Workgroup: OAuth Working Group
Internet-Draft: draft-sakimura-oauth-wmrm-01
Published: 8 November 2023
Intended Status: Standards Track
Expires: 11 May 2024
Authors: T. Yamaguchi N. Sakimura, Ed. N. Matake
Timee, Inc. NAT Consulting LLC YAuth.JP LLC

OAuth 2.0 Web Message Response Mode

Abstract

This specification defines a new response mode for RFC6749 that uses HTML5 Web Messaging (a.k.a window.postMessage()) instead of the redirect for the Authorization Response from the Authorization Endpoint. It defines two modes: simple mode and relay mode. Relay mode can be used to protect the access token in the implicit grant case by confining it within the origins of authorization server or resource server and preventing it from being read by the client.

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 11 May 2024.

Copyright Notice

Copyright (c) 2023 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 (https://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 Revised BSD License text as described in Section 4.e of the

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

Table of Contents

1. Introduction

This specification defines a new response mode for RFC6749 that uses HTML5 Web Messaging (a.k.a window.postMessage()) instead of the redirect for the Authorization Response from the Authorization Endpoint.

This specification provides two modes

1. Simple mode returns the Authorization Response directly to the client web page.

2. Relay mode does not return the Authorization Response directly to the client web page but returns it to a child frame created by the client web page via the client web page.

1.1. Requirements Language

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].

2. Terms and definitions

For the purposes of this specification, the following terms and definitions apply.

2.1. Main Window

window object that the client created

2.2. Authenticated Window

A window object in the iframe created by the Main Window(Section 2.1) or its child iframe.

It is used to relay the Authorization Request from the client to the Authorization Server. The request is expected to have prompt=none defined in OpenID Connect Core.

It is expected to be used when the End User is in the "authenticated" state at the Authorization Server so that prompt=none succeeds. Authorization Server SHOULD NOT send Set-Cookie header in the response.

2.3. Unauthenticated Window

A popup window object created by the Main Window(Section 2.1), via window.open(), to send the Authorization Request to the Authorization Endpoint.

It is used when the user is not in the Authenticated state on the Authorization Server or the client has not received the authorization yet.

2.4. Message Targeted Window

A window object in the iframe created by the Main Window(Section 2.1) that receives Authorization Response in Relay Mode (Section 4.2).

3. Symbols and abbreviated terms

Authz Authorization

4. Protocol Flows

As stated above, this response mode provides two modes.

4.1. Simple mode

In the Simple mode, the protocol flows as follows.

4.1.1. Simple mode:no prompt

The authorization request and response sequence when using Authenticated Window will be like this.

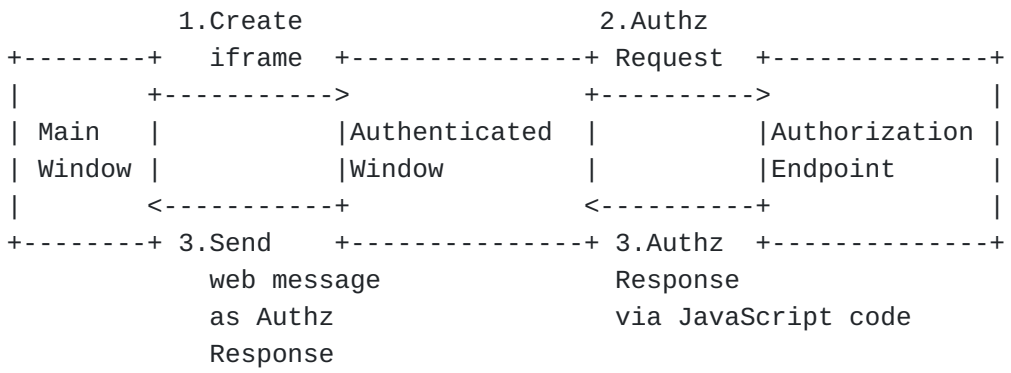

Figure 1: Simple mode (Authenticated Window)

Below is the description of the each steps in the above sequence diagram.

1. Main Window creates the Unauthenticated Windows as an iframe to send Authorization Request and sets the URI of the Authorization Request as the src value of the iframe.

2. The Authorization Request that is specified by the src value of the iframe of the Unauthenticated Window, which was set by the Main Window, is sent to Authorization Endpoint.

3. Authorization Server determines if the End User is logged in state and checks if it can return the Authorization Response without interacting with the User. If it determines that it can, it will render the HTML5 that includes the JavaScript code that sends the Authorization Response.

4. The Authorization Response is passed to the Main Window from the Unauthenticated Window using Web Message through the JavaScript.

4.1.2. Simple mode:with prompt

When using Unauthenticated Window, the Authorization sequence would be as follows:

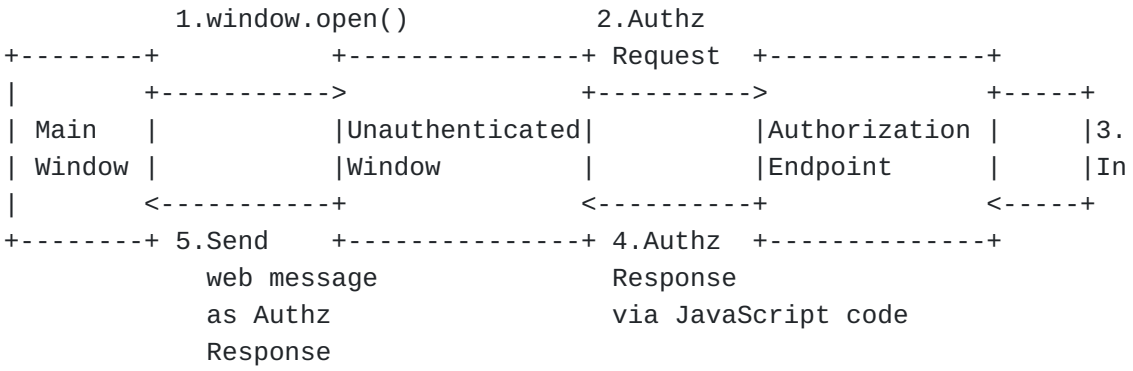

Figure 2: Simple mode (Unauthenticated Window)

It is almost identical to the sequence that used Authenticated Window except:

1. The way the window object that is used to send the Authorization Request is different. i.e., iframe v.s. window.open().

2. End User may interact with the Authorization Server before the Authorization Response is being sent.

3. The relationship of the window that accesses the Authorization Endpoint and the Main Window is different. i.e., window.parent v.s. window.opener.

4.2. Relay Mode

The protocol flow of the Relay mode will be as follows.

4.2.1. Relay mode:no prompt

The authorization sequence that uses Unauthenticated Window will be:

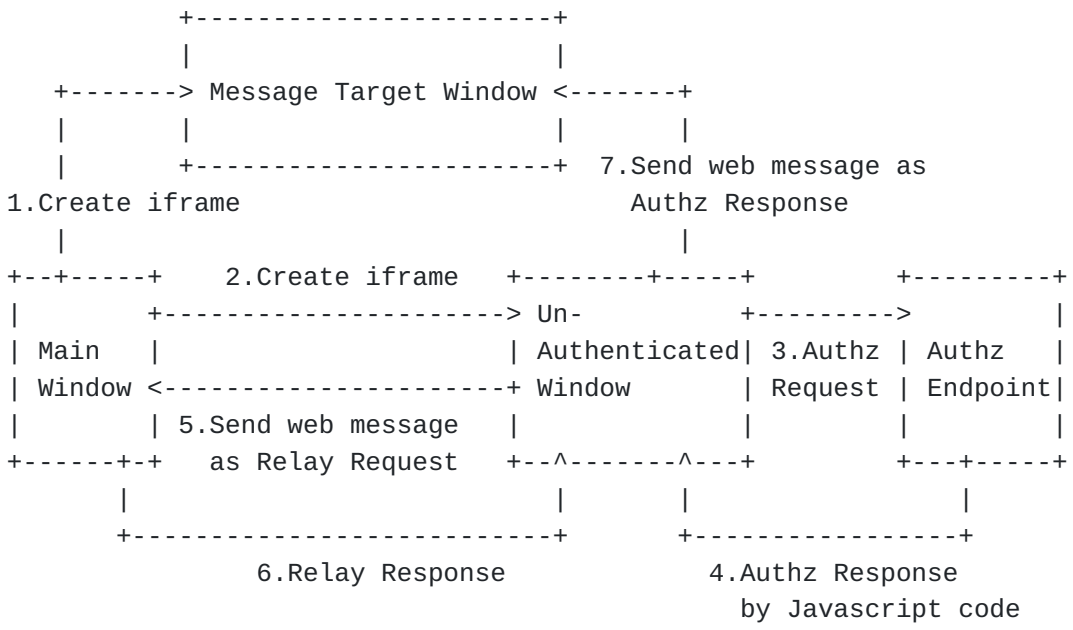

Figure 3: Relay Mode (Authenticated Window)

1. Create iframe (Message Target Window) through Main Window. It will receive the Authorization Response.

2. Create iframe (Unautenticated Window) through Main Window. The Authorization Request URI is set as the src value of the iframe. This iframe is used to send the Authorization Request.

3. Authorization Request specified in the src value by the Main Window is sent to the Authorization Endpoint.

4. Authorization Server checks the login status of the End User and whether it can return the Authorization Response without the End User interaction. If it is, then it will render the HTML5 that includes JavaScript code that sends Relay Request and Authorization Response.

5. Unauthenticated Window sends Relay Request as a Web Message to the Main Window throught the JavaScript code.

6. Main Window returns the Relay Response.

7. Unauthenticated window obtains the window object of the Message Target Window via the MessageEvent object in the Relay Response and send Authorization Response as a Web Message.

4.2.2. Simple mode:with prompt

The authorization sequence that uses the Authenticated Window follows almost the same sequence.

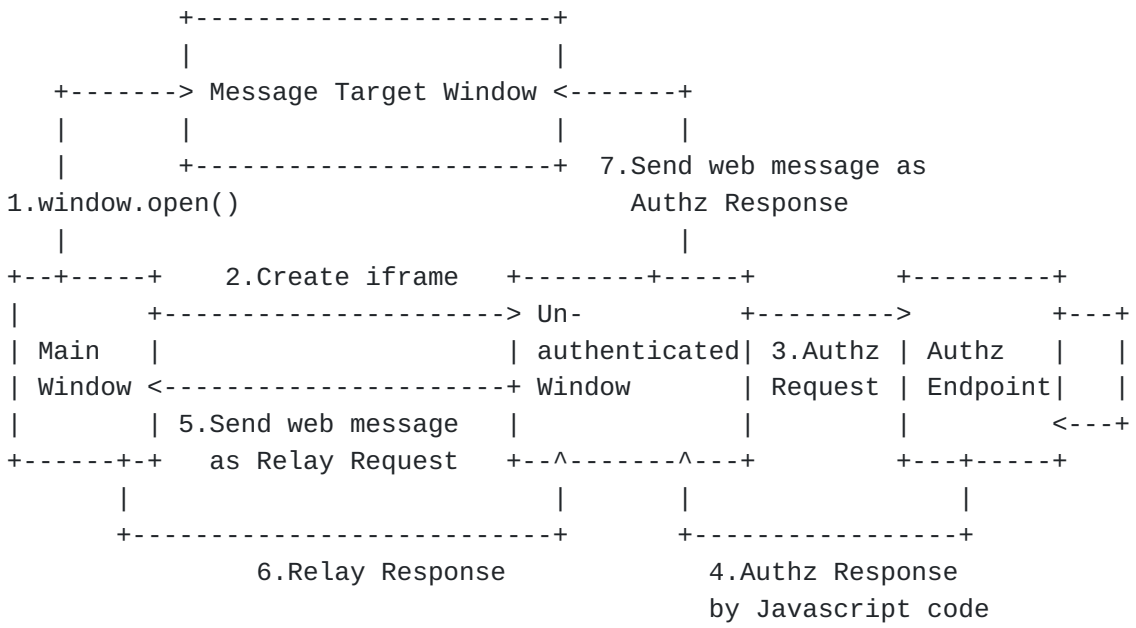

Figure 4: Relay Mode (Authenticated Window)

The differences are, just like in the Simple Mode:

1. The way the window object that is used to send the Authorization Request is different. i.e., iframe v.s. window.open().

2. End User may interact with the Authorization Server before the Authorization Response is being sent.

3. The relationship of the window that accesses the Authorization Endpoint and the Main Window is different. i.e., window.parent v.s. window.opener.

5. Authorization

5.1. Authorization Request

Web Messaging Response Mode defines the following Authorization Request parameters.

response_mode REQUIRED. ASCII string "web_message".

redirect_uri REQUIRED. The origin URI of the URI of the Main Window. If web_message_uri is not specified, Authorization Responsewill be sent to the origin specifed by the redirect_uri.

web_message_uri OPTIONAL. The origin URI that Message Target Window references. When it is specified, Authorization Response will not

be returned to the redirect_uri but Relay Request/Responses are used.

web_message_target OPTIONAL or REQUIRED. The DOM id value that points to the Message Target Window. REQUIRED if web_message_uri is used. Authorization Response obtains the window object of Message Target Window via Relay Request/Relay Response with the Main Window. If it is not specified, Authorization Responsewill be sent to the calling window.

Main WIndow creates an event listener before sending the Authorization Request, and sends Authorization Request that uses these parameters to either Authenticated Window or Unauthenticated Window.

The following example depicts the Authorization Request to the Unauthenticated Window in the Simple Mode.

```
function connect(request, callback) {
    var authorizationEndpoint = (function(url) {
        var a = document.createElement("a");
        a.setAttribute("href", url);
        return a;
    })("https://as.example.com/authorize");
    authorizationEndpoint.search = buildQueryString(request, {
        "redirect_uri": location.origin,
        "response_mode": "web_message"
    });
    window.addEventListener("message", function(evt) {
        if (evt.origin != "https://as.example.com")
            return;
        if (!evt.data.type)
            return;
        switch (evt.data.type) {
            case "authorization_response":
                evt.source.close();
                (evt.data.error) ? callback(null, evt.data): callback(ev
                window.removeEventListener("message", arguments.callee,
                break;
            default:
        }
    }, false);
    var unauthentecatedWindow = window.open(authorizationEndpoint.getAtt
    return unauthenticatedWindow;
}
```

Figure 5: Registration of the event listener (Unauthenticated Window in the Simple mode)

Actual authorization request will look like:

```
GET /authorize?response_type=token&client_id=s6BhdRkqt3&state=xyz
 &redirect_uri=https%3A%2F%2Fclient%2Eexample%2Ecom
 &response_mode=web_message HTTP/1.1
Host: as.example.com:443
```

Figure 6: Authorization Request

Following depicts the Authorization Request to Authenticated Window in Relay Mode.

```
function getConnectedStatus(request, callback) {
    var authorizationEndpoint = (function(url) {
        var a = document.createElement("a");
        a.setAttribute("href", url);
        return a;
    })("https://as.example.com/authorize");
    authorizationEndpoint.search = buildQueryString(request, {
        "redirect_uri": location.origin,
        "response_mode": "web_message",
        "web_message_uri": "https://api.example.com",
        "web_message_target": "apiFrame"
    });
    window.addEventListener("message", function(evt) {
        if (evt.origin != "https://as.example.com")
            return;
        if (!evt.data.type)
            return;
        switch (evt.data.type) {
            case "relay_request":
                evt.source.postMessage("message", {
                    type: "relay_response"
                }, false);
                (evt.data.error) ? callback(null, evt.data): callback(ev
                window.removeEventListener("message", arguments.callee,
                break;
            default:
        }
    }, false);
    var authenticatedWindow = (function(url) {
        var iframe = document.getElementById("apiFrame");
        if (!iframe) {
            iframe = document.createElement("iframe");
            iframe.setAttribute("width", 0);
            iframe.setAttribute("height", 0);
        }
        iframe.setAttribute("href", url);
        return iframe.contentWindow;
    })(authorizationEndpoint.getAttribute("href"));
    return authenticatedWindow;
}
```

Figure 7: Registration of the event listener that receives Authorization Response (Authenticated Window in Relay Mode)

Actual authorization request will look like:

```
GET /authorize?response_type=token&client_id=s6BhdRkqt3&state=xyz
&redirect_uri=https%3A%2F%2Fclient%2Eexample%2Ecom
&response_mode=web_message
&web_message_uri=https%3A%2F%2Fapi%2Eexample%2Ecom
&web_message_target=apiFrame HTTP/1.1
Host: as.example.com:443
```

Figure 8: Authorization Request (Authenticated Window)

Message Target Window in Relay mode creates an event listener to receive Authorization Response.

```
(function(window, document, undefined) {
    window.addEventListener("message", function(evt) {
        if (evt.origin != "https://as.example.com")
            return;
        if (!evt.data.type)
            return;
        switch (evt.data.type) {
            case "authorization_response":
                if (evt.source.parent == evt.source) {
                    evt.source.close();
                }
                processAuthorizationResponse(evt.data);
                break;
            default:
        }
    }, false);
})(this, this.document);
```

Figure 9: Receiving Authorization Response in Message Target Window

Web Messages between Authenticated Window or Unauthenticated Window and Main Window or Message Target Window takes the following fields.

field	type	required	description
type	string	true	prepare_authorization_response OR authorization_response
response	object	false	used when type=authorization_response

Table 1: Web Messages Structure

Type attribute values are described in the following table.

mode	sender	receiver	type	description
Simple Mode	Authenticated Window or	Main Window	authorization_response	response including

mode	sender	receiver	type	description
	Unauthenticated Window			Authorization Response
Relay Mode	Authenticated Window or Unauthenticated Window	Main Window	relay_request	Request to get the reference to the window object of the Main Window
Relay Mode	Main Window	Authenticated Window or Unauthenticated Window	relay_response	The response to the relay_request
Relay Mode	Authenticated Window or Unauthenticated Window	Message Target Window	authorization_response	Response that includes Authorization Response

Table 2: Type attributes

5.2. Authorization Response

Authorization Server needs to render the JavaScript code to return the Authorization Response when response_mode was web_message at the time of Authorization Request at Authorization Endpoint.

Authorization Server MUST verify the following before returning Authorization Response.

(1) The origin specified by redirect_uri is white-listed.

(2) The origin specified by web_message_uri is white-listed.

If verified, it MUST return the response including the JavaScript code such as:

```
<!DOCTYPE html>
<html>

<head>
    <title>Authorization Response</title>
</head>

<body>
    <script type="text/javascript">
        (function(window, document, undefined) {
                // Begin : these values rendered by server
                var redirectURI = "https://client.example.com";
                var webMessageRequest = {};
                var authorizationResponse = {
                    type: "authorization_response",
                    response: {
                        code: "SplxlOBeZQQYbYS6WxSbIA",
                        state: "xyz"
                    }
                };
                // End
                var mainWin = (window.opener != window) ? window.opener
                // For relay mode
                if (webMessageRequest["web_message_uri"] && webMessageRe
                    window.addEventListener("message", function(evt) {
                            if (evt.origin != redirectURI)
                                return; // replay mode
                            switch (evt.data.type) {
                                case "relay_response":
                                    messageTargetWindow =
                                        evt.source.document.getElementBy
                                    if (messageTargetWindow) {
                                        messageTargetWindow.postMessage(
                                            type: "authorization_respons
                                            response: authorizationRespo
                                        }, webMessageRequest["web_messag
                                    }
                                default:
                            }
                        }
                        mainWin.postMessage({
                            type: "relay_request"
                        }, redirectURI);
                    } else {
                        mainWin.postMessage({
                            type: "authorization_response",
                            response: authorizationResponse
                        }, redirectURI);
```

```
                    }
                })(this, this.document);
    </script>
</body>

</html>
```

Figure 10: Authorization Response with web messaging response mode

If web_message_uri and web_message_target request parameters are specified in Authorization Request, window object sent by postMessage() is not to be set to window.opener or window.parent but to a specific frame, responses such as follows should be returned.

```
<!DOCTYPE html>
<html>

<head>
    <title>Authorization Response</title>
</head>

<body>
    <script type="text/javascript">
        (function(window, document, undefined) {
                // Begin : these values rendered by server
                var redirectURI = "https://client.example.com";
                var webMessageRequest = {
                    web_message_uri: "https://api.example.com",
                    web_message_target: "apiFrame"
                };
                var authorizationResponse = {
                    type: "authorization_response",
                    response: {
                        code: "SplxlOBeZQQYbYS6WxSbIA",
                        state: "xyz"
                    }
                };
                // End
                var mainWin = (window.opener != window) ? window.opener
                // For relay mode
                if (webMessageRequest["web_message_uri"] && webMessageRe
                    window.addEventListener("message", function(evt) {
                            if (evt.origin != redirectURI)
                                return; // replay mode
                            switch (evt.data.type) {
                                case "relay_response":
                                    messageTargetWindow =
                                        evt.source.document.getElementBy
                                    if (messageTargetWindow) {
                                        messageTargetWindow.postMessage(
                                            type: "authorization_respons
                                            response: authorizationRespo
                                        }, webMessageRequest["web_messag
                                    }
                                default:
                            }
                        }
                        mainWin.postMessage({
                            type: "relay_request"
                        }, redirectURI);
                    } else {
                        mainWin.postMessage({
```

```
                        type: "authorization_response",
                        response: authorizationResponse
                    }, redirectURI);
                }
            })(this, this.document);
    </script>
</body>

</html>
```

Figure 11: Authorization Response w/ web messaging response mode and web_message_target

6. Client Metadata

The following field is added to RFC7519.

field	type	description
web_message_uris	array	List of origins that are allowed as web_message_uri in the Authorization Request.

Table 3: Client Metadata Addition

7. IANA Considerations

Followings are added to OAuth Dynamic Client Registration Metadata Registry.

*Client Metadata Name: "web_message_uris"

*Client Metadata Description: List of origins that are allowed as web_message_uri in the Authorization Request.

*Change Controller: IESG

*Specification Document(s): This document

8. Security Considerations

In addition to the all the security considerations discussed in OAuth 2.0 [RFC6819], the following security considerations SHOULD be taken into account.

9. Acknowledgements

Follwoing people contributed to the creation of this document .

10. Revision History

-01

*Updated authors contacts

-00

*Initial Draft.

11. Normative References

[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>.

[RFC6819] Lodderstedt, T., Ed., McGloin, M., and P. Hunt, "OAuth 2.0 Threat Model and Security Considerations", RFC 6819, DOI 10.17487/RFC6819, January 2013, <https://www.rfc-editor.org/info/rfc6819>.

[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>.

[Web.Messaging] Hickson, I., "HTML5 Web Messaging", 19 May 2015, <http://www.w3.org/TR/webmessaging/>.

Authors' Addresses

Toru Yamaguchi
Timee, Inc.

Email: toru.yamaguchi@timee.co.jp
URI: https://corp.timee.co.jp/

Nat Sakimura (editor)
NAT Consulting LLC

Email: nat@nat.consulting
URI: http://nat.sakimura.org/

Nov Matake
YAuth.JP LLC

Email: nov@matake.jp
URI: http://matake.jp