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

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

"Авторизаційний протокол OAuth2"

Виконав(ла)	ІП-13 Бабіч Денис	
	(шифр, прізвище, ім'я, по батькові)	
Перевірив(ла)	Соколовський В. В.	
	(шифр, прізвище, ім'я, по батькові)	

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 токену 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": "JIvC05c2IBHlAe2patn6l6q5H35qxti0",
    "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 токену

У цьому коді реалізується запит на отримання токена доступу за допомогою протоколу ОАuth 2.0. Далі задається змінна URL, що містить адрес веб-сервісу, до якого надсилається запит, у даному випадку це URL для отримання токена 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 Веагег, 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':

'eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsImtpZCI6IjVCZTIBZFhrMERaUjhmR1dZYjdkViJ9.eyJpc3MiOiJodHRwczovL2twaS5ldS5hdXRoMC5jb20vIiwic3ViIjoiSkl2Q081YzJJQkhsQWUycGF0bjZsNnE1SDM1cXh0aTBAY2xpZW50cyIsImF1ZCI6Imh0dHBzOi8va3BpLmV1LmF1dGgwLmNvbS9hcGkvdjIvIiwiaWF0IjoxNzI3NjUwNzA3LCJleHAiOjE3Mjc3MzcxMDcsInNjb3BlIjoicmVhZDp1c2VycyBjcmVhdGU6dXNlcnMiLCJndHkiOiJjbGllbnQtY3JlZGVudGlhbHMiLCJhenAiOiJKSXZDTzVjMklCSGxBZTJwYXRuNmw2cTVIMzVxeHRpMCJ9.fXnVM6clYkUp3HsYBPnvzsOgBWxP6rejao7depzsPu5wlQApwtnxuTsptevvY-YFfJjS9imIS9JQpV_T317nKSo9HGrpeZQy2E_2xiMl4ht1SLvjXQD9z-nDMUvaw0uz93uYgwCGCr5U1reBkaxhMmNLCjeOHXueyCMZUnsaVuuZg32S_apaRAOPgmx5ump4-4aBtu4Axk-o2WBxFEs6aPzMtOwu7zF3wCTZ5juHzH-rukKBrKXCl7CJmOX-q5JQjOe0DC5ZUwu9PfThsYju6Fer4vymJqdL7Swyz6chaC-83DdR90pecSh4lY2RLB5z2B12JOr0P0U9U_-KUl1QQg',

```
'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 заголовку 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.

2. Створити юзера з власним email в системі, використовуючи метод https://auth0.com/docs/api/management/v2#!/Users/post_users

```
nport json
        ort requests
URL = "https://kpi.eu.auth0.com/api/v2/users"
HEADERS = {
           "Content-Type": "application/json",
          "Accept": "application/json"
          "Authorization": "Bearer eyJhbGci0iJSUzI1NiIsInR5cCI6IkpXVCIsImtpZCI6IjVCZTlBZFhrMERaUjhmR1dZYjdkViJ9.
          eyJpc3MiOiJodHRwczovL2twaS5ldS5hdXRoMC5jb20vIiwic3ViIjoiSkl2Q081YzJJQkhsQWUycGF0bjZsNnE1SDM1cXh0aTBAY2>
          pŹW50cyIsImF1ZCI6Imh0dHBz0i8va3BpLmV1LmF1dGgwLmNvbS9hcGkvdjIvÌiwiaWF0IjoxNzÍ3NjUwNzA3LCJleHAi0jE3Mjc3MzcxMDcsInNjb3BlIjoicmVhZDp1c2VycyBjcmVhdGU6dXNlcnMiLCJndHki0iJjbGllbnQtY3JlZGVudGlhbHMiLCJhenAi0iJKSXZDT
          zVjMklCSGxBZTJwYXRuNmw2cTVIMzVxeHRpMCJ9.
          fXnVM6clYkUp3HsYBPnvzsOgBWxP6rejao7depzsPu5wlQApwtnxuTsptevvY-YFfJjS9imIS9JQpV_T317nKSo9HGrpeZQy2E_2xiM
          14 ht 15 LvjXQD9z-nDMUvaw0uz93uYgwCGCr5U1 reBkaxhMmNLCje0HXueyCMZUnsaVuuZg32S\_apaRA0Pgmx5ump4-4aBtu4Axk-o2Wallowered and the contraction of the 
          BxFEs6aPzMtOwu7zF3wCTZ5juHzH-rukKBrKXCl7CJmOX-q5JQj0e0DC5ZUwu9PfThsYju6Fer4vymJqdL7Swyz6chaC-83DdR90pec
          Sh4lY2RLB5z2B12J0r0P0U9U_-KUl1QQg'
PAYLOAD = json.dumps({
    "email": "denis.babich@mail.com",
           "user_metadata": {},
           "app_metadata": {},
          "given_name": "Denys",
"family_name": "Babich",
          "name": "Denys",
          "nickname": "Ngenx"
           "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 = requests.request("POST", URL, headers = HEADERS, data = PAYLOAD)
 print(response.text)
 print(response.status_code)
```

Рисунок 1.2 – Код для створення запиту на створення користувача Цей код виконує HTTP-запит для створення нового користувача через API Auth0, використовуючи метод POST. Спочатку задається URL-адресу API, заголовки, що включають тип вмісту, прийняті формати та авторизаційний токен. Далі код формується корисне навантаження у форматі JSON, яке містить електронну адресу, метадані користувача, підтвердження електронної адреси, дані про користувача, такі як ім'я, прізвище, нікнейм, посилання на зображення профілю, метод підключення та пароль. Після формування корисного навантаження код виконує POST-запит до API з зазначеними заголовками та даними.

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","i dentities":[{"connection":"Username-Password-Authentication","user_id":"66f9dd5b 99d33f07238e0a45","provider":"auth0","isSocial":false}],"name":"Denys","nicknam e":"Ngenx","picture":"https://i.etsystatic.com/26254942/r/il/f54dcc/4754579609/il_3 00x300.4754579609_aps7.jpg","updated_at":"2024-09-29T23:06:03.896Z","user_id":"auth0|66f9dd5b99d33f07238e0a45","user_metadata":{}}

Додаткове завдання:

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

← Applications

Create Application Who is going to authenticate? End users authenticate through my application My application authenticates to access an API **b** Native or Web application 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 Q Filter 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 read:user_custom_blocks delete:users_app_metadata create:users_app_metadata ☐ create:user custom blocks ☐ delete:user custom blocks ☐ create:user tickets ☐ read:clients At least one permission must be selected Name Labwork2

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

```
import http.client
connection = http.client.HTTPSConnection("dev-hfzb6seuth5jesyp.eu.auth0.com")
headers = { 'content-type': "application/json" }

payload = "{\"client_id\":\"MvGakD6bdVR0GezUBWgcEbBqtjegQopG\",
\"client_secret\":\"D796h3AHQg-JJxsa2OiMmfbJ3tcW9qSowOY-FaDWQ0vxVdsGuLln1xNz45lUtWRo\",
\"audience\":\"https://dev-hfzb6seuth5jesyp.eu.auth0.com/api/v2/\\",
\"grant_type\":\"client_credentials\"}"

connection.request("POST", "/oauth/token", payload, headers)

res = connection.getresponse()
data = res.read()
print(data.decode("utf-8"))
```

Рисунок 1.4 – Приклад коду, для звернення до ендпоінту та отримання токену Результат виконання коду з рисунку 1.4 можна побачити нижче:

"access_token":"eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsImtpZCI6IkxUUEVRen BxeDNkU3VYY2lyUGlHSSJ9.eyJpc3MiOiJodHRwczovL2Rldi1oZnpiNnNldXRoN Wplc3lwLmV1LmF1dGgwLmNvbS8iLCJzdWIiOiJNdkdha0Q2YmRWUjBHZXpV QldnY0ViQnF0amVnUW9wR0BjbGllbnRzIiwiYXVkIjoiaHR0cHM6Ly9kZXYtaG Z6YjZzZXV0aDVqZXN5cC5ldS5hdXRoMC5jb20vYXBpL3YyLyIsImlhdCI6MTcy NzY1MjQxNCwiZXhwIjoxNzI3NzM4ODE0LCJzY29wZSI6InJIYWQ6dXNlcnMg Y3JIYXRIOnVzZXJzIiwiZ3R5IjoiY2xpZW50LWNyZWRlbnRpYWxzIiwiYXpwIjo iTXZHYWtENmJkVIIwR2V6VUJXZ2NFYkJxdGplZ1FvcEcifQ.NvvIrY5H_xFESL_oiVYxbv0iwH_-7PnpqxMHgawk-VpK0lAhhZC3gcknxTicUOFYIyKu_fc_FDFM wexgMmlZXZ1IiKUkzxpziRGPVZ5xV1U02cgTNd4vEcLHwb0iDzREpKL-O48N7 M-K44_6LQ_gRlsNZg7i5caxlaib0RMJFwlbkmVjPw6C7roak84JswIWcVZNwJzIpb vJtlsv4xoi9FSDHOFNZ91erJIyPz6OZPZkQYLg0haUnOKE4ohTrIn_rWfB4uea2mo f94zIsk2-gKw8HOKe6N_vD3brwIqYkok_LZJIU12_6DEwe1WpnxGezvYo8QSKB 7ctPeHn0QgBYA",

{

[&]quot;scope":"read:users create:users",

[&]quot;expires_in":86400,

[&]quot;token_type":"Bearer"}

```
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.

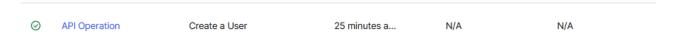


Рисунок 1.5 – Підтвердження успішної операції створення користувача у логах У результаті запуску отримується такий вивід, який має статус-код 201, що свідчить про успішно-створеного користувача:

Response Data: {"blocked":false,"created_at":"2024-09-29T23:42:30.424Z","email":"denis.babich@t est.com","email_verified":false,"family_name":"Babich","given_name":"Denys","ide ntities":[{"connection":"Username-Password-Authentication","user_id":"66f9e5e61d 6c705c7c47b630","provider":"auth0","isSocial":false}],"name":"Denys","nickname": "TestUser","picture":"https://i.etsystatic.com/26254942/r/il/f54dcc/4754579609/il_30 0x300.4754579609_aps7.jpg","updated_at":"2024-09-29T23:42:30.424Z","user_id": "auth0l66f9e5e61d6c705c7c47b630","user_metadata":{}}

Status Code: 201

лабораторної роботи Висновок: У ході виконання 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 АРІ. 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 АРІ. 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.