Matthias Biehl

Getting Started in Web-API Security

OAuth 2.0

OAuth 2.0

Getting Started in Web-API Security

by Matthias Biehl

API University Series
www.api-university.com

ISBN-13: 978-1507800911
ISBN-10: 1507800916

Table of Contents

Summary

This book offers an introduction to API Security with OAuth 2.0. In less than 80 pages you will gain an overview of the capabilities of OAuth. You will learn the core concepts of OAuth. You will get to know all 4 OAuth Flows that are used in cloud solutions and mobile apps.

If you have tried to read the official OAuth specification, you may get the impression that OAuth is complex. This book explains OAuth in simple terms. The different OAuth Flows are visualized graphically using sequence diagrams. The diagrams allow you to see the big picture of the various OAuth interactions. This high-level overview is complemented with rich set of example requests and responses and an explanation of the technical details.

In the book the challenges and benefits of OAuth are presented, followed by an explanation of the technical concepts of OAuth. The technical concepts include the actors, endpoints, tokens and the four OAuth flows. Each flow is described in detail, including the use cases for each flow. Extensions of OAuth are presented, such as OpenID Connect and the SAML2 Bearer Profile.

Who should read this book?

You do not have the time to read long books? This book provides an overview, the core concepts, without getting lost in the small-small details. This book provides all the necessary information to get started with OAuth in less than 80 pages.

You believe OAuth is complicated? OAuth may seem complex with flows and redirects going back and forth. This book will give you clarity by introducing the seemingly complicated material by many illustrations. These illustrations clearly show all the involved interaction parties and the messages they exchange.

You want to learn the OAuth concepts efficiently? This book uses many illustrations and sequence diagrams. A good diagram says more than 1000 words.

You want to learn the difference between OAuth and OpenID Connect? You wonder when the two concepts are used, what they have in common and what is different between them. This book will help you answer this question.

You want to use OAuth in your mobile app? If you want to access resources that are protected by OAuth, you need to get a token first, before you can access the resource. For this, you need to understand the OAuth flows and the dependencies between the steps of the flows.

You want to use OAuth to protect your APIs? OAuth is perfectly suited to protect your APIs. You can learn which OAuth endpoints need to be provided and which checks need to be made within the protected APIs.

1 Introduction

People have gotten a bit sensitive about internet security and privacy. "Mobile apps, web-APIs and Cloud Services - yes, I like and use them, but ... is my data really secure there? Can I control what happens to my data and who can access is?" These and many related questions are top-of-mind for many cloud and mobile users. And, who can blame them? With the recent incidents of compromised accounts and stolen passwords, these types of question are more than justified. Organizations that offer mobile apps and cloud services have to address these questions of their users. These organizations are not any longer only web-startups, Google and Facebook. Today, the business of almost every industry is transforming into a digital business. Businesses across the different industries thus need to think about information security. To differentiate, more and more traditional businesses increasingly create digital services for their customers.

That is why all types of businesses need to face the security questions of their users. Users demand the responsible processing, storing and transmission of their data – and companies have to react now. To win the trust of their customers and users, organizations need to take the concerns of their users seriously. They can do this by building on established standards instead of building proprietary solutions.

In the context of web-APIs, mobile apps and cloud services, there are two established standards for authentication and authorization: OAuth 2 and OpenID Connect. But which standard should be used in a given scenario? How does the technology work? Which experiences have been gathered from practical use of these technologies?

1.1 The Password Anti-Pattern

Let us start with an example: Sarah uses the mobile app of her car insurance to register a claim for a minor accident. On the mobile app of her insurance company, she first has to authenticate by entering her username and password. Because entering passwords is cumbersome, the mobile app saves the credentials on the mobile.

A second example: Tim wants his tweets from Twitter to appear on LinkedIn automatically to stay in touch with his business contacts. To realize this functionality, LinkedIn would need to have access to Tim's Twitter account. The simplistic solution would be to provide LinkedIn with the credentials of Twitter, so LinkedIn can directly access Tim's tweets.

However, both "solutions" would be quite a security risk, since Sarah's password is saved unprotected on the mobile and Tim's password is provided to another cloud service. Both instances are examples of the "Password Anti-Pattern". In practice, this solution cannot be used.

1.2 What is OAuth 2?

OAuth 2 is a standard for delegating authorization for accessing resources via HTTP.

OAuth 2 offers a solution for the scenarios of the examples above without the risks of the password anti-pattern. With OAuth 2 we can give access rights to the mobile app, without providing the password. Instead, a token is handed to the app. The token represents the access rights for a subset of the data, for a short time frame. To obtain the token, the user first logs in on the website of the OAuth server. The generated token can be an authorization code, an access token or a refresh token. An access token allows access to a resource during a limited time period. In case the token gets compromised, the access rights associated with the token can be revoked.

Sarah and Tim from the previous example will not notice any difference, whether OAuth is used or not. They can use their mobile apps and cloud apps in a secure manner, if OAuth is used under the hood.

In fact, most of OAuth is happening under the hood of modern cloud, mobile and web applications. The end user can directly notice a few advantages. The advantages are that they have a fine-granular control over the access to their data, do not need to give their password to third parties, and if they should lose their mobile, they can remotely revoke all OAuth tokens which are stored on the lost device.

OAuth is used for mobile, cloud services, and web APIs. OAuth 2 is a standard that is used in mobile integration use cases, when mobile apps need to communicate securely with server-side backend systems. OAuth 2 is a standard for securing many well-known web APIs, such as the APIs offered by Google, Twitter, LinkedIn, Amazon and Ebay. Most cloud-based Software-as-a-Service offerings use OAuth for protecting their services and the data of their users.

OAuth 2 is specified and standardized by the IETF in RFC6749 (http://tools.ietf.org/html/rfc6749). OAuth 1 has been replaced by OAuth 2, is outdated and not presented here. This is why we use the short form OAuth to refer to OAuth 2.

1.3 Terms

Two similar terms -- authentication and authorization -- are used in the context of OAuth and API Security. To understand the details of OAuth, it is essential to know the distinction between the two:

Authentication is a concept for answering the question: Who are you? Authentication provides a method for providing proof for the claimed identity.

Authorization is a concept that answers the question: What are you allowed to do? Authorization provides the rights assigned to the confirmed identity, for example access rights. For OAuth, authentication is a precondition for proper authorization.

OAuth relies on authentication and authorization but does neither. This can be confusing, since the name "OAuth" suggests that it might be related to one of them.

OAuth 2 is a framework for delegation of HTTP-based access. Authentication is performed by another component, for example by the mechanisms of a login page. Authorization needs to be performed by the API, which uses the token and information related to the token for authorizing access to the protected resource.