OAuth
Internet-Draft
Updates: 6750 (if approved)
Intended status: Standards Track
Expires: August 22, 2013

H. Tschofenig, Ed.
Nokia Siemens Networks
February 18, 2013

OAuth 2.0: Audience Information
draft-tschofenig-oauth-audience-00.txt

Abstract

The OAuth 2.0 Bearer Token specification allows any party in possession of a bearer token to get access to the associated resources (without demonstrating possession of a cryptographic key). To prevent misuse, two important security assumptions must hold: bearer tokens must be protected from disclosure in storage and in transport and the access token must only be valid for use with a specific resource server (the audience) and with a specific scope.

This document defines a new header that is used by the client to indicate what resource server, as the intended recipient, it wants to access. This information is subsequently also communicated by the authorization server securely to the resource server, for example within the audience field of the access token.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction 3
2. Terminology 4
3. Audience Parameter 5
4. Processing Instructions 6
5. Security Considerations 8
6. IANA Considerations 9
7. Acknowledgments 10
8. References 11
8.1. Normative References 11
8.2. Informative References 11
Author's Address 12

1. Introduction

The OAuth 2.0 Bearer Token specification [1] allows any party in possession of a bearer token to get access to the associated resources (without demonstrating possession of a cryptographic key). To prevent misuse, two important security assumptions must hold: bearer tokens must be protected from disclosure in storage and in transport and the access token must only be valid for use with a specific resource server with a specific scope.

[1] describes this requirement in the following way:

> "To deal with token redirect, it is important for the authorization server to include the identity of the intended recipients (the audience), typically a single resource server (or a list of resource servers), in the token. Restricting the use of the token to a specific scope is also RECOMMENDED."

In general, if there is an authorization restriction then the respective parties must be aware of this restriction. In our case, the respective parties are authorization server (who has a trust relationship with the resource owner to accept for reject requests for data sharing and creates the access token), the client (who initiates the access to the protected resource), and the resource server (who protects the access to the resource and grants only access to those clients who have been approved by the authorization server).

Unfortunately, at the time of writing of [1] the access token format was still in early stages of the design and more details about how to communicate the audience information between the different parties was left unspecified. This document defines a new field for usage with OAuth 2.0. Note that it is not only useful for OAuth 2.0 bearer tokens but also for MAC tokens [5]: the authorization server needs to be told which resource server has to obtain the session key securely in order for the security properties to hold.

Restricting the usage of access tokens is important for several reasons: First, a stolen access token cannot be used with resource servers it has not been created for. Second, if the scope is included it cannot be used for requesting access to resources that exceed the indicated permissions. A resource server, who obtains an access token legitimately, cannot access resources on behalf of the resource owner at other resource servers.

2. Terminology

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 [2].

3. Audience Parameter

When the client interacts with the resource server it constructs the access token request to the token endpoint by adding the audience parameter using the "application/x-www-form-urlencoded" format with a character encoding of UTF-8 in the HTTP request entity-body.

The audience URI MUST be an absolute URI as defined by Section 4.3 of [3]. It MAY include an "application/x-www-form-urlencoded" formatted query component (Section 3.4 of [3]). The URI MUST NOT include a fragment component.

The ABNF syntax is defined as follows where by the "URI-reference" definition is taken from [3]:

```
audience = URI-reference
```

[QUESTION: Is it OK to just assume a URI here as the audience identifier?]

4. Processing Instructions

Step (0): As an initial step the client typically determines the resource server it wants to interact with, for example, as part of a discovery procedure.

[QUESTION: Should we talk about WebFinger or SWD to be more specific?]

Step (1): The client starts the OAuth 2.0 protocol interaction based on the selected grant type.

Step (2): When the client interacts with the token endpoint to obtain an access token it MUST populate the newly defined ’audience’ parameter with the information obtained in step (0).

Step (2): The resource server who obtains the request needs to parse it to determine whether the provided audience value matches any of the authorized resource servers it has a relationship with. If the authorization server fails to parse the provided value it MUST reject the request using an error response with the error code "invalid_request". If the authorization server does not consider the resource server acceptable then it MUST return an error response with the error code "access_denied". In both cases additional error information may be provided via the error_description, and the error_uri parameters. If the request has, however, been verified successfully then the authorization server MUST include the audience claim into the access token with the value copied from the audience field provided by the client. In case the access token is encoded using the JSON Web Token format [6] the "aud" claim MUST be used. The access token MUST be protected against modification by protecting it with either a digital signature or a keyed message digest. The authorization server returns the access token to the client, as specified in [4].

[QUESTION: Should we just focus on a JSON-based encoding of the access token since it is the only specified format?]

Step (3): The client follows the OAuth 2.0 specification [4] and the specification relevant for the selected token type (e.g., the bearer token specification) to interact with the resource server to make a request to the protected resource with the attached access token.

Step (4): When the resource server receives the access token it verifies it according to chosen access token encoding. For example, in case the JSON Web Token format is used then it must adhere to the guidance in [6]. In any case, the resource server MUST verify whether the URI contained in the "aud" claim matches it’s own. If the comparison fails the resource server MUST return an error to the client.

> [NOTE: More guidance is required in [6] regarding the matching procedure.]

5. Security Considerations

The sole purpose of this document is to extend the OAuth 2.0 protocol to improve security.

6. IANA Considerations

This document requires IANA to add a new value to the OAuth parameters registry:

- Parameter name: audience

- Parameter usage location: token request

- Change controller: IETF

- Specification document(s): [[This document.]

7. Acknowledgments

The author would like to thank Leif Johansson, and Eve Maler for their feedback.

8. References

8.1. Normative References

[1] Jones, M. and D. Hardt, "The OAuth 2.0 Authorization Framework: Bearer Token Usage", RFC 6750, October 2012.

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

[3] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

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

8.2. Informative References

[5] Richer, J., Mills, W., and H. Tschofenig, "OAuth 2.0 Message Authentication Code (MAC) Tokens", draft-ietf-oauth-v2-http-mac-02 (work in progress), November 2012.

[6] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", draft-ietf-oauth-json-web-token-06 (work in progress), December 2012.

Author's Address

Hannes Tschofenig (editor)
Nokia Siemens Networks
Linnoitustie 6
Espoo 02600
Finland

Phone: +358 (50) 4871445
Email: Hannes.Tschofenig@gmx.net
URI: http://www.tschofenig.priv.at