RFC6749 - The OAuth 2.0 Authorization Framework

return to the IETF Standards

Table 1: Data sheet for The OAuth 2.0 Authorization Framework

Title	Transmission Control Protocol
Acronym	OAuth
Version	2.0
Document Number	RFC6749
Release Date	October 2012
Reference	https://tools.ietf.org/html/rfc6749

Note: The following is an excerpt from the official IETF RFC. It is provided here as a convenience and is not authoritative. Refer to the original document as the authoritative reference.

Introduction

In the traditional client-server authentication model, the client requests an access-restricted resource (protected resource) on the server by authenticating with the server using the resource owner's credentials. In order to provide third-party applications access to restricted resources, the resource owner shares its credentials with the third party. This creates several problems and limitations:

- *Third-party applications are required to store the resource owner's credentials for future use, typically a password in clear-text.*
- *Servers are required to support password authentication, despite the security weaknesses inherent in passwords.*
- *Third-party applications gain overly broad access to the resource owner's protected resources, leaving resource owners without any ability to restrict duration or access to a limited subset of resources.*
- *Resource owners cannot revoke access to an individual third party without revoking access to all third parties, and must do so by changing the third party's password.*
- *Compromise of any third-party application results in compromise of the end-user's password and all of the data protected by that password.*

OAuth addresses these issues by introducing an authorization layer and separating the role of the client from that of the resource owner. In OAuth, the client requests access to resources controlled by the resource owner and hosted by the resource server, and is issued a different set of credentials than those of the resource owner.

Instead of using the resource owner's credentials to access protected resources, the client obtains an access token – a string denoting a specific scope, lifetime, and other access attributes. Access

tokens are issued to third-party clients by an authorization server with the approval of the resource owner. The client uses the access token to access the protected resources hosted by the resource server.

For example, an end-user (resource owner) can grant a printing service (client) access to her protected photos stored at a photo- sharing service (resource server), without sharing her username and password with the printing service. Instead, she authenticates directly with a server trusted by the photo-sharing service (authorization server), which issues the printing service delegation- specific credentials (access token).

This specification is designed for use with HTTP (RFC2616). The use of OAuth over any protocol other than HTTP is out of scope.

The OAuth 1.0 protocol (RFC5849), published as an informational document, was the result of a small ad hoc community effort. This Standards Track specification builds on the OAuth 1.0 deployment experience, as well as additional use cases and extensibility requirements gathered from the wider IETF community. The OAuth 2.0 protocol is not backward compatible with OAuth 1.0. The two versions may co-exist on the network, and implementations may choose to support both. However, it is the intention of this specification that new implementations support OAuth 2.0 as specified in this document and that OAuth 1.0 is used only to support existing deployments. The OAuth 2.0 protocol shares very few implementation details with the OAuth 1.0 protocol. Implementers familiar with OAuth 1.0 should approach this document without any assumptions as to its structure and details.

From:
https://omgwiki.org/dido/ - **DIDO Wiki**

Permanent link:
https://omgwiki.org/dido/doku.php?id=dido:public:ra:xapend:xapend.b_stds:tech:ietf:oauth&rev=1588892182

Last update: **2020/05/07 18:56**