

o Removed duri description from token response as it is not needed.

o Made the processing instruction more precise.
-05

-04

-03

-02

-01

-00
10. References

10.1. Normative References


10.2. Informational References


Authors’ Addresses

Nat Sakimura
Nomura Research Institute

Email: sakimura@gmail.com

Nov Matake
GREE, Inc.

Email: nov@matake.jp
URI: http://matake.jp

Sascha Preibisch
CA Technologies

Email: Sascha.Preibisch@gmail.com