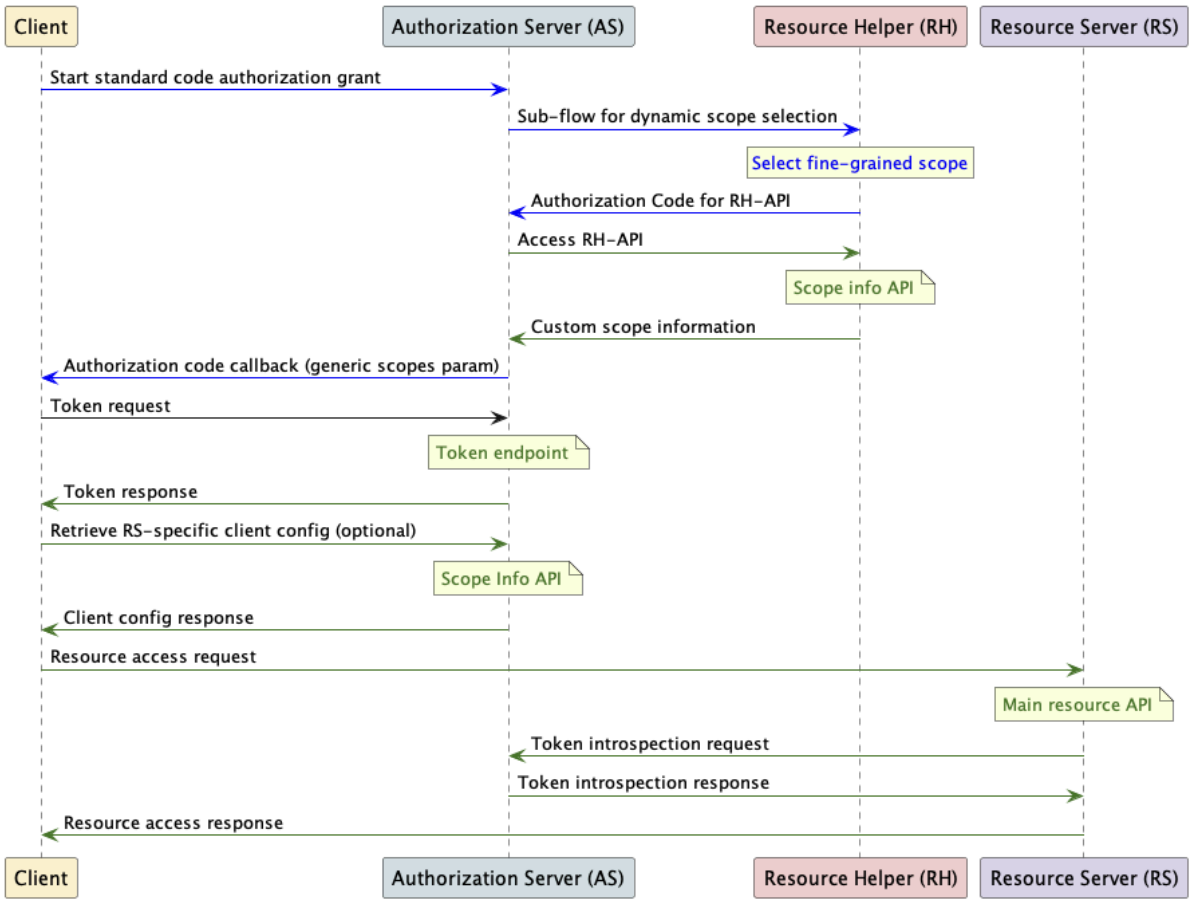
OAuth Scope Info

Abstract

We define the Resource Helper as a subsystem of an OAuth authorization server that is associated with a resource server. Implementations of OAuth authorization servers can delegate the scope selection to a Resource Helper. We further define the scope info data structure, which can convey more detailed information than the unstructured strings in an OAuth scopes parameter. The authorization server can retrieve this data structure from the Resource Helper, and use the information from it to display a scope label to the user, provide a Scope Info endpoint to clients, and include detailed scope information in access token payload and in introspection responses to the resource server. We also define an optional mechanism by which a client can request a token for a resource server that it does not yet know about by requesting a scope that refers to a protocol instead of to a specific resource server.

1. Introduction

In OAuth 2.0 [RFC6749], the part of an Authorization Server that deals with scope selection is necessarily tied to the specifics of the resource server. In some deployment scenarios, where one authorization server deals with many diverse resource servers, it is desirable to make this part of the authorization server pluggable.



2. The Resource Helper

The Resource Helper consists of two parts: a Scope Info API (RH-API), and a Scope Info Authorization Server (RH-AS).

[TODO: RH identifier and .well-known-endpoint]

2.1 Interaction between main AS and RH-AS

In the OAuth 2.0 authorization code and implicit flows, the resource owner is first redirected from the Client to the main Authorization Server.

[TODO: specify front channel redirection from AS to RH-AS, binding of AS session to RH-AS session]

In the OAuth Scope Info extension, before being redirected back to the client with a code or token, the resource owner is redirected to the authorization endpoint of the RH-AS, where they select or mint a specific scope description, and then grant the main AS access to it. The authorization code grant would typically be used here. Note that the main AS and the RH-AS may each require the RO to authenticate, even if the RO may already have authenticated to the client.

For the scopes parameter in this authorization request, the main AS MAY forward the exact value that it received from its client, OR apply a translation.

Afterwards, the resource owner is redirected back to the main AS, and the main authorization flow is completed as normal.

2.2 Interaction between main AS and RH-API

Once the main AS (acting as a Client) has obtained access to the RH-API (acting as a Resource Server), it can retrieve the Scope Info from the RH-API, contains four parts:

- A REQUIRED field “type” that must be set to “description”, to make clear that this JSON document only describes a resource, it does not represent access to this resource.
- A REQUIRED field “label”, to be used for display in the authorization dialog and in the revocation dialog of the main AS. This can either be a string or a locale-keyed object containing one string per locale. This label is analogous to the ones used for custom scopes in some existing AS configuration mechanisms.
- An OPTIONAL “protocols” field, containing an opaque object to be made available to clients of the main AS (see section 3). This is analogous to the existing “protocols” field in the Open Cloud Mesh protocol.
- A REQUIRED “payload” field, containing a string to be included into the access token.
- An OPTIONAL “introspect” field, containing an opaque object to be included into the introspection response, as defined in [RFC7662].

3. The client-facing Scope Info API

Optionally, the main AS can make a scope info API available to its own clients, where it makes available the “protocols” field from the RH-API response as client configuration. For

instance, when one main AS manages access to many Resource Servers, this could be a way for clients to discover the API location and capabilities of the Resource Server that was selected. The response from this API contains two parts:

- A REQUIRED field “type” that must be set to “grant”, to make clear that this JSON document doesn’t just describe a resource, but also confirms that this client has access to the resource that is described.
- A REQUIRED field “protocols” which must match the opaque object from the same key in the RH-API response.

4. The “scope_info” field in the introspection endpoint response
If the RH-API response contains an “introspect” field, and the main AS offers an introspection endpoint, then it SHOULD include the opaque object from the RH-API response into a “scope_info” field in the introspection response.

5. Authentication Considerations
The RH is responsible for authenticating the current user during the interaction through the authorization dialog of the RH-AS. The Client SHOULD NOT be granted access to which resources for which the authentication at the RH-AS is insufficient, regardless of authentication that may happen at the main AS or at the Client.

6. Security Considerations
The scope descriptions offered by the RH only attenuate the access that is granted by the main AS. The RH only helps the main AS to describe fine-grained scopes which the resource owner chooses to delegate in a language which the RS will understand.

The authentication of the user at the main AS and the authentication of the user at the RH-AS may be different, i.e. they might result in different identities. The RS, when interpreting the access token must be aware of this. The RS can learn about the identity of the user at the RH through the scope info that is opaquely passed on by the main AS in access token payload and, if applicable, in introspection information. The exact format and mechanism for this are out of the scope of this document.