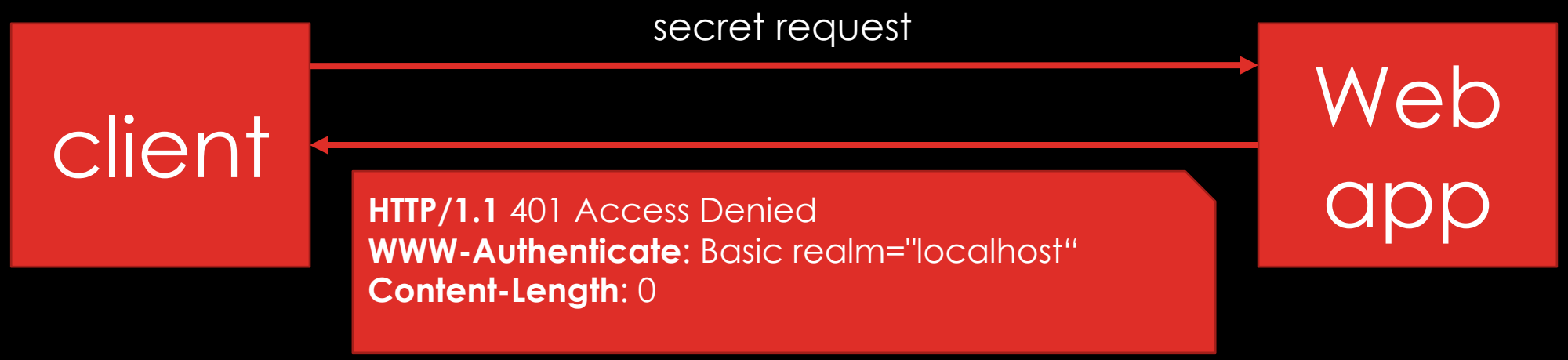
SLIDES AND CODE

- https://gitlab.com/JHelm/oauth2-demo
- Tutorial: https://gitlab.com/JHelm/oauth2-demo/tree/master/tutorial

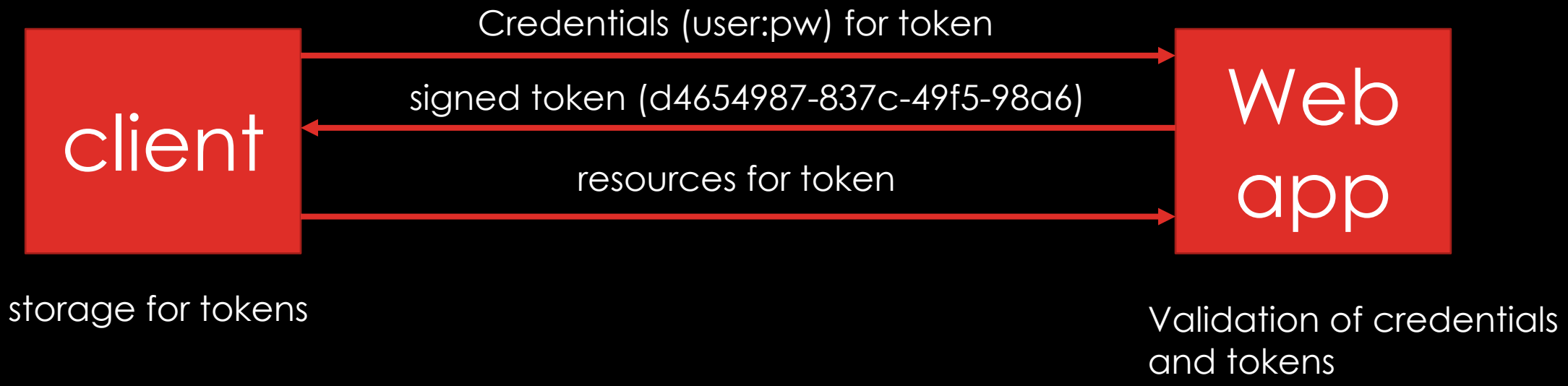
BASIC AUTHENTICATION

- HTTP is stateless: for **each** request you have to send your credentials (username and password)



BASIC AUTHENTICATION

TOKEN BASED AUTHENTICATION



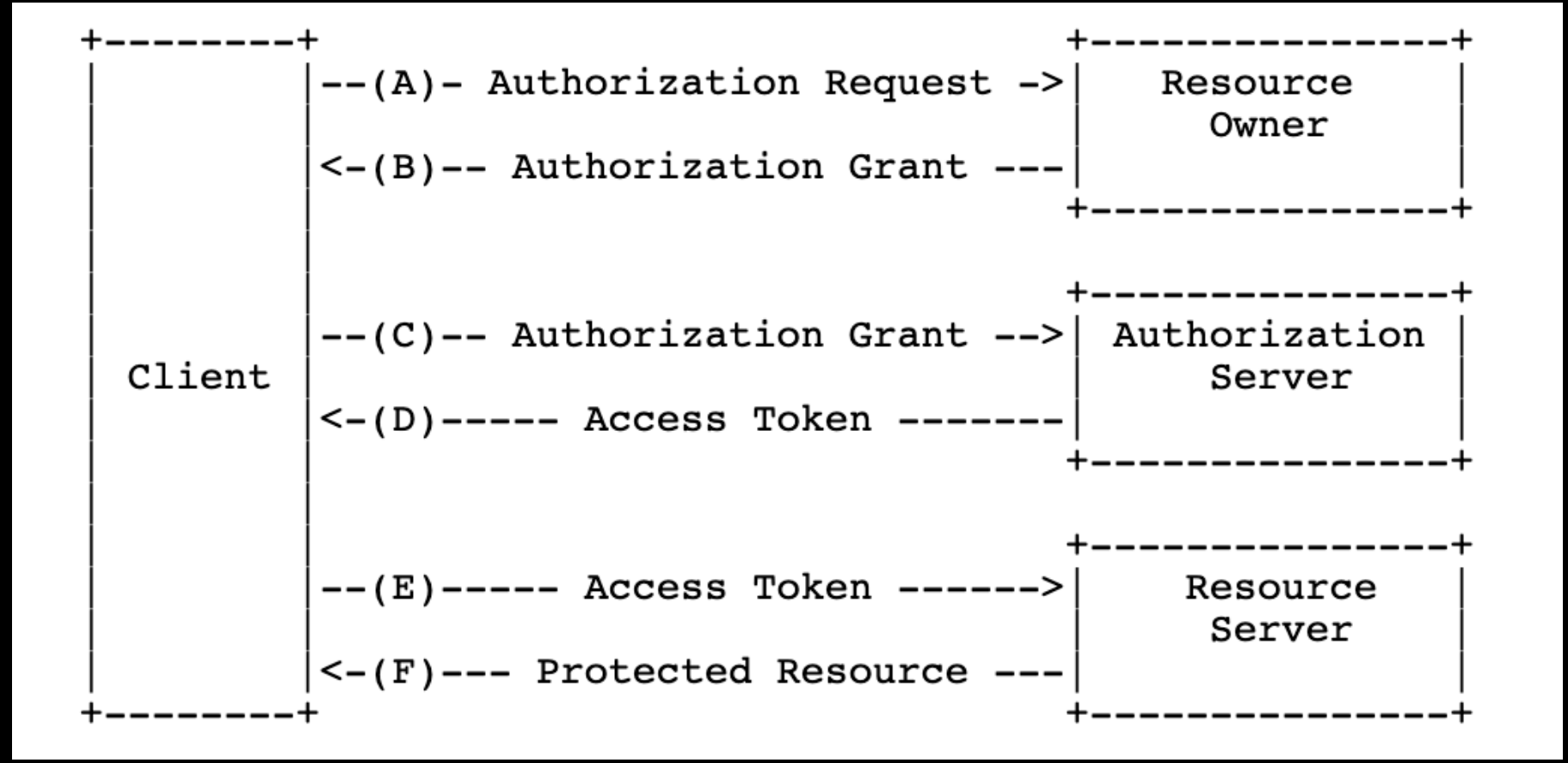
THE OAUTH 2.0 AUTHORIZATION FRAMEWORK

https://tools.ietf.org/html/rfc6749#section-1.3.3

TERMINOLOGY

resource owner	• user • restricts the scope of the data
client	• application that wants to access protected resources on behalf of a user • can only access data that is absolutely necessary (restricted by user)
resource server	• management of protected resources • only the user is allowed to access his data
authorization server	• authorizes the access of an user after its valid authentication
access token	• temporary valid • used by resource-server to access the user data and check its rights
refresh token	• used to get a new access-token

PROTOCOL FLOW



https://tools.ietf.org/html/rfc6749, page 6, access: 14.10.2019 12:09

AUTHORIZATION GRANT

- credential, that represents the clients/ressource owners authorization
  - **client credentials**
  - **authorization code**
  - **resource owner password credentials**
  - implicit

CLIENT CREDENTIALS

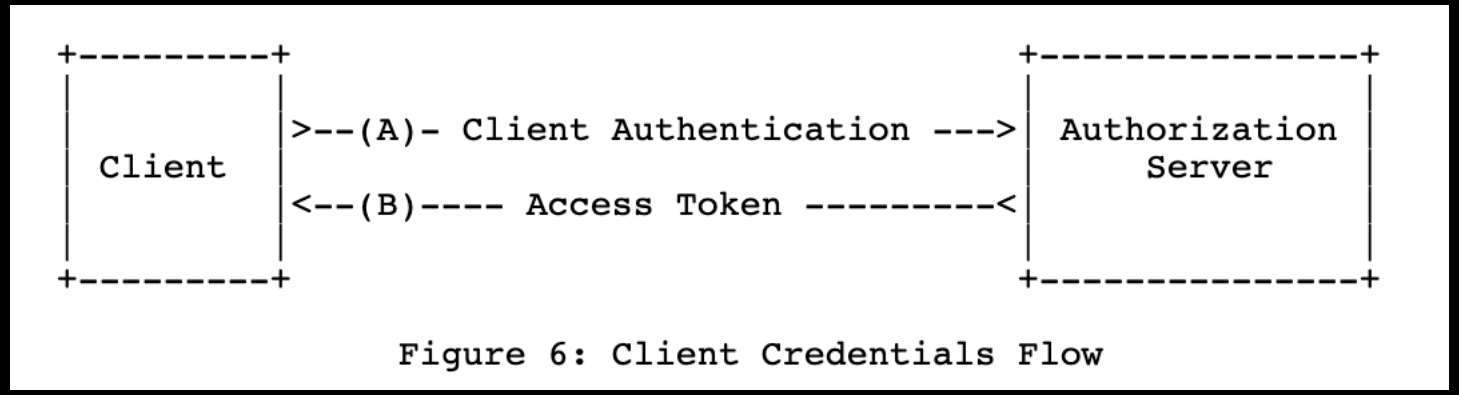


Figure 6: Client Credentials Flow

(a) authentication with the authorization server and request access token from token endpoint
(b) authorization server authenticates client and issues an access token

CLIENT CREDENTIALS

curl *--request* POST
--url http://localhost:8080/oauth/token
--header 'authorization: Basic d2ViOnNlY3JldA==‘
--header 'content-type: application/x-www-form-urlencoded‘
--data grant_type=client_credentials

RESPONSE:

{
"**access_token**": "16cac3a3-1e0e-452a-bbd3-36478eb7d96a",
"**token_type**": "bearer",
"**expires_in**": 59,
"**scope**": "read write“
}

DEMO

Client Credentials

RESOURCE OWNER PASSWORD CREDENTIALS GRANT

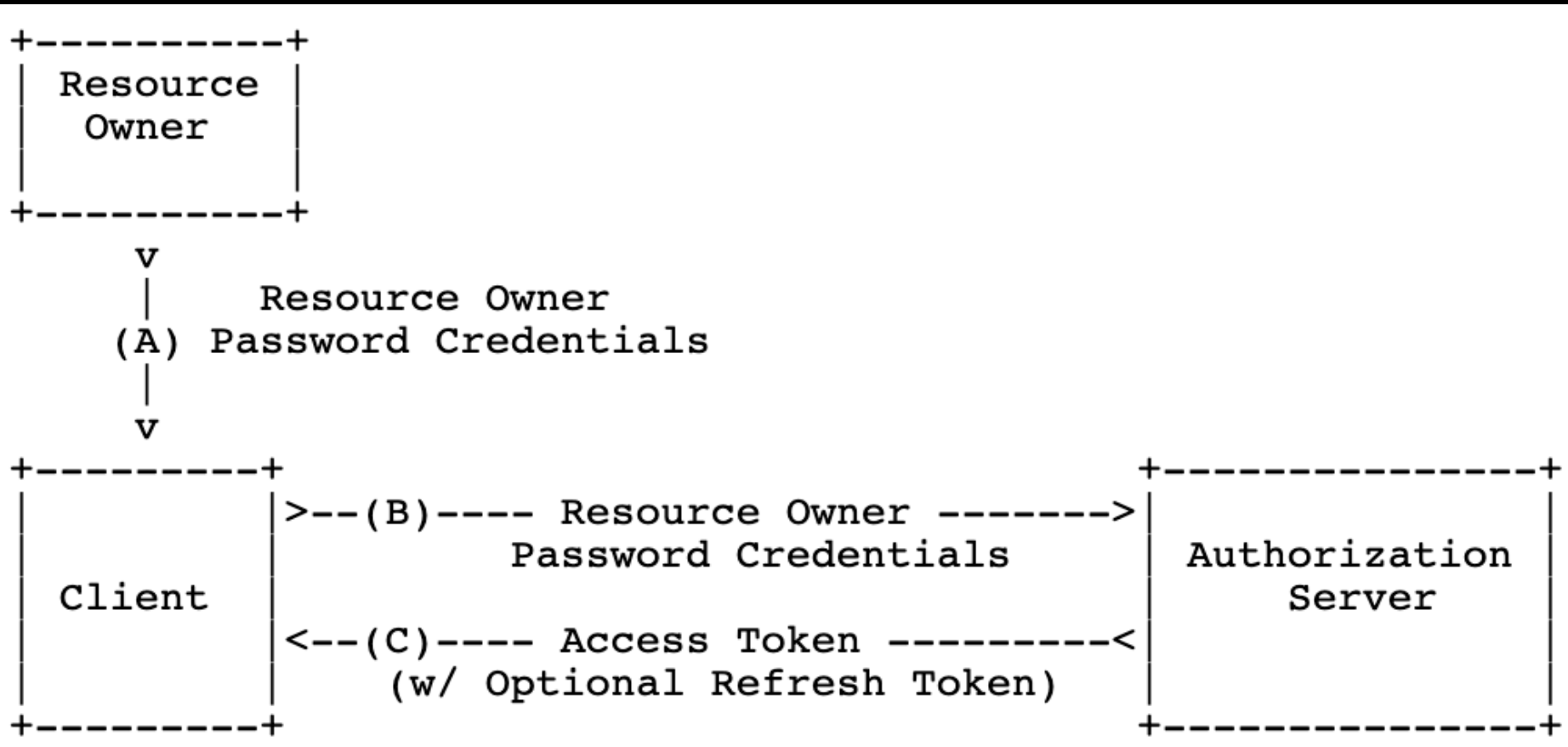


Figure 5: Resource Owner Password Credentials Flow

RESOURCE OWNER PASSWORD CREDENTIALS

curl *--request* POST
--url http://localhost:8080/oauth/token
--header 'authorization: Basic d2ViOnNlY3JldA==‘‘ <- client credentials!
--header 'content-type: multipart/form-data; boundary=---011000010111000001101001‘
--form username=maxmuster
--form grant_type=password
--form password=secret

RESPONSE:

{

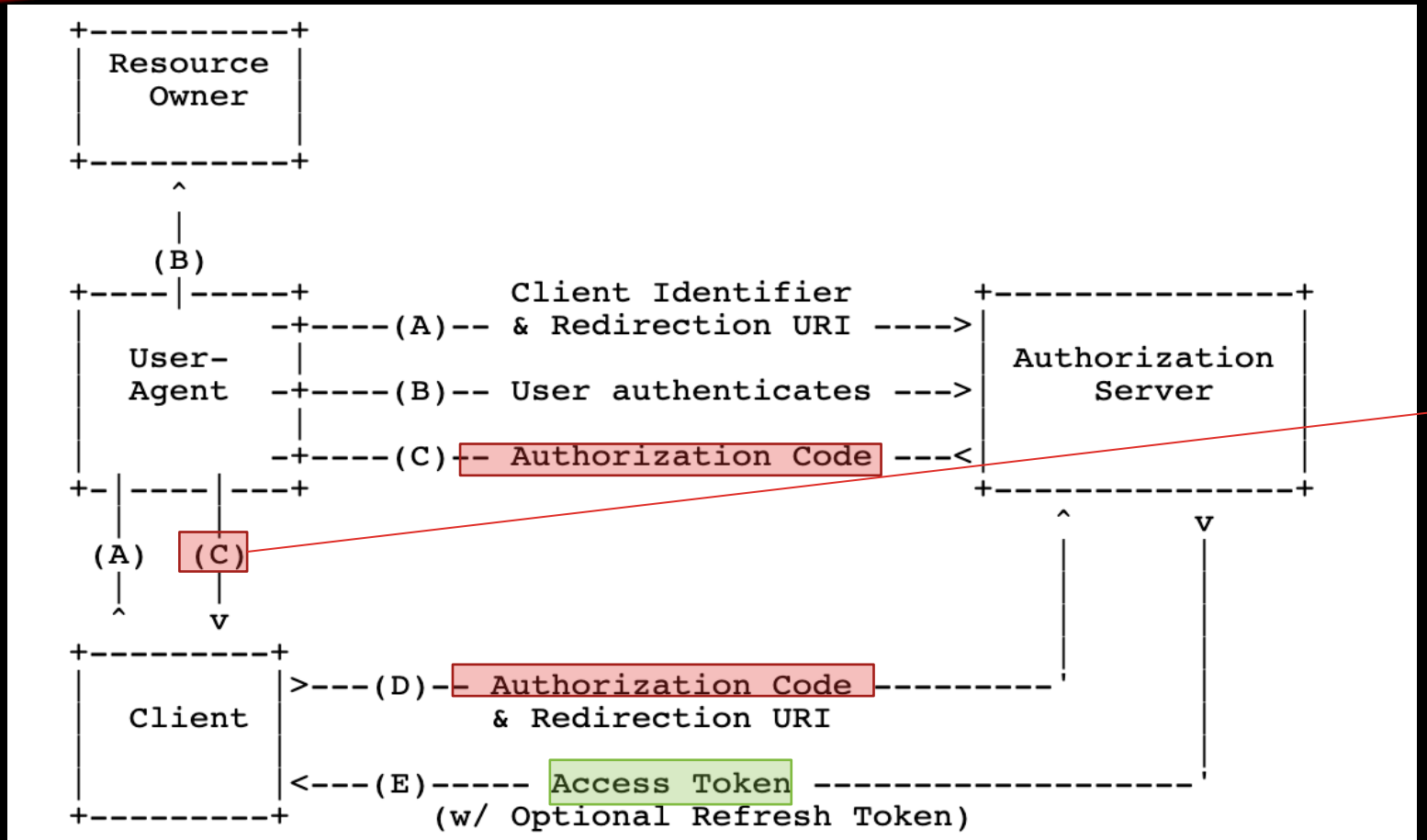
"**access_token**": "08d5565f-f7db-4b26-9e76-6d59b3c7d6a6",
"**token_type**": "bearer",
"**refresh_token**": "8638e3c3-93e2-4e88-a1fe-3c880857f918",
"**expires_in**": 59,
"**scope**": "read write”

}

DEMO

Resource Owner Password Credentials

AUTHORIZATION CODE GRANT

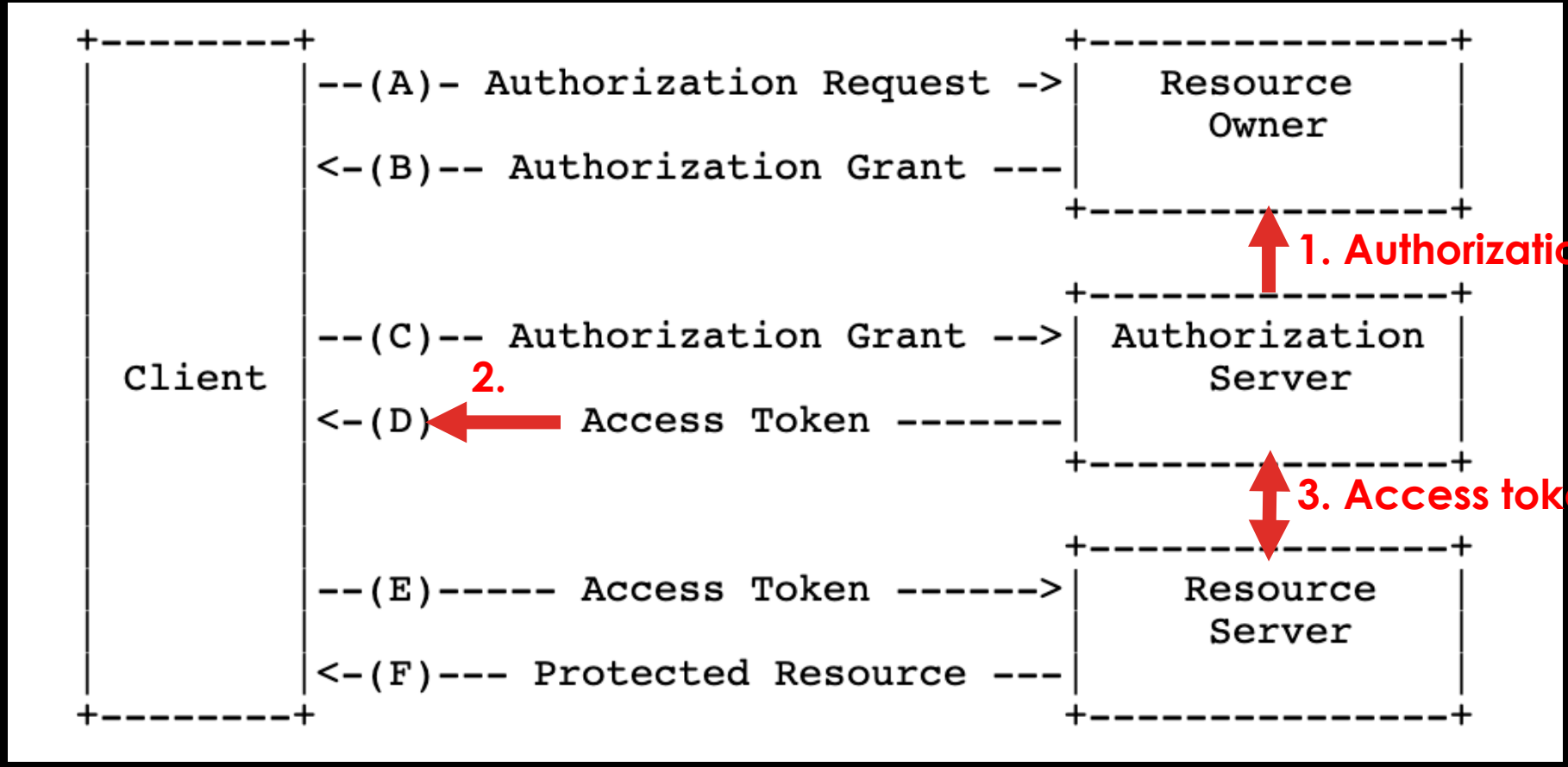


Note: The lines illustrating steps (A), (B), and (C) are broken into two parts as they pass through the user-agent.

Figure 3: Authorization Code Flow

http://localhost:8080/login?**code=b087c5d6a9993ddebd64**

AUTHORIZATION CODE



https://tools.ietf.org/html/rfc6749, page 6, access: 14.10.2019 12:09

DEMO

Authorization Code

TUTORIAL

https://gitlab.com/JHelm/oauth2-demo/tree/master/tutorial

SOURCES

- https://tools.ietf.org/html/rfc6749
- Entwickler Magazin Spezial Volume 16 “Security – Sichere IT-Systeme bauen“, Software & Support Media GmbH, 2018
- https://spring.io/guides/tutorials/spring-boot-oauth2/