OAuth   P. Hunt, Ed.
Internet-Draft   Oracle
Intended status: Informational   July 29, 2013
Expires: January 30, 2014

OAuth 2.0 User Authentication For Client
draft-hunt-oauth-v2-user-a4c-00

Abstract

This specification defines a new OAuth2 endpoint that enables user authentication session information to be shared with client applications.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Section 4.1 of the OAuth 2.0 Authorization Framework [RFC6749] defines the "Authorization Code Grant" flow which defines a redirect flow, typically via a web browser, that enables confidential clients to obtain access and refresh tokens. As part of this flow, resource owners are authenticated via the user agent so that their consent may be obtained. This flow defines the "Authentication Code Grant" extension which enables clients to request re-authentication and makes authentication session information available to the client in a standardized format.

This document focuses on extending OAuth2 to provide authentication session information only. The specification does not define a standardized resource owner profile information API. It is assumed that other APIs such as the SCIM API could be used for this purpose. As part of the session information, a subject profile URL may optionally be provided.

This specification is meant to be an authentication only minimum profile of OpenID Foundation’s Connect [OIDC] specification. Where OpenID is intended to define a full user profile service, this specification focuses exclusively on providing authentication only and can be used in conjunction with any service provider resource service. Where possible, parameters that are the same have been made equivalent or the same. Implementers of this specification should also consider using OIDC as a standardized identity profile service.

1.1. Requirements Language

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].

2. Authenticate Code Grant

The Authentication Code Grant type is used in exactly the same manor as the Authorization Code Grant Section 4.1 [RFC6749] and has the same features and conditions. The Authorization Code Grant extends the features available by making it possible for clients to test and request re-authenticaiton and authorization as well as obtain login session information at the end of the grant flow.

2.1. Authentication Request

In addition to the parameters defined in Section 4.1.1 [RFC6749], the following additional parameters are defined:

prompt

OPTIONAL. Space delimited, case sensitive list of ASCII string values that specifies whether the Authorization Server prompts the End-User for reauthentication and consent. The defined values are:

none	The Authorization Server MUST NOT display any authentication or consent user interface pages. An error is returned if the End-User is not already authenticated or the Client does not have pre-configured consent for the requested Claims or does not fulfill other conditions for processing. This can be used as a method to check for existing authentication and/or consent.
login	The Authorization Server SHOULD prompt the End-User for reauthentication. If it cannot prompt the End-User, it MUST return an error.
consent	The Authorization Server SHOULD prompt the End-User for consent before returning information to the Client.
select_account	The Authorization Server SHOULD prompt the End-User to select a user account. This allows an End-User who has multiple accounts at the Authorization Server to select amongst the multiple accounts that they might have current sessions for.

If it cannot prompt the End-User, it MUST return an error.

display OPTIONAL. ASCII string value that specifies how the Authorization Server displays the authentication and consent user interface pages to the End-User. The defined values are:

page	The Authorization Server SHOULD display authentication and consent UI consistent with a full User-Agent page view. If the display parameter is not specified this is the default display mode.
popup	The Authorization Server SHOULD display authentication and consent UI consistent with a popup User-Agent window. The popup User-Agent window SHOULD be 450 pixels wide and 500 pixels tall.
touch	The Authorization Server SHOULD display authentication and consent UI consistent with a device that leverages a touch interface. The Authorization Server MAY attempt to detect the touch device and further customize the interface.
wap	The Authorization Server SHOULD display authentication and consent UI consistent with a "feature phone" type display.

hint

OPTIONAL. A helpful text message that should be displayed to the user during a re-authentication or re-authorization process.

For xample, the client directs the user-agent to make the following HTTP request using TLS (with extra line breaks for display purposes only):

```
 GET /authenticate?
 response_type=code
 &client_id=s6BhdRkqt3
 &redirect_uri=https%3A%2F%2Fclient.example.com%2Fcb
 &state=af0ifjsldkj
 &prompt=login
 Host: server.example.com
```

The authorization server MUST:

- Perform the normal OAuth2 authorization process,
- MAY elect not to request consent if no access token is to be issued (i.e. this is an authentication only request),
- MUST re-authenticate the user if "prompt" contains the parameter "login",
- MUST obtain consent from the user if "prompt" contains the parameter "consent", and,
- MUST return an error if "prompt" contains "none" and the user is not currently authenticated.

2.2. Authentication Response

The response is identical to the one described in Section 4.1.2 [RFC6749].

2.2.1. Error Responses

In addition to those defined in Section 4.1.2.1 [RFC6749], an additional "error" type is defined. "unauthenticated_user", MUST be returned after an authentication request parameter "prompt" is provided containing value "none" and the user is found to be currently unauthenticated.

2.3. Access Token Request

The access token request is identical to the one described in Section 4.1.3 [RFC6749]. In cases where there is no associated resource API and an access token is not to be issued, the normal OAuth2 token request is still made.

2.4. Access Token Response

If the access token request is valid and authorized, the authorization server issues an access token and optional refresh token as described in Section 5.1 [RFC6749]with the exception that the issuance of access_token is OPTIONAL. If the request client authentication failed or is invalid, the authorization server returns an error response as described in Section 5.2.

In addition to the parameters described in Section 5, a new "session" parameter or "session_token" is returned containing the following possible claims:

sub	REQUIRED. An identifier for the authenicated subject. The same identifier MUST be return for the same authenticated user on the same client_id. The authenticated user’s "sub" value MAY change for different client_id values.
sub_url	OPTIONAL. A URL which can be used to access the authenticated subject user profile data. The URL MUST point to the same user profile as the one that was authenticated. The URL MUST be valid for the duration of the associated access token and refresh_tokens lifetimes.
lat	REQUIRED. The time at which the subject user was authenticated expressed in number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the date/time. See [RFC3339] for details regarding date/times in general and UTC in particular.
exp	OPTIONAL. The time at which the user authenticated session (login) expires expressed in number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the date/time. See [RFC3339] for details regarding date/times in general and UTC in particular. Note "expires_in" referes to the normal access token lifespan whereas "exp" refers to the lifespace of the user login session.
alv	OPTIONAL. The authentication assurance level as described by [NIST_SP-800-63-2].
iss	REQUIRED for session token. An identifier representing the issuer of the authentication. MAY be the authorization endpoint URL.
aud	REQUIRED for session_token. Contains the client_id of the client receiving the assertion.

Any claims, whether in the session parameter or the session token defined above MUST be understood before proceeding. Additional claims/parameters that are not understood MUST be ignored.

The client MUST confirm the "lat" is not future dated and "exp" is not a date currently in the past.

If an assurance level (alv) is to be returned higher than "2", then the information must be contained in a session token.

An example successful response using session (with carriage returns for readability):

```
     HTTP/1.1 200 OK
     Content-Type: application/json;charset=UTF-8
     Cache-Control: no-store
     Pragma: no-cache
     {
       "access_token":"2YotnFZFEjr1zCsicMWpAA",
       "token_type":"example",
       "expires_in":3600,
       "refresh_token":"tGzv3JOkF0XG5Qx2TlKWIA",
       "session":{
         "sub":"5dedcc8b-735c-405f-e029f",
         "sub_url":"https://example.com/Users/5dedcc8b-735c-405f-e029f",
         "lat":"1367956096",
         "exp":"1368042496",
         "alv":"2",
         "example_session_parameter":"example_value"
       }
       "example_parameter":"example_value"
     }
```

2.4.1. Session Token Processing

The "session_token" is a JSON Web Token [I-D.ietf-oauth-json-web-token] that contains the claims as described above. In addition to the attribute/claims validation rules above, If the assurance level (alv) is greater than "2", the token MUST be signed by the issuer. Clients MUST verify the validity of the signature and the values of "iss" and "aud" match the issuer and client_id.

As session tokens are bound to the client, clients SHOULD NOT share session tokens with other parties.

3. Privacy Considerations

Identifiers and URLs issued in [sub] and [sub_url] should be directed and valid only for the current OAuth client_id. This prevents multiple clients and non-OAuth clients from being able to gather and correlate information about individuals authenticated by the OAuth Authrization Server.

4. Acknowledgements

Thanks to members of the OAuth WG for their contributions and comments.

5. IANA Considerations

No IANA request registration is anticipated at this time.

6. Security Considerations

This draft carries the same risk profiles as those outlined in the Security Considerations for [RFC6749] and OAuth2 Threat Model [RFC6819].

7. References

7.1. Normative References

[I-D.ietf-oauth-json-web-token]
Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", draft-ietf-oauth-json-web-token-07 (work in progress), April 2013.

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

[RFC3339] Klyne, G., Ed. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, July 2002.

[RFC6749] Hardt, D., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012.

[RFC6755] Campbell, B. and H. Tschofenig, "An IETF URN Sub-Namespace for OAuth", RFC 6755, October 2012.

7.2. Informative References

[I-D.narten-iana-considerations-rfc2434bis]
Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", draft-narten-iana-considerations-rfc2434bis-09 (work in progress), March 2008.

[NIST_SP-800-63-2]
Burr, W., Dodson, D., Newton, E., Perlner, R., Polk, W., Newton, S., and E. Nabbus, "DRAFT NIST Special Publication 800-63-2: Electronic Authentication Guideline", February 2013.

[OIDC] Sakimura, N., Ping Identity, Jones, M., de Medeiros, B., and C. Mortimore, "OpenID Connect Basic Client Profile 1.0 - draft 28", OpenID Connect Specs http://openid.net/connect/, .

[RFC6819] Lodderstedt, T., McGloin, M., and P. Hunt, "OAuth 2.0 Threat Model and Security Considerations", RFC 6819, January 2013.

Author’s Address

Phil Hunt (editor)
Oracle