

**Міністерство освіти і науки України**  
**Національний технічний університет України «Київський політехнічний**  
**інститут імені Ігоря Сікорського»**  
**Факультет інформатики та обчислювальної техніки**  
  
**Кафедра інформатики та програмної інженерії**

**Звіт**

З лабораторної роботи № 2 з дисципліни  
«Безпека програмного забезпечення»

**“Авторизаційний протокол OAuth2”**

**Виконав(ла)**

*ІІ-13 Бабіч Денис*

\_\_\_\_\_  
(шифр, прізвище, ім'я, по батькові)

**Перевірів(ла)**

*Соколовський В. В.*

\_\_\_\_\_  
(шифр, прізвище, ім'я, по батькові)

Київ 2024

## ЛАБОРАТОРНА РОБОТА № 2

**Тема роботи:** Авторизаційний протокол OAuth2.

**Мета роботи:** Засвоїти базові навички OAuth2 авторизаційного протокола.

**Основне завдання:**

1. Використовуючи наведені налаштування, створити запит на отримання токenu через client\_credential grant.

```
import requests

URL = "https://kpi.eu.auth0.com/oauth/token"

HEADERS = { "content-type": "application/x-www-form-urlencoded" }

PAYLOAD = {
    "audience": "https://kpi.eu.auth0.com/api/v2/",
    "grant_type": "client_credentials",
    "client_id": "JIvC05c2IBH1Ae2patn6l6q5H35qxti0",
    "client_secret": "ZRF80p0tWM36p1_hxXTU-B0K_Gq_-eAVtLrQpY24CasYiDmcXBhNS6IJMNcz1EgB"
}

response = requests.post(URL, headers = HEADERS, data = PAYLOAD)

print(response.json())
```

Рисунок 1.1 – Код для звернення до ендпоінту з метою отримання токenu

У цьому коді реалізується запит на отримання токenu доступу за допомогою протоколу OAuth 2.0. Далі задається змінна URL, що містить адрес веб-сервісу, до якого надсилається запит, у даному випадку це URL для отримання токenu в Auth0. Наступна змінна HEADERS визначає заголовки запиту, де вказується, що тип контенту - "application/x-www-form-urlencoded", що є стандартним форматом для передачі даних у формі ключ-значення. Потім створюється словник PAYLOAD, який містить дані, що будуть надіслані в тілі запиту і цей словник включає тип запиту, ідентифікатор клієнта та секрет клієнта. Далі виконується POST-запит за допомогою методу requests.post, який приймає URL, заголовки та дані, результат запиту зберігається в змінній response.

У результаті виконання коду на рисунку 1.1 було отримано наступну відповідь, яка містить доступний токен авторизації типу Bearer, що дозволяє

доступ до API, такий токен складається з трьох частин, шифрується та підписується, що забезпечує безпеку під час передачі даних. Домен дії токена включає права на читання та створення користувачів, що вказує на можливість взаємодії з користувацькими даними

```
{
  'access_token':
    'eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsImtpZCI6IjVCZTlBZFhrMERaUjhmR1dZYjdkViJ9.eyJpc3MiOiJodHRwczovL2twaS5ldS5hdXRoMC5jb20vIiwic3ViIjoiaSk12Q081YzJJQkhsQWUycGF0bjZsNnE1SDM1cXh0aTBAY2xpZW50cyIsImF1ZCI6Imh0dHBzOi8va3BpLmV1LmF1dGgwLmNvbS9hcGkvZjIvIiwiaWF0IjoxNzI3NjUwNzA3LCJleHAiOiJlE3Mjc3MzcwMDcsInNjb3BlIjoicmVhZDp1c2VycyBjcmVhdGU6dXNlcnMiLCJndHkiOiJjbGllbnQtY3JlZGVudGllbmhmMiLCJhenAiOiJKSXZDTzVjMklCSGxBZTJwYXRuNmww2cTVIMzVxeHRpMCJ9.fXnVM6clYkUp3HsYBPnvzsOgBWxP6rejao7depzsPu5wlQApwtnxuTsptevvY-YFfJjS9imIS9JQpV_T317nKS09HGrpeZQy2E_2xiMl4ht1SLvjXQD9z-nDMUvaw0uz93uYgwCGCr5U1reBkaxhMmNLCjeOHXueyCMZUnsaVuuZg32S_apaRAOPgmX5ump4-4aBtu4Axk-o2WBxFEs6aPzMtOwu7zF3wCTZ5juHzH-rukKBrKXC17CJmOX-q5JQjOe0DC5ZUwu9PfThsYju6Fer4vymJqdL7Swyz6chaC-83DdR90pecSh4lY2RLB5z2B12JOr0P0U9U_-KU11QQg',
  'scope': 'read:users create:users',
  'expires_in': 86400,
  'token_type': 'Bearer'
}
```

Тип токена "Bearer" вказує, що для доступу до ресурсу необхідно надіслати токен доступу в заголовок Authorization у форматі "Bearer {token}". Це означає, що будь-хто, хто має цей токен, може отримати доступ до захищених ресурсів, що робить важливим правильне управління та захист токенів від несанкціонованого використання.



У результаті виконання коду на рисунку 1.2 було повернуто статус-код 201, що свідчить про те, що користувача було успішно створено.

```
{"blocked":false,"created_at":"2024-09-29T23:06:03.895Z","email":"denis.babich@mail.com","email_verified":false,"family_name":"Babich","given_name":"Denys","identities":[{"connection":"Username-Password-Authentication","user_id":"66f9dd5b99d33f07238e0a45","provider":"auth0","isSocial":false}],"name":"Denys","nickname":"Ngenx","picture":"https://i.etsystatic.com/26254942/r/il/f54dcc/4754579609/il_300x300.4754579609_aps7.jpg","updated_at":"2024-09-29T23:06:03.896Z","user_id":"auth0|66f9dd5b99d33f07238e0a45","user_metadata":{}}
```

### Додаткове завдання:

← Applications

## Create Application

Who is going to authenticate?



End users authenticate through my application  
Native or Web application



My application authenticates to access an API  
Machine to Machine authentication

Which API would your application access?

Auth0 Management API

Please select the API you want to authorize. Don't have an API yet? [Create a new API](#)

What permissions would you like to give to your applications?

Select: [All](#) | [None](#)

☐ read:client\_grants

☐ create:client\_grants

☐ delete:client\_grants

☐ update:client\_grants

☒ read:users

☐ update:users

☐ delete:users

☒ create:users

☐ read:users\_app\_metadata

☐ update:users\_app\_metadata

☐ delete:users\_app\_metadata

☐ create:users\_app\_metadata

☐ read:user\_custom\_blocks

☐ create:user\_custom\_blocks

☐ delete:user\_custom\_blocks

☐ create:user\_tickets

☐ read:clients

At least one permission must be selected

Name

Labwork2

This name is for your own reference. It's not visible to end users.

Рисунок 1.3 – Створення власного ендпоінту, де у користувача є права для створення та читання користувачів



```

import json
import http.client
from typing import Tuple

def get_access_token(url: str, payload: str) -> str:
    connection = http.client.HTTPSConnection(url)
    headers = {'content-type': "application/json"}
    connection.request("POST", "/oauth/token", payload, headers)
    data = connection.getresponse().read()
    return json.loads(data.decode("utf-8"))['access_token']

def create_user(url: str, access_token: str, payload: str) -> Tuple[str, int]:
    headers = {
        "Accept": "application/json",
        "Content-Type": "application/json",
        "Authorization": f"Bearer {access_token}"
    }

    connection = http.client.HTTPSConnection(url)
    connection.request("POST", "/api/v2/users", payload, headers)
    result = connection.getresponse()
    data = result.read()

    return data.decode("utf-8"), result.status

if __name__ == "__main__":
    url = "dev-hfzb6seuth5jesyp.eu.auth0.com"

    payload_token = json.dumps({

```

```

        "client_id": "MvGakD6bdVR0GezUBWgcEbBqtjegQopG",
                                                    "client_secret":
"D796h3AHQg-JJxsa2OiMmfbJ3tcW9qSowOY-FaDWQ0vxVdsGuLln1xNz45lUtW
Ro",
        "audience": "https://dev-hfzb6seuth5jesyp.eu.auth0.com/api/v2/",
        "grant_type": "client_credentials"
    })

    access_token = get_access_token("dev-hfzb6seuth5jesyp.eu.auth0.com",
payload_token)

    payload_user = json.dumps({
        "email": "denis.babich@test.com",
        "user_metadata": {},
        "blocked": False,
        "email_verified": False,
        "app_metadata": {},
        "given_name": "Denys",
        "family_name": "Babich",
        "name": "Denys",
        "nickname": "TestUser",
                                                    "picture":
"https://i.etsystatic.com/26254942/r/il/f54dcc/4754579609/il_300x300.4754579609_
aps7.jpg",
        "connection": "Username-Password-Authentication",
        "password": "BabichDenysPassword123@",
        "verify_email": False,
    })

    response_data, status_code = create_user(url, access_token, payload_user)
    print(f"Response Data: {response_data}")
    print(f"Status Code: {status_code}")

```



Вищезазначений код реалізує отримання токена доступу та створення нового користувача через API Auth0. Функція `get_access_token` виконує POST-запит до точки `/oauth/token`, передаючи необхідні дані для отримання токена доступу, який використовується для авторизації наступних запитів. Після цього функція `create_user` здійснює POST-запит до `/api/v2/users`, передаючи дані нового користувача, включаючи електронну адресу, ім'я, прізвище, пароль та інші метадані.

✓	API Operation	Create a User	25 minutes a...	N/A	N/A
---	---------------	---------------	-----------------	-----	-----

Рисунок 1.5 – Підтвердження успішної операції створення користувача у логах

У результаті запуску отримується такий вивід, який має статус-код 201, що свідчить про успішно-створеного користувача:

Response

Data:

```
{ "blocked": false, "created_at": "2024-09-29T23:42:30.424Z", "email": "denis.babich@test.com", "email_verified": false, "family_name": "Babich", "given_name": "Denys", "identities": [ { "connection": "Username-Password-Authentication", "user_id": "66f9e5e61d6c705c7c47b630", "provider": "auth0", "isSocial": false } ], "name": "Denys", "nickname": "TestUser", "picture": "https://i.etsystatic.com/26254942/r/il/f54dcc/4754579609/il_300x300.4754579609_aps7.jpg", "updated_at": "2024-09-29T23:42:30.424Z", "user_id": "auth0|66f9e5e61d6c705c7c47b630", "user_metadata": { } }
```

Status Code: 201

**Висновок:** У ході виконання лабораторної роботи було досліджено авторизаційний протокол OAuth2, зокрема механізм client\_credentials grant. Під час роботи були проаналізовані ключові аспекти реалізації даного протоколу, які включають отримання токена доступу для авторизації запитів до API. Використовуючи API Auth0, було успішно реалізовано запит на отримання токена типу Bearer, що надає можливість взаємодії з захищеними ресурсами. Крім того, у рамках лабораторної роботи було виконано створення нового користувача через API. Запит на створення користувача було здійснено з використанням отриманого токена, що підтвердило успішність операції через статус-код 201, який свідчить про те, що механізм авторизації працює коректно, дозволяючи створювати користувачів та виконувати інші операції.

У результаті аналізу були виявлені переваги використання OAuth2, зокрема можливість обмеження доступу до ресурсів без необхідності надання паролів користувачів. Проте, як і в будь-якій системі авторизації, залишаються потенційні ризики, пов'язані з управлінням токенами, які можуть бути скомпрометовані, якщо не дотримуватися належних заходів безпеки, таких як використання HTTPS, регулярна ротація токенів та контроль за їх використанням. Загалом, протокол OAuth2 є ефективним рішенням для організації авторизації, але потребує уважного підходу до забезпечення безпеки даних.