OAuth Working Group	B. Campbell
Internet-Draft	J. Bradley
Intended status: Standards Track	Ping Identity
Expires: January 9, 2017	M. Jones
	Microsoft
	July 8, 2016

A Token Binding method for OAuth 2.0 Proof Key for Code Exchange

draft-campbell-oauth-tbpkce-00

Abstract

This specification describes a Proof Key for Code Exchange (PKCE) [RFC7636] method utilizing Token Binding over HTTP [I-D.ietf-tokbind-https] to cryptographically bind the OAuth 2.0 [RFC6749] authorization code to a key pair on the client, which it proves possession of during the access token request with the authorization code.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This specification minimally describes an OAuth 2.0 [RFC6749] PKCE [RFC7636] method based on the Token Binding Protocol [I-D.ietf-tokbind-protocol] and Token Binding over HTTP [I-D.ietf-tokbind-https]. The general details and motivations of PKCE are discussed in that document and this specification defines only the additional pieces needed for a Token Binding PKCE method.

1.1. Requirements Notation and Conventions

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 RFC 2119 [RFC2119].

1.2. Terminology

This specification uses the terms "authorization code", "authorization endpoint", "authorization server", "authorization request", "access token request", "client", and "token endpoint" defined by OAuth 2.0 [RFC6749], and the terms "Provided", "Token Binding" and "Token Binding ID" defined by Token Binding over HTTP [I-D.ietf-tokbind-https].

2. Code Challenge

As defined in Proof Key for Code Exchange [RFC7636], the client sends the code challenge as part of the OAuth 2.0 Authorization Request with the two additional parameters: code_challenge and code_challenge_method.

For the Token Binding method of PKCE defined herein, tb2 is used for the value of the code_challenge_method parameter.

The value of the code_challenge parameter is the base64url encoding (per Section 5 of [RFC4648] with all trailing pad ('=') characters omitted and without the inclusion of any line breaks or whitespace) of the SHA-256 hash [RFC6234] of the Provided Token Binding ID that the client will use when calling the authorization server's token endpoint. Note that, prior to making the authorization request, the client may need to establish a TLS connection between itself and the authorization server's token endpoint in order to obtain the appropriate Token Binding ID.

When the authorization server issues the authorization code in the authorization response, it associates the code challenge and method values with the authorization code so it can be verified later when the code is presented in the access token request.

3. Code Verifier

Upon receipt of the authorization code, the client sends the access token request to the token endpoint. The Token Binding Protocol [I-D.ietf-tokbind-protocol] is negotiated on the TLS connection between the client and the authorization server and the Sec-Token-Binding header, as defined in Token Binding over HTTP [I-D.ietf-tokbind-https], is included in the access token request. The authorization server extracts the Provided Token Binding ID from the header value, hashes it with SHA-256, and compares it to the code_challenge value previously associated with the authorization code. If the values match, the token endpoint MUST continue processing as normal (as defined by OAuth 2.0 [RFC6749]). If the values do not match, an error response indicating "invalid_grant" MUST be returned.

The Sec-Token-Binding header contains sufficient information for verification of the authorization code and its association to the original authorization request. However, PKCE [RFC7636] requires that a code_verifier parameter be sent with the access token request, so the static value provided is used to meet that requirement and indicate that the Provided Token Binding ID is used for the verification.

4. Security Considerations

TBD

5. IANA Considerations

5.1. PKCE Code Challenge Method Registration

This specification requests registration of the following Code Challenge Method Parameter Name in the IANA "PKCE Code Challenge Methods" registry [IANA.OAuth.Parameters] established by [RFC7636].

5.1.1. Registry Contents

- Code Challenge Method Parameter Name: tb2
- Change controller: IESG
- Specification document(s): Section 2 of [[this specification]]

6. Normative References

[I-D.ietf-tokbind-https]	Popov, A., Nystrom, M., Balfanz, D., Langley, A. and J. Hodges, "Token Binding over HTTP", Internet-Draft draft-ietf-tokbind-https-03, March 2016.
[I-D.ietf-tokbind-protocol]	Popov, A., Nystrom, M., Balfanz, D., Langley, A. and J. Hodges, "The Token Binding Protocol Version 1.0", Internet-Draft draft-ietf-tokbind-protocol-06, May 2016.
[IANA.OAuth.Parameters]	IANA, "OAuth Parameters"
[RFC2119]	Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997.
[RFC4648]	Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006.
[RFC6234]	Eastlake 3rd, D. and T. Hansen, "US Secure Hash Algorithms (SHA and SHA-based HMAC and HKDF)", RFC 6234, DOI 10.17487/RFC6234, May 2011.
[RFC6749]	Hardt, D., "The OAuth 2.0 Authorization Framework", RFC 6749, DOI 10.17487/RFC6749, October 2012.
[RFC7636]	Sakimura, N., Bradley, J. and N. Agarwal, "Proof Key for Code Exchange by OAuth Public Clients", RFC 7636, DOI 10.17487/RFC7636, September 2015.

Appendix A. Acknowledgements

Dirk Balfanz, William Dennis (and others?) also provided input to this specification.

Appendix B. Document History

[[to be removed by the RFC Editor before publication as an RFC]]

draft-campbell-oauth-tbpkce-00

- Initial version.

Authors' Addresses

Brian Campbell
Ping Identity
EMail: brian.d.campbell@gmail.com
URI: https://twitter.com/__b_c

John Bradley
Ping Identity
EMail: ve7jtb@ve7jtb.com
URI: http://www.thread-safe.com/

Michael B. Jones
Microsoft
EMail: mbj@microsoft.com
URI: http://self-issued.info/