Workgroup: Web Authorization Protocol
Internet-Draft:
draft-ietf-oauth-iss-auth-resp-02
Published: 6 October 2021
Intended Status: Standards Track
Expires: 9 April 2022
Authors: K. Meyer zu Selhausen
Hackmanit

D. Fett
yes.com

OAuth 2.0 Authorization Server Issuer Identification

Abstract

This document specifies a new parameter iss that is used to explicitly include the issuer identifier of the authorization server in the authorization response of an OAuth authorization flow. The iss parameter serves as an effective countermeasure to "mix-up attacks".

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 https://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 9 April 2022.

Table of Contents

1. Introduction
   1.1. Conventions and Terminology
2. Response Parameter iss
   2.1. Example Authorization Response
   2.2. Example Error Response
   2.3. Providing the Issuer Identifier iss
   2.4. Validation of the Issuer Identifier iss
3. Authorization Server Metadata
4. Security Considerations
5. IANA Considerations
   5.1. OAuth Authorization Server Metadata
   5.2. OAuth Parameters Registration
6. Acknowledgements
7. Normative References
8. Informative References

Appendix A. Document History

Authors' Addresses

1. Introduction

The OAuth authorization framework [RFC6749] allows clients to interact with multiple independent authorization servers under the control of separate entities. Some OAuth grant types utilize the resource owner's user-agent to deliver the authorization server's response to the OAuth client. One example of this pattern is the authorization response of the authorization code grant.

The authorization response as specified in Section 4.1.2. of [RFC6749] does not contain any information about the identity of the authorization server which issued the response. Therefore, clients receiving a response from the resource owner's user-agent cannot be sure who initially issued the response and the secrets contained therein. The lack of certainty about the origin of the response enables a class of attacks called "mix-up attacks".

Mix-up attacks are a potential threat to all OAuth clients that interact with multiple authorization servers. When at least one of these authorization servers is under an attacker's control, the attacker can launch a mix-up attack to acquire authorization codes or access tokens issued by any one of the other authorization servers. There are multiple ways in which an attacker can gain control over an authorization server supported by the client: For instance, an authorization server could become compromised, or the attacker could register their own authorization server, for example, using dynamic client registration ([RFC7591]).

OAuth clients that interact with only one authorization server are not vulnerable to mix-up attacks. However, when such clients decide to add support for a second authorization server in the future they become vulnerable and need to apply countermeasures to mix-up attacks.

Mix-up attacks aim to steal an authorization code or access token by tricking the client into sending the authorization code or access token to the attacker instead of the honest authorization or resource server. This marks a severe threat to the confidentiality and integrity of resources whose access is managed with OAuth. A detailed description and different variants of the mix-up attack class can be found in Section 4.4. of the OAuth Security Best Current Practice [I-D.ietf-oauth-security-topics].

This document defines a new parameter in the authorization response called iss. The iss parameter allows the authorization server to include its identity in the authorization response explicitly. The client can compare the value of the iss parameter to the issuer identifier of the authorization server (e.g., retrieved from its metadata) it believes it is interacting with. The iss parameter gives the client certainty about the authorization server's identity and enables it to send credentials such as authorization codes and access tokens only to the intended recipients. Therefore, the implementation of the iss parameter serves as an effective countermeasure to mix-up attacks.

1.1. Conventions and Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

This specification uses the terms "access token", "authorization code", "authorization code grant", "authorization server", "resource server", "authorization response", "grant type", and "client" defined by the OAuth 2.0 Authorization Framework [RFC6749] and the term "issuer identifier" defined by OAuth 2.0 Authorization Server Metadata [RFC8414].

2. Response Parameter iss

In authorization responses to the client, including error responses, an authorization server supporting this specification MUST indicate its identity by including the iss parameter in the response.

The iss parameter value is the issuer identifier of the authorization server which created the authorization response, as

defined in [RFC8414]. Its value MUST be a URL that uses the "https" scheme without any query or fragment components. If the authorization server provides metadata as defined in [RFC8414], the value of the parameter iss MUST be identical to the authorization server metadata value issuer.

2.1. Example Authorization Response

The following example shows an authorization response from the authorization server whose issuer identifier is https://honest.as.example (extra line breaks and indentation are for display purposes only):

```
HTTP/1.1 302 Found
Location: https://client.example/cb?
  code=x1848ZT64p4IirMPT0R-X3141MFPTuBX-VFL_cvaplMH58
  &state=ZWVlNDBlYzA1NjdkMDNhYjg3ZjUxZjAyNGQzMTM2NzI
  &iss=https%3A%2F%2Fhonest.as.example
```

2.2. Example Error Response

The following example shows an error response from the same authorization server (extra line breaks and indentation are for display purposes only):

```
HTTP/1.1 302 Found
Location: https://client.example/cb?
  error=access_denied
  &state=ZWVlNDBlYzA1NjdkMDNhYjg3ZjUxZjAyNGQzMTM2NzI
  &iss=https%3A%2F%2Fhonest.as.example
```

2.3. Providing the Issuer Identifier iss

Authorization servers supporting this specification MUST provide their issuer identifier to enable clients to validate the iss parameter effectively.

For authorization servers publishing metadata according to [RFC8414], the following rules apply:

- The issuer identifier included in the server's metadata value issuer MUST be identical to the iss parameter's value.

- The server MUST indicate its support for the iss parameter by setting the metadata parameter

authorization_response_iss_parameter_supported, defined in Section 3, to true.

Authorization servers MAY additionally provide the issuer identifier to clients by any other mechanism which is outside of the scope of this specification.

2.4. Validation of the Issuer Identifier iss

Clients that support this specification MUST extract the value of the iss parameter from authorization responses they receive if the parameter is present. Clients MUST compare the extracted and URL-decoded value to the issuer identifier of the authorization server where the authorization request was sent to. This comparison MUST use simple string comparison as defined in Section 6.2.1. of [RFC3986]. If the value does not match the expected issuer identifier, clients MUST reject the authorization response and MUST NOT proceed with the authorization grant. For error responses, clients MUST NOT assume that the error originates from the intended authorization server.

More precisely, clients that interact with authorization servers supporting OAuth metadata [RFC8414] MUST compare the iss parameter value to the issuer value in the server's metadata document. If OAuth metadata is not used, clients MUST use deployment-specific ways, for example a static configuration, to decide if the returned iss value is the expected value in the current flow (see also Section 4).

If clients interact with both authorization servers supporting this specification and authorization servers not supporting this specification, clients MUST store the information which authorization server supports the iss parameter. Clients MUST reject authorization responses without the iss parameter from authorization servers which do support the parameter according to the client's configuration. Clients SHOULD discard authorization responses with the iss parameter from authorization servers which do not indicate their support for the parameter. However, there might be legitimate authorization servers that provide the iss parameter without indicating their support in their metadata. The decision of whether to accept such responses is individual for every scenario and it is not in the scope of this specification.

In general, clients that support this specification MAY accept authorization responses that do not contain the iss parameter or reject them and exclusively support authorization servers which provide the iss parameter in the authorization response. This decision is individual for every scenario and it is not in the scope of this specification.

In OpenID Connect [OIDC.Core] flows where an ID Token is returned from the authorization endpoint, the value in the iss parameter MUST always be identical to the iss claim in the ID Token.

Section 4.1.2. of [RFC6749] already mandates that clients that do not support this specification MUST ignore the unrecognized iss parameter.

Note: The "JWT Secured Authorization Response Mode for OAuth 2.0 (JARM)" [JARM] defines a mechanism that conveys all authorization response parameters in a JWT. This JWT contains an iss claim that provides the same protection if it is validated as described in Section 2.4. Therefore, an additional iss parameter outside the JWT is not necessary when JARM is used.

3. Authorization Server Metadata

The following parameter for the authorization server metadata [RFC8414] is introduced to signal the authorization server's support for this specification:

authorization_response_iss_parameter_supported Boolean parameter indicating whether the authorization server provides the iss parameter in the authorization response as defined in Section 2. If omitted, the default value is false.

4. Security Considerations

The authorization response parameter iss defined in this specification serves as a countermeasure to mix-up attacks described in Section 4.4. of the OAuth Security Best Current Practices [I-D.ietf-oauth-security-topics] and in detail in "On the security of modern Single Sign-On Protocols: Second-Order Vulnerabilities in OpenID Connect" [arXiv.1508.04324] and "A Comprehensive Formal Security Analysis of OAuth 2.0" [arXiv.1601.01229]. The latter provides a formal proof of the effectiveness of the countermeasure.

Clients MUST validate the iss parameter precisely as described in Section 2.4 and MUST NOT allow multiple authorization servers to use the same issuer identifier. In particular, when authorization server details can be manually configured in the client, the client MUST ensure that the accepted iss values are unique for each authorization server.

The iss parameter enables a client to decide if an authorization server "expects" to be used in an OAuth flow together with a certain token endpoint and potentially other endpoints, like the userinfo endpoint ([OIDC.Core]). When OAuth metadata is used, the iss parameter identifies the issuer and therefore the respective OAuth metadata document which points to the other endpoints. When OAuth

metadata is not used, the client can use, for example, a statically configured expected iss value for each configured authorization server.

The issuer identifier contained in the authorization response is not cryptographically protected against tampering. In general, mechanisms such as JWTs (as specified in JARM [JARM]) could be used to protect the integrity of the authorization response. However, in mix-up attacks, the client generally receives the authorization response from an uncompromised authorization server. If an attacker can tamper this authorization response before it is received by the client, the attacker would also have direct access to the authorization code. The attacker does not need to execute a mix-up attack to steal the authorization code. Therefore, integrity protection for the authorization response is not necessary to defend against mix-up attacks.

There are also alternative countermeasures to mix-up attacks. When an authorization response already includes an authorization server's issuer identifier by other means, and this identifier is checked as laid out in Section 2.4, the use and verification of the iss parameter is not necessary and MAY be omitted. This is the case when OpenID Connect response types that return an ID token from the authorization endpoint (e.g., response_type=code id_token) or JARM response mode are used, for example. However, if a client receives an authorization response that contains multiple issuer identifiers, the client MUST reject the response if these issuer identifiers do not match. The details of alternative countermeasures are outside of the scope of this specification.

Mix-up attacks are only relevant to clients that interact with multiple authorization servers. However, clients interacting with only one authorization server might add support for a second authorization server in the future. By supporting multiple authorization servers they become vulnerable to mix-up attacks and need to apply countermeasures.

5. IANA Considerations

5.1. OAuth Authorization Server Metadata

This specification requests registration of the following values in the IANA "OAuth Authorization Server Metadata" registry of [IANA.OAuth.Parameters] established by [RFC8414].

Metadata Name: authorization_response_iss_parameter_supported
Metadata Description: Boolean value indicating whether the authorization server provides the iss parameter in the authorization response.

Change Controller: IESG
Specification Document(s): Section 3 of [[this document]]

5.2. OAuth Parameters Registration

This specification requests registration of the following values in the IANA "OAuth Parameters" registry of [IANA.OAuth.Parameters] established by [RFC6749].

Parameter name: iss
Parameter usage location: authorization response
Change Controller: IESG
Specification Document(s): Section 2 of [[this document]]

6. Acknowledgements

TBD

7. Normative References

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, DOI 10.17487/RFC6749, October 2012, <https://www.rfc-editor.org/info/rfc6749>.

[I-D.ietf-oauth-security-topics] Lodderstedt, T., Bradley, J., Labunets, A., and D. Fett, "OAuth 2.0 Security Best Current Practice", Work in Progress, Internet-Draft, draft-ietf-oauth-security-topics-18, 13 April 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-oauth-security-topics-18>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[IANA.OAuth.Parameters] IANA, "OAuth Parameters", <http://www.iana.org/assignments/oauth-parameters>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8414] Jones, M., Sakimura, N., and J. Bradley, "OAuth 2.0 Authorization Server Metadata", RFC 8414, DOI 10.17487/RFC8414, June 2018, <https://www.rfc-editor.org/info/rfc8414>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC

3986, DOI 10.17487/RFC3986, January 2005, <https://www.rfc-editor.org/info/rfc3986>.

8. Informative References

[OIDC.Core] Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., and C. Mortimore, "OpenID Connect Core 1.0 incorporating errata set 1", 8 November 2014, <http://openid.net/specs/openid-connect-core-1_0.html>.

[arXiv.1508.04324] Mainka, C., Mladenov, V., and J. Schwenk, "On the security of modern Single Sign-On Protocols: Second-Order Vulnerabilities in OpenID Connect", 18 August 2015, <http://arxiv.org/abs/1508.04324>.

[RFC7591] Richer, J., Ed., Jones, M., Bradley, J., Machulak, M., and P. Hunt, "OAuth 2.0 Dynamic Client Registration Protocol", RFC 7591, DOI 10.17487/RFC7591, July 2015, <https://www.rfc-editor.org/info/rfc7591>.

[JARM] Lodderstedt, T. and B. Campbell, "Financial-grade API: JWT Secured Authorization Response Mode for OAuth 2.0 (JARM)", 17 October 2018, <https://openid.net/specs/openid-financial-api-jarm.html>.

[arXiv.1601.01229] Fett, D., Kuesters, R., and G. Schmitz, "A Comprehensive Formal Security Analysis of OAuth 2.0", 6 January 2016, <https://arxiv.org/abs/1601.01229>.

Appendix A. Document History

[[To be removed from the final specification]]

-02 [[Working Group Draft]]
* Incorporated feedback from shepherd review
* Changed SHOULD to MUST (clients MUST store which AS support `iss` par
* Added note for clients receiving unexpected `iss` parameter
* Editorial changes

-01 [[Working Group Draft]]
* Incorporated WG feedback
* Changed title of the draft to make it shorter
* Clarified mix-up attacks in introduction
* Improved note on JARM in validation section

-00 [[Working Group Draft]]
* Working group draft

-02
* Incorporated WG feedback
* Clarifications for unique issuer identifier
* Clarifications when multiple issuer identifier could be present
* Added note that iss parameter MUST NOT be used with JARM
* Added note on error responses and example for error response
* Editorial changes

-01

* Incorporated first WG feedback
* Clarifications for use with OIDC
* Added note that clients supporting just one AS are not vulnerable
* Renamed metadata parameter
* Various editorial changes

-00

* initial draft

Authors' Addresses

Karsten Meyer zu Selhausen
Hackmanit

Email: karsten.meyerzuselhausen@hackmanit.de

Daniel Fett
yes.com

Email: mail@danielfett.de