Міністерство освіти і науки України

Харківський національний університет імені В. Н. Каразіна

Факультет комп’ютерних наук

Спеціальність 125 «Кібербезпека»

Освітня програма «Кібербезпека»

«Допущено до захисту»

В. о. завідувача кафедрою БІСТ

Мелкозьорова О. М.

_____

« » 2024 р.

Пояснювальна записка

до кваліфікаційної роботи бакалавра

на тему: «Аналіз систем автентифікації та авторизації, побудованих на основі протоколу OAuth»

оцінка « »

Голова ЕК

Лемешко О. В. _____

Керівник д.т.н. Єсін В. І.

Рецензент к.т.н. Бобух В.А.

Виконавець: студент групи КБ-41

_____ Домбровський Р.О.

Харків – 2024

РЕФЕРАТ

Дипломна робота присвячена аналізу систем автентифікації та авторизації, побудованих на основі протоколу OAuth. Робота складається з 51 сторінки, 5 рисунків, 6 таблиць. У роботі використано 33 джерела.

Мета роботи - аналіз систем автентифікації та авторизації, побудованих на основі протоколу OAuth для надання практичних рекомендацій щодо усунення існуючих вразливостей та вдосконалення безпеки систем на основі OAuth.

Методи дослідження включають основні положення теорії складних систем, системного аналізу, теорії алгоритмів.

Результати роботи: досліджено протокол OAuth, проаналізовано низку систем на його основі, виявлено їх переваги та недоліки, розглянуто вразливості та найпоширеніший інцидент безпеки, надані практичні рекомендації щодо усунення існуючих вразливостей та вдосконалення безпеки систем на основі OAuth. Новизна досліджень полягає у комплексному порівнянні існуючих рішень.

Результати досліджень рекомендується використовувати для вибору оптимальної системи автентифікації та авторизації на основі OAuth, а також для підвищення безпеки таких систем.

Значущість роботи полягає у систематизації знань про системи на основі OAuth та виявленні їх сильних і слабких сторін, що дозволить зробити більш обґрунтований вибір рішення та підвищити рівень захисту.

Можливими напрямками подальших досліджень є аналіз нових систем OAuth, дослідження нових атак та методів захисту, розробка вдосконалених архітектур.

КЛЮЧОВІ СЛОВА: OAUTH, АВТЕНТИФІКАЦІЯ, АВТОРИЗАЦІЯ, БЕЗПЕКА, ПРОТОКОЛИ, СИСТЕМИ, АНАЛІЗ, ІНЦИДЕНТ.

ABSTRACT

The thesis is dedicated to the analysis of authentication and authorization systems based on the OAuth protocol. The work consists of 50 pages, 5 figures, and 6 tables. It uses 33 sources.

The goal of the work is to analyze authentication and authorization systems based on the OAuth protocol to provide practical recommendations for addressing existing vulnerabilities and improving the security of OAuth-based systems.

The research methods include the main principles of the theory of complex systems, systems analysis, and algorithm theory.

Results of the work: the OAuth protocol was studied, several systems based on it were analyzed, their advantages and disadvantages were identified, vulnerabilities and the most common security incident were examined, and practical recommendations for eliminating existing vulnerabilities and improving the security of OAuth-based systems were provided. The novelty of the research lies in the comprehensive comparison of existing solutions.

The research results are recommended for use in selecting the optimal authentication and authorization system based on OAuth, as well as for enhancing the security of such systems.

The significance of the work lies in the systematization of knowledge about OAuth-based systems and the identification of their strengths and weaknesses, which will allow for a more informed choice of solutions and an improved level of protection.

Possible directions for further research include the analysis of new OAuth systems, the study of new attacks and protection methods, and the development of improved architectures.

KEYWORDS: OAUTH, AUTHENTICATION, AUTHORIZATION, SECURITY, PROTOCOLS, SYSTEMS, ANALYSIS, INCIDENT.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

OAuth – Open Authorization (протокол авторизації)

URL – Uniform Resource Locator (уніфікований локатор ресурсів)

URI – Uniform Resource Identifier (уніфікований ідентифікатор ресурсу)

IoT – Internet of Things (Інтернет речей)

SSO – Single Sign-On (єдиний вхід)

XSS – Cross-Site Scripting (міжсайтовий скриптинг)

CSRF – Cross-Site Request Forgery (підробка міжсайтових запитів)

HTTPS – HyperText Transfer Protocol Secure (захищений протокол передачі гіпертексту)

SSL – Secure Sockets Layer (рівень захищених сокетів)

TLS – Transport Layer Security (захист транспортного рівня)

XSS – Cross-Site Scripting (міжсайтовий скриптинг)

LDAP – Lightweight Directory Access Protocol (легкий протокол доступу до каталогів)

RFC – Request for Comments (запит коментарів)

МФА – можливість багатофакторної автентифікації (multi-factor authentication)

API – Application Programming Interface (інтерфейс програмування додатків)

ЗМІ – Засоби масової інформації

США – Сполучені Штати Америки

GDPR – General Data Protection Regulation (Загальний регламент захисту даних)

ЄС – Європейський Союз

B2C – Business-to-Consumer

B2B - Business-to-Business

ВСТУП

Системи автентифікації та авторизації є критично важливими для забезпечення безпеки та конфіденційності даних у різноманітних застосунках та сервісах. Зі зростанням кількості кіберзагроз та випадків витоку конфіденційної інформації потреба у надійних та безпечних механізмах автентифікації та авторизації стає все більш потрібною. Одним із широко використовуваних підходів до реалізації таких систем є протокол OAuth.

Протокол OAuth був розроблений як відкритий стандарт для делегування дозволів на доступ до ресурсів користувача сторонніми застосунками, не розкриваючи облікових даних користувача. Він забезпечує безпечний спосіб надати доступ до веб-сервісів, таких як соціальні мережі, хмарні сховища даних та інші ресурси, що вимагають автентифікації. OAuth широко використовується провідними технологічними компаніями, такими як Google, Facebook, Microsoft та іншими.

Незважаючи на свою популярність, імплементації OAuth можуть мати вразливості, що створюють ризики для безпеки систем та даних користувачів. Аналіз існуючих систем автентифікації та авторизації на основі OAuth, їх переваг, недоліків, а також відомих інцидентів порушення безпеки є справді важливим.

Метою даної роботи є проведення аналізу систем автентифікації та авторизації, побудованих на основі протоколу OAuth, порівняння їх функціональності, безпеки та особливостей реалізації. У тому числі з проведенням аналізу одного з найпоширеніших інцидентів порушення безпеки таких систем (причини, наслідки та можливі шляхи удосконалення безпеки).

Результати дослідження можуть бути корисними для розробників програмного забезпечення, системних адміністраторів, фахівців з інформаційної безпеки та інших зацікавлених сторін, які працюють з системами автентифікації та авторизації на основі OAuth або планують їх впровадження.

1 ОСНОВНІ ПРИНЦИПИ СИСТЕМИ АВТОРИЗАЦІЇ OAUTH

1.1. Принципи роботи протоколу OAuth

OAuth - це відкритий протокол авторизації, який дозволяє користувачам надавати обмежений доступ до своїх ресурсів стороннім застосункам без необхідності розкривати власні облікові дані (логін та пароль). Ключовою ідеєю протоколу є делегування доступу, що забезпечує більш безпечний спосіб взаємодії між різними веб-сервісами та застосунками.

Протокол OAuth (Open Authorization) визначає набір термінів і понять, які є ключовими для розуміння його принципів роботи. Одним із основних понять є провайдер ресурсів (Resource Server) - це сервер, який зберігає захищені ресурси користувача, такі як дані профілю, електронна пошта, фотографії тощо. Він відповідає за забезпечення безпечного доступу до цих ресурсів та перевірку дійсності токенів доступу, наданих клієнтськими додатками.

Токен доступу (Access Token) - це рядок символів, що видається авторизаційним сервером після успішної автентифікації користувача та надання дозволу на доступ до ресурсів (рис. 1.1). Токен доступу використовується клієнтським додатком для отримання доступу до захищених ресурсів від провайдера ресурсів. Токени доступу мають обмежений термін дії з міркувань безпеки (від кількох годин до кількох тижнів).

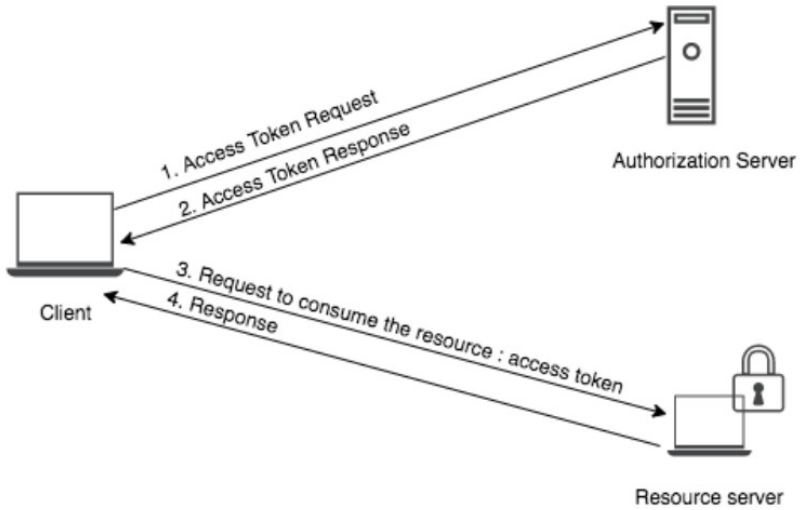

Рисунок 1.1 – Принцип роботи токена доступу

Токен оновлення (Refresh Token) - це додатковий рядок символів, який видається разом з токеном доступу. Він використовується для отримання нового токена доступу після закінчення терміну дії поточного токена, без необхідності повторної автентифікації користувача (рис. 1.2).

Рисунок 1.2 - Принцип роботи токена оновлення

Обсяг дозволів (Scope) - це набір прав, які визначають, до яких ресурсів клієнтський додаток матиме доступ. Користувач явно надає дозвіл на певний обсяг під час процесу авторизації. Це дозволяє реалізувати принцип мінімальних привілеїв, надаючи додаткам лише ті права, які їм необхідні для виконання своїх функцій [1].

У процесі авторизації за протоколом OAuth задіяні чотири основні ролі: власник ресурсу, клієнтський додаток, провайдер ресурсів та авторизаційний сервер.

Власник ресурсу (Resource Owner) - це користувач, який володіє захищеними даними, що зберігаються на провайдері ресурсів. Власник ресурсу надає дозвіл на доступ до своїх даних клієнтському додатку під час процесу авторизації.

Клієнтський застосунок (Client Application) - це програма, яка потребує доступу до захищених ресурсів користувача з метою надання певного сервісу або функціональності (рис. 1.3). Клієнтський додаток ініціює процес авторизації та використовує отримані токени доступу для взаємодії з

провайдером ресурсів. Клієнтські застосунки можуть бути веб-сайтами, мобільними застосунками, настільними програмами або серверними компонентами. Вони повинні бути зареєстровані на авторизаційному сервері та отримати унікальний ідентифікатор клієнта (client_id) і секретний ключ клієнта (client_secret).

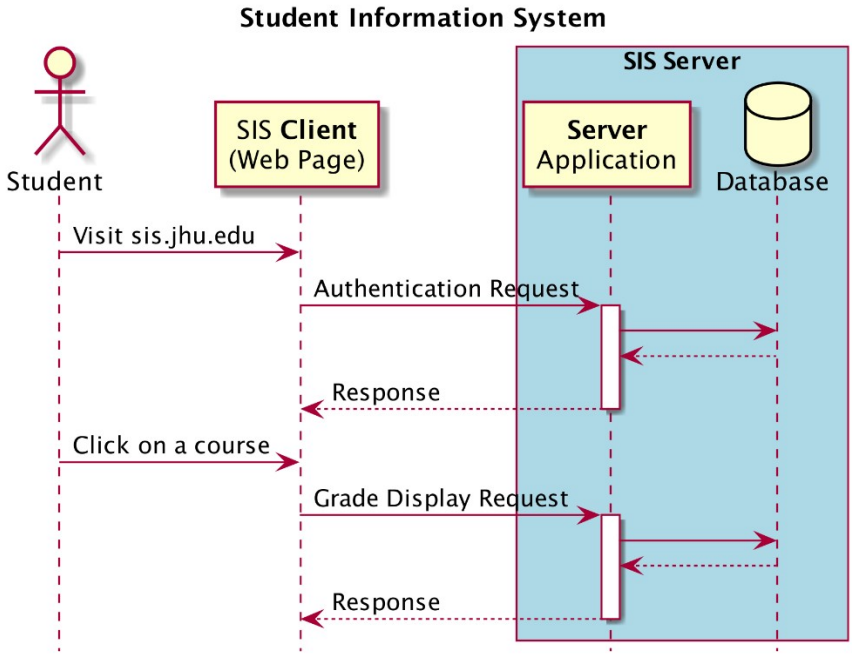

Рисунок 1.3 – Приклад додатку Client-Server Application

Перш ніж клієнтський додаток зможе використовувати протокол OAuth для отримання доступу до захищених ресурсів, він повинен пройти процес реєстрації на авторизаційному сервері. Під час реєстрації клієнтський додаток отримує унікальний ідентифікатор клієнта та секретний ключ клієнта.

Ідентифікатор клієнта використовується для ідентифікації клієнтського додатка під час взаємодії з авторизаційним сервером та провайдером ресурсів. Секретний ключ клієнта є конфіденційним і використовується для автентифікації клієнтського додатка під час обміну токенів або виконання певних типів запитів.

Під час реєстрації клієнтського додатка також можуть бути визначені такі параметри, як URI перенаправлення (redirect_uri), обсяги дозволів (scopes) та типи підтримуваних способів авторизації. Ці налаштування визначають

поведінку та права доступу клієнтського додатка в системі на основі OAuth [10,].

Належна реєстрація та налаштування клієнтських додатків є важливим кроком для забезпечення безпеки та належного функціонування систем на основі OAuth.

Після отримання токенів доступу та оновлення від авторизаційного сервера клієнтський додаток повинен забезпечити належне зберігання та управління цими токенами. Способи зберігання залежать від типу клієнтського додатка та вимог безпеки. Деякі підходи включають зберігання у пам'яті додатка, базі даних або використання спеціалізованих сховищ токенів.

Провайдер ресурсів у свою чергу повинен забезпечити безпечне зберігання токенів доступу, які були видані авторизаційним сервером. Це може бути реалізовано шляхом збереження токенів у захищеній базі даних або використання спеціалізованих сховищ токенів [11].

Провайдер ресурсів (Resource Server) - це сервер, який зберігає захищені ресурси користувача та надає доступ до них клієнтським додаткам після перевірки дійсності токенів доступу.

Авторизаційний сервер (Authorization Server) - це компонент, який відповідає за видачу токенів доступу та оновлення клієнтським додаткам після успішної автентифікації користувача та отримання його дозволу. Авторизаційний сервер також перевіряє дійсність наданих клієнтських ідентифікаторів і секретних ключів [2].

Взаємодія між цими ролями та компонентами відбувається відповідно до чітко визначених кроків протоколу OAuth, що забезпечує безпечний спосіб делегування доступу до ресурсів користувача.

Загальний робочий процес OAuth складається з кількох ключових кроків, які забезпечують безпечну авторизацію клієнтського додатка та надання йому доступу до захищених ресурсів користувача. Цей процес розпочинається з того, що клієнтський додаток ініціює запит на авторизацію, надсилаючи запит на авторизаційний сервер. У цьому запиті вказується ідентифікатор клієнта

(client_id), обсяг запитуваних дозволів (scope) та URL для перенаправлення (redirect_uri) [3].

Після цього авторизаційний сервер перенаправляє користувача на сторінку автентифікації, де користувач повинен ввести свої облікові дані для підтвердження особи. Якщо автентифікація пройшла успішно, авторизаційний сервер відображає користувачеві інформацію про клієнтський додаток та обсяг запитуваних дозволів. Користувач має можливість прийняти або відхилити цей запит на авторизацію.

У разі надання дозволу, авторизаційний сервер генерує код авторизації (authorization code) або токен доступу (access token), залежно від використовуваного способу авторизації. Цей код або токен повертається клієнтському додатку через URI перенаправлення.

Якщо було видано код авторизації, клієнтський додаток використовує його для отримання токена доступу та, у деяких випадках, токена оновлення (refresh token) від авторизаційного сервера. Токен доступу потім використовується клієнтським додатком для виконання запитів до провайдера ресурсів та отримання доступу до захищених ресурсів користувача.

Токен оновлення дозволяє клієнтському додатку отримувати новий токен доступу, коли поточний токен втратив чинність, уникаючи необхідності повторної авторизації користувача [3].

Протокол OAuth визначає кілька способів авторизації, які можуть бути використані залежно від типу клієнтського додатка та вимог безпеки. Розглянемо кожен з них.

Код авторизації (Authorization Code Flow) - найбільш поширений і безпечний спосіб, який використовується для веб-додатків і мобільних додатків з серверною частиною. У цьому способі клієнтський додаток отримує тимчасовий код авторизації, який потім обмінюється на токен доступу та токен оновлення.

Імпліцитний потік (Implicit Flow) - цей спосіб використовується для клієнтських додатків на основі JavaScript, які працюють у браузері. Токен

доступу видається безпосередньо клієнтському додатку через URI перенаправлення, без використання коду авторизації. Цей спосіб вважається менш безпечним, оскільки токен доступу передається через фрагмент URI.

Ресурсний власник пароля (Resource Owner Password Credentials Flow) - у цьому способі клієнтський додаток безпосередньо отримує токен доступу від авторизаційного сервера, надаючи облікові дані користувача (логін і пароль). Цей спосіб використовується лише в довірених клієнтських додатках, наприклад, у офіційних мобільних додатках компанії [4].

Облікові дані клієнта (Client Credentials Flow) – даний спосіб використовується для машина-машинної (M2M) автентифікації, коли клієнтський додаток потребує доступу до ресурсів без участі кінцевого користувача. Токен доступу видається на основі ідентифікатора клієнта та секретного ключа клієнта.

Вибір способу авторизації залежить від типу клієнтського додатка, вимог безпеки та особливостей реалізації протоколу OAuth у конкретній системі.

Протокол OAuth вважається безпечнішим і гнучкішим підходом до автентифікації та авторизації порівняно з традиційними методами, такими як базова автентифікація або використання сесійних cookie. Розглянемо його ключові переваги.

1. Захист облікових даних користувача. На відміну від базової автентифікації, де облікові дані (логін і пароль) передаються безпосередньо клієнтському додатку, при використанні OAuth користувач не розкриває свої облікові дані сторонньому додатку. Замість цього видається обмежений токен доступу до певних ресурсів.

2. Делегований доступ. OAuth дозволяє користувачам надавати сторонні додатки обмеженим доступом до своїх ресурсів без необхідності повної передачі облікових даних. Це зменшує ризики витоку конфіденційної інформації та забезпечує більший контроль з боку користувача [5].

3. Гнучкість і масштабованість. Протокол OAuth забезпечує гнучкий і масштабований підхід до автентифікації та авторизації в різноманітних

системах, від веб-додатків до мобільних застосунків та серверних компонентів. Він підтримує різні способи авторизації та сценарії використання.

4. Централізоване управління доступом. При використанні OAuth користувачі можуть централізовано керувати доступом до своїх ресурсів для різних сторонніх додатків через провайдера ресурсів. Це спрощує відкликання доступу або зміну дозволів для окремих додатків [6].

5. Мінімальні привілеї та обсяги дозволів. OAuth дозволяє реалізувати принцип мінімальних привілеїв за допомогою обсягів дозволів (scopes). Користувач може надати сторонньому додатку доступ лише до певного набору ресурсів, необхідного для виконання його функцій. Це зменшує ризики компрометації даних і підвищує рівень безпеки [5].

Протокол OAuth забезпечує гнучкий механізм для надання обмеженого доступу до ресурсів користувача через обсяги дозволів (scopes). Під час процесу авторизації клієнтський додаток запитує певний набір дозволів, необхідних для його функціонування. Користувач має можливість переглянути ці запитувані дозволи та явно надати або відхилити їх.

Обсяги дозволів можуть бути різними залежно від типу ресурсів і функціональності, яку потребує клієнтський додаток. Наприклад, додаток для редагування фотографій може запитувати дозвіл на доступ до фотогалереї користувача, але не буде потребувати доступу до його контактів або електронної пошти.

Цей механізм обсягів дозволів дозволяє реалізувати принцип мінімальних привілеїв, надаючи клієнтським додаткам лише ті права, які їм необхідні для виконання своїх функцій. Це зменшує ризики витоку конфіденційної інформації та компрометації даних користувача в разі компрометації клієнтського додатка.

Використання обсягів дозволів також надає користувачам більший контроль над доступом до їхніх ресурсів. Вони можуть явно вказати, які дозволи вони надають кожному окремому додатку, і в будь-який момент змінити або відкликати ці дозволи через провайдера ресурсів [7].

Механізм обсягів дозволів у протоколі OAuth є важливим елементом безпеки, який забезпечує реалізацію принципу мінімальних привілеїв та надає користувачам більший контроль над доступом до їхніх ресурсів.

1.2. Архітектура та компоненти систем на основі OAuth.

Протокол OAuth визначає набір ролей, компонентів та взаємодій між ними для забезпечення безпечного делегування доступу до захищених ресурсів користувача. Системи, побудовані на основі OAuth, мають чітку архітектуру, яка відображає ці ролі та компоненти, а також визначає їхні обов'язки та правила взаємодії.

Розуміння архітектури та компонентів систем на основі OAuth є важливим для ефективної реалізації та впровадження цього протоколу в різноманітних додатках та сервісах.

Система на основі протоколу OAuth складається з декількох ключових компонентів та ролей, які взаємодіють між собою для забезпечення безпечного делегування доступу до захищених ресурсів користувача. Ці ролі включають власника ресурсу, клієнтський додаток, провайдера ресурсів та авторизаційний сервер.

Провайдер ресурсів (Resource Server) - це сервер, який зберігає захищені ресурси користувача та надає доступ до них клієнтським додаткам після перевірки дійсності токенів доступу. Провайдер ресурсів відповідає за забезпечення безпеки та цілісності даних користувача.

Авторизаційний сервер (Authorization Server) - це компонент, який відповідає за автентифікацію власника ресурсу, отримання його дозволу на доступ до ресурсів та видачу токенів доступу та оновлення клієнтським додаткам. Авторизаційний сервер також перевіряє дійсність ідентифікаторів та секретних ключів клієнтських додатків [8].

Таблиця 1.1 – Характеристики ключових компонентів на основі протоколу OAuth

Компонент	Технології	Протоколи та стандарти	Безпека
Клієнтський додаток	Веб-технології, мобільні платформи	HTTPS, OAuth 2.0	Безпечне зберігання токенів, мінімізація прав доступу
Власник ресурсу	Інтерфейси користувача (UI)	OAuth 2.0	Паролі, двофакторна автентифікація
Провайдер ресурсів	Серверні технології (API)	OAuth 2.0, HTTPS	Захист API, контроль доступу, моніторинг запитів
Авторизаційний сервер	Серверні технології, база даних	OAuth 2.0, OpenID Connect	Шифрування токенів, захист від CSRF та інших атак, аудити безпеки

Взаємодія між цими компонентами відбувається відповідно до визначених кроків протоколу OAuth та забезпечує безпечний спосіб делегування доступу до ресурсів користувача.

Системи на основі OAuth можуть бути побудовані за централізованою або децентралізованою архітектурною моделлю. У централізованій моделі авторизаційний сервер та провайдер ресурсів є єдиним компонентом, який надає як функції автентифікації та авторизації, так і управління захищеними ресурсами користувача. Ця модель є більш простою у реалізації та адмініструванні, але може створювати єдину точку відмови [9].

У децентралізованій моделі авторизаційний сервер та провайдер ресурсів є окремими компонентами, які можуть бути розгорнуті та адмініструватися незалежно. Ця модель забезпечує більшу гнучкість та масштабованість, але вимагає додаткових заходів безпеки для захисту комунікації між компонентами.

Провайдер ресурсів у свою чергу повинен забезпечити безпечне зберігання токенів доступу, які були видані авторизаційним сервером. Це може бути реалізовано шляхом збереження токенів у захищеній базі даних або використання спеціалізованих сховищ токенів [11].

Важливим аспектом безпеки OAuth є механізми валідації токенів доступу та оновлення на провайдері ресурсів. Перш ніж надати доступ до захищених ресурсів, провайдер ресурсів повинен перевірити дійсність поданого токена. Це

включає перевірку терміну дії, його цілісності та відповідності клієнтському додатку та обсягам дозволів.

Провайдер ресурсів може використовувати різні методи для валідації токенів, такі як локальна перевірка з використанням спільного секретного ключа, віддалена перевірка запитом до авторизаційного сервера або використання криптографічних підписів [12].

В деяких складних архітектурах систем на основі OAuth можуть бути присутні додаткові компоненти, такі як сервер ресурсів користувача (Resource Owner Resource Server) або брокер токенів (Token Broker).

Сервер ресурсів користувача - це компонент, який зберігає захищені ресурси користувача та взаємодіє з провайдером ресурсів для надання доступу клієнтським додаткам. Він може бути використаний для розподілу навантаження або додаткового рівня безпеки.

Брокер токенів - це проміжний компонент, який виконує перетворення та проксування токенів між клієнтськими додатками, авторизаційним сервером та провайдером ресурсів. Це може бути корисним для інтеграції різних систем або додатків, що використовують різні формати токенів [13].

Архітектура на основі OAuth може бути реалізована у різних сценаріях використання, залежно від типу додатка та вимог безпеки.

У випадку використання веб-додатка клієнтська частина взаємодіє з авторизаційним сервером та провайдером ресурсів для отримання доступу до захищених ресурсів користувача, використовуючи коди авторизації або токени доступу.

Мобільні додатки можуть використовувати різні способи авторизації OAuth, такі як коди авторизації, імпліцитний потік або облікові дані власника ресурсу. Вони зберігають токени доступу та оновлення на пристрої користувача та використовують їх для взаємодії з провайдером ресурсів [14].

Використовуючи серверні компоненти, такі як мікросервіси або API, можуть бути задіяні облікові дані клієнта або власника ресурсу для отримання

токенів доступу. Ці компоненти взаємодіють безпосередньо з авторизаційним сервером та провайдером ресурсів для виконання авторизованих операцій.

У сценаріях Інтернету речей (IoT) пристрої можуть використовувати облікові дані клієнта або власника ресурсу для отримання токенів доступу, які потім використовуються для безпечного обміну даними з провайдером ресурсів або іншими пристроями.

Вибір архітектурної моделі та компонентів залежить від вимог безпеки, масштабованості та специфіки додатка, що реалізує систему на основі OAuth.

1.3 Процеси автентифікації та авторизації в OAuth

Протокол OAuth визначає чіткі процеси автентифікації та авторизації, які забезпечують безпечне делегування доступу до захищених ресурсів користувача для клієнтських додатків. Ці процеси включають декілька етапів взаємодії між власником ресурсу, клієнтським додатком, авторизаційним сервером та провайдером ресурсів.

Процес автентифікації користувача на авторизаційному сервері є ключовим етапом у забезпеченні безпеки та цілісності системи на основі OAuth. Цей процес розпочинається коли клієнтський додаток ініціює запит на авторизацію, переправляючи користувача на авторизаційний сервер. Авторизаційний сервер відображає сторінку автентифікації, де користувач повинен ввести свої облікові дані, зазвичай електронну пошту або ім'я користувача та пароль.

Методи автентифікації можуть різнитися залежно від реалізації авторизаційного сервера, але зазвичай використовуються традиційні методи, такі як автентифікація за паролем або через зовнішні системи. Наприклад через провайдерів ідентифікації, таких як Google, Facebook або інші служби єдиного входу (SSO) [15].

Після успішної автентифікації користувача авторизаційний сервер відображає інформацію про клієнтський додаток, який запитує доступ до ресурсів користувача. Ця інформація включає назву додатка, логотип,

розробника та, що найважливіше, обсяги дозволів (scopes), які додаток запитує. Обсяги дозволів визначають, до яких саме ресурсів користувача додаток бажає отримати доступ, наприклад, фотографії, контакти, електронна пошта тощо.

Користувач має можливість переглянути запитувані дозволи та прийняти рішення про надання або відхилення доступу клієнтському додатку. Якщо користувач надає дозвіл, авторизаційний сервер генерує код авторизації або токен доступу, залежно від використовуваного способу авторизації [16].

У випадку використання потоку коду авторизації (Authorization Code Flow), клієнтський додаток отримує тимчасовий код авторизації через URI перенаправлення. Цей код використовується для обміну на токен доступу та токен оновлення, якщо потрібно [17]. Клієнтський додаток надсилає запит на отримання токенів до авторизаційного сервера, передаючи код авторизації, ідентифікатор клієнта та секретний ключ клієнта (якщо доступний).

Таблиця 1.2. - Порівняння способів авторизації OAuth

Спосіб авторизації	Опис	Переваги	Недоліки
Код авторизації	Клієнтський додаток отримує тимчасовий код авторизації, який потім обмінюється на токен доступу та оновлення.	Високий рівень безпеки, підтримка конфіденційних клієнтів.	Потребує серверної частини для обміну коду на токени.
Імпліцитний потік	Токен доступу видається безпосередньо клієнтському додатку через URI перенаправлення.	Простота реалізації для клієнтських додатків.	Нижчий рівень безпеки, не підтримує конфіденційні клієнти.
Власник ресурсу	Клієнтський додаток отримує токен доступу, надаючи облікові дані користувача.	Простота для довірених клієнтів.	Низький рівень безпеки, потребує довіри користувача.
Облікові дані клієнта	Токен доступу видається на основі ідентифікатора клієнта та секретного ключа клієнта.	Корисний для машина-машинної автентифікації.	Низький рівень безпеки, не передбачає участі користувача.

Авторизаційний сервер перевіряє дійсність наданого коду авторизації та видає токени доступу та оновлення клієнтському додатку. Токен доступу використовується для виконання авторизованих запитів до провайдера ресурсів від імені користувача, тоді як токен оновлення дозволяє отримувати новий

токен доступу після закінчення терміну дії попереднього, без необхідності повторної автентифікації користувача.

Процеси автентифікації та авторизації в OAuth забезпечують безпечний і контрольований спосіб для клієнтських додатків отримувати доступ до захищених ресурсів користувача, дотримуючись принципів делегованого доступу та мінімальних привілеїв.

Після успішного отримання токена доступу від авторизаційного сервера клієнтський додаток може використовувати його для виконання авторизованих запитів до провайдера ресурсів. Токен доступу включається як частина заголовка запиту (наприклад, у заголовку "Authorization" або як параметр запиту) і дозволяє провайдеру ресурсів ідентифікувати користувача та перевірити наявність відповідних дозволів [19].

Перед наданням доступу до захищених ресурсів провайдер ресурсів повинен провести валідацію отриманого токена доступу. Цей процес включає декілька кроків для забезпечення безпеки та цілісності даних (рис. 1.4).

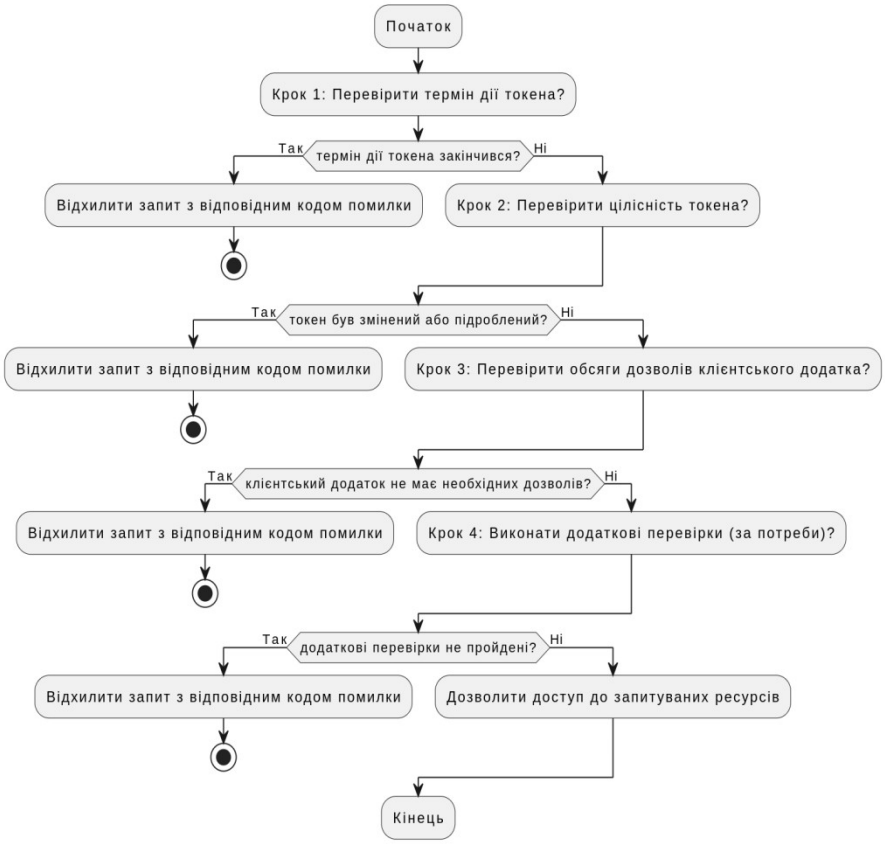

Рисунок 1.3 – Алгоритм перевірки токена доступу провайдером ресурсів

Цей алгоритм відображає основні кроки, які провайдер ресурсів виконує для перевірки дійсності та безпеки токена доступу перед наданням доступу до захищених ресурсів.

Якщо всі перевірки пройдені успішно, провайдер ресурсів надає клієнтському додатку доступ до запитуваних ресурсів відповідно до наданих обсягів дозволів.

Термін дії доступу є обмеженим з міркувань безпеки. Після закінчення терміну клієнтський додаток не може використовувати токен доступу для виконання авторизованих запитів. Однак протокол OAuth передбачає механізм оновлення (refresh token).

Токен оновлення є довгостроковим, зберігається клієнтським додатком і використовується для отримання нового токена доступу після закінчення терміну дії поточного. Клієнтський додаток надсилає запит до авторизаційного сервера, передаючи токен оновлення та ідентифікатор клієнта. Авторизаційний сервер перевіряє дійсність оновлення та видає новий токен доступу клієнтському додатку [21].

Протокол OAuth визначає кілька способів авторизації, які мають певні відмінності у своїх процесах.

Код авторизації (Authorization Code Flow) є найбільш поширеним і безпечним, оскільки ніколи не передається через браузер користувача. Клієнтський додаток отримує тимчасовий код авторизації, який потім обмінюється на токени доступу та оновлення на авторизаційному сервері.

Імпліцитний потік (Implicit Flow). Токен доступу видається безпосередньо клієнтському додатку через URI перенаправлення. Він не підтримує токени оновлення і вважається менш безпечним, ніж код авторизації.

Ресурсний власник пароля (Resource Owner Password Credentials Flow). Клієнтський додаток отримує токен доступу, надаючи облікові дані користувача (логін і пароль) авторизаційному серверу. Це вважається небезпечним і повинно використовуватися лише у довірених додатках.

Облікові дані клієнта (Client Credentials Flow). Токен доступу видається на основі ідентифікатора та секретного ключа клієнта, без участі користувача. Він зазвичай застосовується для машина-машинної автентифікації [22].

Вибір способу авторизації залежить від вимог безпеки, типу клієнтського додатка та специфіки реалізації системи на основі OAuth.

1.4 Безпека та проблеми OAuth

Незважаючи на те, що протокол OAuth був розроблений для забезпечення безпечного делегованого доступу до захищених ресурсів користувача, його реалізації не позбавлені певних проблем і потенційних вразливостей. Оскільки OAuth широко використовується багатьма великими компаніями та додатками, будь-які недоліки у його безпеці можуть мати серйозні наслідки для конфіденційності даних користувачів та цілісності систем.

Розглянемо загальні проблеми безпеки.

Протокол OAuth надає потужні можливості для авторизації та делегування доступу до ресурсів. Однак, як і будь-яка інша система безпеки, він не позбавлений певних ризиків та вразливостей. Розглянемо основні проблеми безпеки, пов'язані з OAuth.

Однією з найбільших загроз для систем на основі OAuth є витік облікових даних, таких як клієнтські секрети або токени доступу. Такі випадки можуть призвести до несанкціонованого доступу до захищених ресурсів та конфіденційних даних користувачів [6]. Крім того, існують ризики підробки токенів, атак переповнення, а також міжсайтових сценаріїв (XSS та CSRF), які можуть поставити під загрозу цілісність та конфіденційність системи. Проблеми конфіденційності також є актуальними, оскільки деякі реалізації OAuth можуть розкривати більше інформації про користувача, ніж це необхідно для надання доступу до ресурсів.

Специфічні вразливості різних способів авторизації OAuth.

Способи, такі як код авторизації, імпліцитний потік, ресурсний власник пароля та облікові дані клієнта, мають свої специфічні ризики та вразливості. Наприклад, імпліцитний потік вважається менш безпечним, оскільки токен доступу передається у фрагменті URL, що робить його більш вразливим до перехоплення. Спосіб ресурсного власника пароля може призвести до витоку облікових даних користувача, якщо клієнтський додаток не є достатньо захищеним

Ризики, пов'язані з неналежним зберіганням та управлінням токенами.

Неналежне зберігання та управління токенами доступу та оновлення на стороні клієнтського додатка та провайдера ресурсів також може призвести до серйозних проблем безпеки. Наприклад, зберігання токенів у незахищеному сховищі або передача їх через незахищені канали зв'язку може призвести до витоку конфіденційної інформації [3]. Крім того, відсутність належного управління терміном дії токенів та їх своєчасного оновлення може підвищити ризик їх компрометації.

Проблеми неправильної реалізації або конфігурації.

Проблеми безпеки часто виникають через неправильну реалізацію або конфігурацію компонентів системи, таких як авторизаційний сервер або провайдер ресурсів. Недотримання рекомендацій та найкращих практик безпеки, таких як використання захищених протоколів зв'язку (HTTPS), належне шифрування даних, перевірка вхідних даних та коректне налаштування параметрів безпеки, може призвести до виникнення вразливостей у системі.

Отже, протокол OAuth є потужним та гнучким рішенням для авторизації та делегування доступу до ресурсів у веб-додатках та сервісах. Він дозволяє користувачам надавати обмежений доступ до своїх даних сторонніми додатками без необхідності розкривати облікові дані. Основними принципами роботи OAuth є розподіл ролей між клієнтом, сервером ресурсів та авторизаційним сервером, а також використання токенів доступу для представлення дозволів користувача.

Архітектура систем на основі OAuth складається з кількох ключових компонентів, таких як клієнтський додаток, авторизаційний сервер, сервер ресурсів та ресурси, до яких потрібен доступ. Ці компоненти взаємодіють між собою через чітко визначені процеси автентифікації та авторизації, які включають отримання коду авторизації, обмін його на токен доступу та використання токена для доступу до ресурсів.

Хоча протокол OAuth забезпечує потужні можливості для авторизації та делегування доступу, він також має певні ризики та виклики з точки зору безпеки. Основними проблемами є витік облікових даних, підробка токенів, міжсайтові сценарії (XSS та CSRF), а також ризики, пов'язані з неналежним зберіганням та управлінням токенами. Крім того існують специфічні вразливості різних способів авторизації OAuth, таких як імпліцитний потік або ресурсний власник пароля. Неправильна реалізація або конфігурація компонентів системи також може призвести до виникнення вразливостей безпеки.

Для зміцнення безпеки систем на основі OAuth необхідно дотримуватися кращих практик реалізації, конфігурації та адміністрування, таких як використання захищених протоколів зв'язку, належне шифрування даних, регулярні оновлення та аудити безпеки. Крім того, важливо враховувати нові тенденції та виклики безпеки, пов'язані з розвитком технологій, таких як мобільні пристрої, Інтернет речей та хмарні обчислення.

2 АНАЛІЗ ТА ПОРІВНЯННЯ ІСНУЮЧИХ СИСТЕМ АВТЕНТИФІКАЦІЇ ТА АВТОРИЗАЦІЇ, ПОБУДОВАНИХ НА ОСНОВІ OAUTH

2.1. Критерії порівняння систем.

Для аналізу та порівняння існуючих систем автентифікації та авторизації на основі OAuth необхідно визначити чіткі критерії оцінки. Ці критерії допоможуть структурувати порівняння, виділити ключові аспекти систем та забезпечити об'єктивну основу для виявлення переваг і недоліків різних реалізацій.

Безпека та захист даних є найважливішим критерієм для будь-якої системи автентифікації та авторизації, зокрема тих, що побудовані на основі протоколу OAuth. Ключовими аспектами безпеки є використані механізми автентифікації та авторизації, підтримка протоколів безпеки, захист від різноманітних атак, а також процедури перевірки та підтвердження ідентичності.

Фундаментальним елементом безпеки в системах OAuth є механізми автентифікації та авторизації. Згідно зі специфікацією протоколу OAuth 2.0 [1], автентифікація користувачів здійснюється за допомогою власника ресурсу (наприклад, користувача кінцевої системи). Авторизація ж визначає привілеї доступу, які надаються клієнтському додатку до захищених ресурсів власника. Для автентифікації можуть використовуватися різні методи, такі як паролі, одноразові паролі, сертифікати, біометричні дані тощо, в залежності від рівня безпеки, який потрібен для конкретної системи.

Важливим аспектом безпеки є підтримка протоколів безпеки для захисту передачі даних між різними компонентами системи OAuth. Зазвичай для цього використовується протокол HTTPS з використанням SSL/TLS для шифрування трафіку [3]. Це захищає конфіденційну інформацію, таку як токени доступу та оновлення, коди авторизації та інші чутливі дані від перехоплення та несанкціонованого доступу.

Окрім забезпечення конфіденційності даних, важливо також захиститися від різноманітних атак, які можуть бути спрямовані на систему автентифікації та авторизації. Серед них: атаки міжсайтового підроблення запитів (CSRF), міжсайтового скриптингу (XSS), відкритого редиректу (Open Redirector) тощо [7]. Для протидії таким атакам використовуються різні механізми захисту, такі як токени CSRF, фільтрація вхідних даних, перевірка джерела переадресацій та інші.

Також важливим елементом є процедури перевірки та підтвердження ідентичності користувачів у системах OAuth. Це може включати використання багатофакторної автентифікації, перевірку облікових даних за допомогою зовнішніх систем ідентифікації (наприклад, OpenID Connect), процедури підтвердження електронної пошти або номера телефону тощо. Такі заходи допомагають запобігти несанкціонованому доступу та забезпечити належний рівень безпеки даних користувачів.

Зручність використання та інтеграції є важливим фактором, що визначає простоту впровадження та взаємодії з системою. Наявність бібліотек та фреймворків для різних мов програмування та платформ значно спрощує інтеграцію OAuth в існуючі додатки та системи. Провідні реалізації OAuth, такі як Google, Facebook, Microsoft тощо, пропонують офіційні бібліотеки для популярних мов, наприклад, Java, Python, Node.js, .NET. Крім того, існує багато сторонніх бібліотек, створених спільнотою розробників, що покривають менш поширені мови та платформи.

Якість документації та підтримки з боку постачальників систем OAuth також відіграє важливу роль у зручності використання. Детальна та зрозуміла документація, наявність прикладів коду, керівництв та активної спільноти користувачів значно полегшують процес вивчення та впровадження системи [11].

Гнучкість та налаштовуваність дозволяють адаптувати систему автентифікації та авторизації до конкретних вимог та сценаріїв використання. Можливість налаштовувати параметри безпеки та дозволи, такі як строки дії

токенів, обсяги доступу, типи клієнтів тощо, є важливою для забезпечення належного рівня безпеки та контролю. Підтримка різних сценаріїв використання, таких як бізнес-для-споживача (B2C), бізнес-для-бізнесу (B2B) або Інтернету речей (IoT), дозволяє застосовувати систему OAuth в різноманітних галузях та контекстах.

Можливість інтеграції з іншими системами, такими як системи ідентифікації (наприклад, LDAP, Active Directory), бази даних користувачів або токенів, є важливою для створення комплексних рішень автентифікації та авторизації [9]. Така інтеграція може полегшити керування ідентичностями, дозволами та забезпечити єдину точку входу для різних додатків та систем.

Відповідність стандартам та наявність сертифікації також є критерієм, що свідчить про надійність та безпеку системи OAuth. Дотримання специфікацій та рекомендацій протоколу OAuth, таких як RFC 6749 [1] та інших офіційних документів, гарантує сумісність та правильну реалізацію основних принципів та процедур.

Крім того, наявність сертифікації або перевірки відповідності стандартам безпеки, таким як OAuth 2.0 Threat Model and Security Considerations, підтверджує, що система пройшла ретельне тестування та аудит на предмет наявності вразливостей та слабких місць. Це може бути важливим фактором для організацій, які працюють у сферах з високими вимогами до безпеки, таких як фінанси, охорона здоров'я або державний сектор.

2.2. Аналіз існуючих систем на основі OAuth

На ринку існує безліч систем автентифікації та авторизації, побудованих на основі протоколу OAuth. Провідні технологічні компанії, такі як Google, Facebook, Microsoft, Amazon, а також численні сторонні постачальники пропонують власні реалізації OAuth для різноманітності їхнього використання.

Google Sign-In - це набір API та бібліотек від Google для інтеграції автентифікації Google у веб-додатки, мобільні додатки та сервери бекенду.

Архітектура базується на стандартних компонентах OAuth 2.0: авторизаційний сервер Google, провайдер ресурсів та клієнтські додатки. Підтримуються різні потоки OAuth, включаючи авторизаційний код, імпліцитний та пароль ресурсного власника [1]. Google Sign-In пропонує традиційну автентифікацію за обліковим записом Google, OpenID Connect та соціальні ідентифікатори [13, с. 59].

Подібну архітектуру та принципи роботи мають інші популярні системи, такі як Facebook Login та Microsoft Identity Platform. Facebook Login призначений для веб-додатків, мобільних додатків, а також підтримує автентифікацію для Інтернету речей (IoT) та додатків з'єднаних пристроїв [16]. Він підтримує ті самі основні потоки OAuth 2.0 та механізми автентифікації, включаючи традиційну автентифікацію облікових записів Facebook, OpenID Connect та соціальні ідентифікатори.

Microsoft Identity Platform, раніше відома як Azure Active Directory, є комплексним рішенням для автентифікації та авторизації для різноманітних додатків, включаючи веб, мобільні, єдині облікові записи (Single Sign-On) та хмарні служби [17, с. 61]. Вона базується на OAuth 2.0 та OpenID Connect, підтримуючи різні потоки авторизації та можливості федерації ідентичності.

Ще одним провідним постачальником системи на основі OAuth є Amazon з їх сервісом Amazon Cognito. Cognito орієнтований на масштабовані мобільні та веб-додатки, забезпечуючи автентифікацію користувачів, керування ідентичністю та авторизацію. Він підтримує традиційну автентифікацію, соціальні ідентифікатори, а також пропонує власні механізми керування доступом і безпекою.

Багато сторонніх постачальників також пропонують системи автентифікації та авторизації на основі OAuth, орієнтовані на конкретні галузі чи сценарії використання. Наприклад, Auth0, Okta, Ping Identity та інші фокусуються на корпоративних додатках, забезпечуючи функції єдиного входу (Single Sign-On), багатофакторну автентифікацію та інтеграцію з існуючими системами ідентифікації організацій [19].

Щодо безпеки, більшість провідних систем дотримуються передових практик, використовуючи HTTPS, токени CSRF, механізми перевірки ідентичності та відповідають рекомендаціям безпеки OAuth 2.0 [10]. Інколи системи можуть мати специфічні особливості або додаткові заходи безпеки. Наприклад, додатковий аналіз загроз чи спеціальні механізми зберігання токенів.

Безпека та захист даних є ключовим пріоритетом для систем автентифікації та авторизації на основі OAuth. Більшість систем, дотримуються передових практик безпеки, включаючи використання протоколу HTTPS для захисту конфіденційної інформації під час передачі [20]. Для запобігання атакам міжсайтового підроблення запитів вони використовують захисні токени CSRF або інші еквівалентні механізми.

Дані системи застосовують різноманітні методи перевірки ідентичності користувачів, такі як двофакторна автентифікація, перевірка пристроїв, біометричні дані тощо. Деякі постачальники пропонують додаткові рівні безпеки, наприклад, Microsoft Identity Platform має можливість використовувати розширені політики умовного доступу, що дозволяє налаштовувати вимоги до автентифікації на основі різних факторів, таких як місце розташування, пристрій або додаток [17, с. 68].

Підтримка різних платформ та мов програмування є важливим фактором для зручності інтеграції та використання. Провідні постачальники пропонують широкий вибір офіційних бібліотек та фреймворків для популярних мов програмування та платформ. Наприклад, Google Sign-In має бібліотеки для Android, iOS, JavaScript, .NET, Java, Python, Ruby та Go, а Facebook Login - для Android, iOS, JavaScript, PHP, Python та інших. Microsoft Identity Platform також має бібліотеки для .NET, Java, JavaScript, Python, Ruby та Node.js [22].

Крім офіційних бібліотек, існує безліч сторонніх реалізацій та фреймворків, створених спільнотою розробників. Постачальники також забезпечують докладну документацію, керівництва та приклади коду для полегшення процесу інтеграції та налаштування.

Масштабованість та продуктивність є критично важливими для систем автентифікації та авторизації, особливо коли йдеться про обробку великої кількості запитів. Постачальники, такі як Google, Facebook, Microsoft та Amazon, мають високомасштабовані та розподілені системи, здатні обробляти мільярди запитів щодня від різноманітних додатків та сервісів.

Наприклад, Facebook Login використовує високопродуктивну архітектуру з балансуванням навантаження та репліками серверів для забезпечення надійності та масштабованості. Microsoft Identity Platform також спроектована для обробки великої кількості запитів, використовуючи хмарну інфраструктуру Microsoft Azure [17, с. 74].

Додаткові функції та можливості можуть відрізнятися між різними системами автентифікації та авторизації на основі OAuth. Деякі з них забезпечують інтеграцію з іншими системами ідентифікації, такими як LDAP, Active Directory або провайдерами соціальних ідентифікаторів. Наприклад, Microsoft Identity Platform тісно інтегрується з Active Directory, що дозволяє організаціям використовувати існуючі облікові записи користувачів [17, с. 65].

Системи можуть підтримувати різні сценарії використання, такі як бізнес-для-споживача (B2C), бізнес-для-бізнесу (B2B), Інтернет речей (IoT) тощо. Amazon Cognito, наприклад, орієнтований на масштабовані мобільні та веб-додатки, забезпечуючи гнучке керування ідентичністю та доступом.

Провідні системи автентифікації та авторизації на основі OAuth пропонують широкий спектр можливостей, високу безпеку та зручність інтеграції. Проте, вибір конкретної системи залежатиме від вимог конкретного проекту, таких як цільова платформа, рівень безпеки, масштабованість та необхідні додаткові функції.

2.3. Порівняння функціональності та особливостей систем, побудованих на основі OAuth

Порівняння підтримуваних потоків OAuth та сценаріїв їх застосування є важливим аспектом аналізу різних систем автентифікації та авторизації. Згідно зі специфікацією OAuth, визначено чотири основні потоки.

1. Потік авторизаційного коду - найбільш поширений для веб-додатків, що працюють на серверах.

2. Імпліцитний потік - часто використовується для JavaScript-додатків у браузері.

3. Потік пароля ресурсного власника - застосовується для довірених додатків, таких як мобільні додатки.

4. Потік клієнтських креденціалів - для взаємодії між серверними компонентами.

Таблиця 2.1 - Порівняння основних потоків авторизації в OAuth

Потік	Переваги	Недоліки
Авторизаційний код	Безпечний, відокремлює клієнт та секрет клієнта. Підтримка оновлення токенів.	Відносно складний процес з багатьма кроками
Імпліцитний	Простий процес без додаткових кроків обміну. Немає потреби в серверній частині додатка.	Токен передається у відкритому вигляді, що створює ризики. Відсутня підтримка оновлення токенів.
Пароль ресурсного власника	Простий процес, не вимагає додаткових кроків. Підходить для довірених додатків.	Передбачає обмін обліковими даними користувача, що несе ризики. Не підтримується для веб-додатків.
Клієнтські креденціали	Безпечний процес для серверної взаємодії без участі користувача.	Не підходить для автентифікації користувачів. Необхідно забезпечити безпеку секрету клієнта.

Усі провідні системи, такі як Google Sign-In, Facebook Login, Microsoft Identity Platform та Amazon Cognito, підтримують перші три потоки, проте імпліцитний потік поступово виходить з використання через потенційні ризики безпеки.

Щодо механізмів автентифікації та авторизації, більшість систем пропонують традиційну автентифікацію за паролем, OpenID Connect, а також підтримку соціальних ідентифікаторів. Google Sign-In, Facebook Login та Microsoft Identity Platform дозволяють користувачам входити за допомогою облікових записів Google, Facebook або Microsoft відповідно, а також використовувати свої облікові записи для автентифікації в інших додатках [13].

Постачальники систем часто забезпечують можливість багатофакторної автентифікації для підвищення рівня безпеки. Наприклад, Google Sign-In

підтримує МФА з використанням одноразових кодів, біометричних даних або безпечних апаратних ключів. Microsoft Identity Platform також має потужні можливості МФА, включаючи SMS, голосові виклики, push-сповіщення та підтримку сторонніх провайдерів [17, с. 70].

Щодо заходів безпеки, всі провідні системи вимагають використання HTTPS для захисту конфіденційних даних під час передачі та застосовують токени CSRF або інші механізми для запобігання атакам міжсайтового підроблення запитів [20]. Вони також використовують різноманітні методи перевірки ідентичності, такі як перевірка пристроїв, біометричні дані, двофакторна автентифікація тощо.

Деякі системи мають додаткові функції безпеки. Наприклад, Microsoft Identity Platform пропонує розширені політики умовного доступу, що дозволяють налаштовувати вимоги до автентифікації на основі різних факторів, таких як місце розташування, пристрій або додаток [17, с. 68]. Amazon Cognito також має вбудовані функції безпеки, такі як захист від ботів, моніторинг підозрілої активності та можливість налаштовувати політики доступу [26].

У таблиці 2.2 наведено порівняння ключових особливостей безпеки та функціональності основних систем автентифікації та авторизації на основі OAuth.

Як видно з таблиці, хоча всі системи підтримують основні потоки OAuth та механізми автентифікації, вони можуть відрізнятися за додатковими функціями безпеки та гнучкістю налаштування політик доступу.

Підтримка різних платформ та мов програмування є важливою для зручності інтеграції та широкого застосування цих систем. Провідні постачальники пропонують офіційні бібліотеки та фреймворки для популярних мов та платформ, а також докладну документацію та ресурси для полегшення процесу впровадження.

Google Sign-In має офіційну підтримку для Android, iOS, JavaScript, .NET, Java, Python, Ruby, Go та інших мов [21]. Facebook Login також забезпечує бібліотеки для Android, iOS, JavaScript, PHP, Python та інших [22]. Microsoft

Identity Platform пропонує офіційні бібліотеки для .NET, Java, JavaScript, Python, Ruby та Node.js [23]. Крім офіційних бібліотек, існує багато сторонніх реалізацій, створених спільнотою розробників.

Таблиця 2.2 - Порівняння ключових особливостей безпеки та функціональності

Система	Підтримувані потоки OAuth	Механізми автентифікації	Заходи безпеки
Google Sign-In	Авторизаційний код, імпліцитний, пароль власника	Традиційна, OpenID Connect, соціальні ідентифікатори, МФА	HTTPS, токени CSRF, перевірка пристроїв, біометрія
Facebook Login	Авторизаційний код, імпліцитний, пароль власника	Традиційна, OpenID Connect, соціальні ідентифікатор	HTTPS, токени CSRF, додаткові заходи безпеки
Microsoft Identity Platform	Авторизаційний код, імпліцитний, пароль власника	Традиційна, OpenID Connect, соціальні ідентифікатори	HTTPS, токени CSRF, додаткові заходи безпеки
Amazon Cognito	Авторизаційний код, імпліцитний, пароль власника	Традиційна, OpenID Connect, соціальні ідентифікатори, МФА	HTTPS, захист від ботів, моніторинг активності
Facebook Login	Авторизаційний код, імпліцитний, пароль власника	Традиційна, OpenID Connect, соціальні ідентифікатор	HTTPS, токени CSRF, додаткові заходи безпеки
Microsoft Identity Platform	Авторизаційний код, імпліцитний, пароль власника	Традиційна, OpenID Connect, соціальні ідентифікатори	HTTPS, токени CSRF, додаткові заходи безпеки
Amazon Cognito	Авторизаційний код, імпліцитний, пароль власника	Традиційна, OpenID Connect, соціальні ідентифікатори, МФА	HTTPS, захист від ботів, моніторинг активності

Постачальники також забезпечують докладну документацію, керівництва, приклади коду та активні спільноти користувачів, що полегшує процес навчання та інтеграції. Наприклад, Google та Microsoft мають обширні навчальні ресурси та форуми підтримки.

Масштабованість та продуктивність є критично важливими для систем автентифікації та авторизації, особливо коли йдеться про обробку великої кількості запитів. Провідні постачальники, такі як Google, Facebook, Microsoft та Amazon, мають високомасштабовані та розподілені системи, здатні обробляти мільярди запитів щодня від різноманітних додатків та сервісів.

Facebook Login використовує високопродуктивну архітектуру з балансуванням навантаження та репліками серверів для забезпечення надійності та масштабованості. Microsoft Identity Platform також спроектована

для обробки великої кількості запитів, використовуючи хмарну інфраструктуру Microsoft Azure та технології масштабування.

Google Sign-In та Amazon Cognito також є високомасштабованими рішеннями, спроектованими для обслуговування величезної кількості користувачів та додатків з усього світу.

Додаткові функції та можливості можуть відрізнятися між різними системами. Деякі з них забезпечують інтеграцію з іншими системами ідентифікації, такими як LDAP або Active Directory. Наприклад, Microsoft Identity Platform тісно інтегрується з Active Directory, що дозволяє організаціям використовувати існуючі облікові записи користувачів.

Системи також можуть підтримувати різні сценарії використання, такі як бізнес-для-споживача (B2C), бізнес-для-бізнесу (B2B) або Інтернет речей (IoT). Amazon Cognito, наприклад, орієнтований на масштабовані мобільні та веб-додатки, забезпечуючи гнучке керування ідентичністю та доступом [24]. Microsoft Identity Platform також має потужну підтримку сценаріїв B2C та B2B.

Гнучкість керування ідентичністю та доступом також є важливою перевагою деяких систем. Google Sign-In дозволяє налаштовувати обсяги доступу до даних користувачів та API Google для клієнтських додатків. Microsoft Identity Platform пропонує розширені можливості керування доступом, включаючи політики умовного доступу, ролі та привілеї.

Провідні системи автентифікації та авторизації на основі OAuth мають широкі можливості інтеграції, масштабованість та додаткові функції, що дозволяє адаптувати їх до різноманітних вимог та сценаріїв використання.

2.4. Переваги та недоліки різних підходів до реалізації OAuth

Будь-які системи мають як переваги, так і недоліки. Розглянемо їх на прикладі систем, огляд яких здійснюється в даній роботі.

Таблиця 2.2 – Переваги та недоліки різних підходів до реалізації OAuth

Система	Переваги	Недоліки
Google Sign-In	Тісна інтеграція з численними продуктами Google, такими як Gmail, Google Drive, Google Maps тощо. - Зручність використання для користувачів, які вже мають обліковий запис Google. - Високий рівень безпеки завдяки використанню передових технологій автентифікації та шифрування від Google. - Підтримка різних протоколів автентифікації, включаючи OAuth 2.0 та OpenID Connect. - Легка інтеграція з Android та веб-додатками за допомогою бібліотек Google APIs.	- Залежність від Google та їх політик і правил. - Обмежена гнучкість налаштування процесу автентифікації та авторизації. - Обмежений контроль над даними користувачів, які зберігаються в Google.
Facebook Login	- Величезна база користувачів Facebook, що охоплює мільярди людей по всьому світу. - Легка інтеграція для розробників завдяки добре задокументованим інструментам розробника та бібліотекам. - Соціальні функції, такі як можливість ділитися активністю в додатку з друзями у Facebook. - Підтримка різних протоколів автентифікації, включаючи OAuth 2.0 та OpenID Connect.	- Занепокоєння щодо конфіденційності даних та використання персональної інформації користувачів для цілей реклами та цільового маркетингу. - Складність для користувачів, які не мають облікового запису Facebook або не хочуть використовувати його. - Залежність від Facebook та їх політик і правил.
Microsoft Identity Platform	- Тісна інтеграція з продуктами Microsoft, такими як Office 365, Azure та Windows. - Потужні можливості керування ідентифікацією та доступом, включаючи підтримку багатофакторної автентифікації, умовного доступу та керування життєвим циклом ідентифікації. - Підтримка корпоративних політик і вимог щодо безпеки та конфіденційності. - Можливість інтегрувати з локальними каталогами, такими як Active Directory.	- Складність налаштування та інтеграції, особливо для невеликих організацій або проектів. - Потенційно вища вартість для деяких підприємств у порівнянні з іншими рішеннями. - Залежність від продуктів Microsoft, що може бути проблемою для організацій, які використовують інші платформи.
Amazon Cognito	- Легкість налаштування та інтеграції з веб-додатками та мобільними додатками. - Масштабованість та висока доступність завдяки використанню хмарної інфраструктури AWS. - Тісна інтеграція з іншими сервісами AWS, такими як AWS Lambda, Amazon API Gateway та Amazon S3. - Підтримка соціальних провайдерів ідентифікації, таких як Google, Facebook та Amazon. - Можливість створювати власні пули користувачів та керувати ними.	- Залежність від AWS та їх політик і правил. - Відсутність деяких розширених функцій керування ідентифікацією та доступом, доступних у Microsoft Identity Platform. - Потенційно вища вартість для деяких організацій у порівнянні з іншими рішеннями.

Усі системи мають свої сильні та слабкі сторони, і вибір найкращого рішення залежить від конкретних потреб проекту. Google Sign-In та Facebook Login пропонують зручність для користувачів і легку інтеграцію для розробників, але можуть викликати занепокоєння щодо конфіденційності даних і залежності від відповідних компаній. Microsoft Identity Platform забезпечує потужні можливості керування ідентифікацією та доступом, але може бути складним у налаштуванні та інтеграції, а також дорожчим для деяких організацій. Amazon Cognito надає легкість налаштування та інтеграції, масштабованість і тісну інтеграцію з AWS, але може бракувати деяких розширених функцій і також залежить від AWS.

Вибір підходу до реалізації OAuth залежить від конкретних вимог та обставин проекту, а також від існуючої інфраструктури та інструментів, які використовуються в організації. Важливо ретельно оцінити потреби проекту, розглянути питання безпеки, конфіденційності даних, масштабованості та витрат на впровадження та підтримку обраної системи.

3 АНАЛІЗ ДЕЯКОГО ПОШИРЕНОГО ІНЦИДЕНТУ ПОРУШЕННЯ БЕЗПЕКИ СИСТЕМ АВТЕНТИФІКАЦІЇ ТА АВТОРИЗАЦІЇ, ПОБУДОВАНИХ НА ОСНОВІ OAUTH

3.1. Аналіз інциденту на конкретному прикладі.

Проаналізуємо один із гучниих інцидентів порушення безпеки систем автентифікації та авторизації, побудованих на основі OAuth, які мали місце в останні роки. Зокрема, варто розглянути витік даних користувачів Facebook, пов'язаний з Cambridge Analytica у 2018 році. Цей інцидент привернув значну увагу громадськості та висвітлив проблеми, пов'язані з безпекою та конфіденційністю даних користувачів у соціальних мережах.

Інцидент з витоком даних користувачів Facebook, пов'язаний з Cambridge Analytica, розгорнувся у 2018 році і набув широкого розголосу. Компанія Cambridge Analytica, яка спеціалізувалася на політичному консалтингу та аналізі даних, незаконно отримала доступ до особистих даних близько 87 мільйонів користувачів Facebook (рис. 3.1). Це сталося через зловживання функціями OAuth, які дозволяють сторонньому програмному забезпеченню отримувати обмежений доступ до даних користувачів за їхньою згодою.

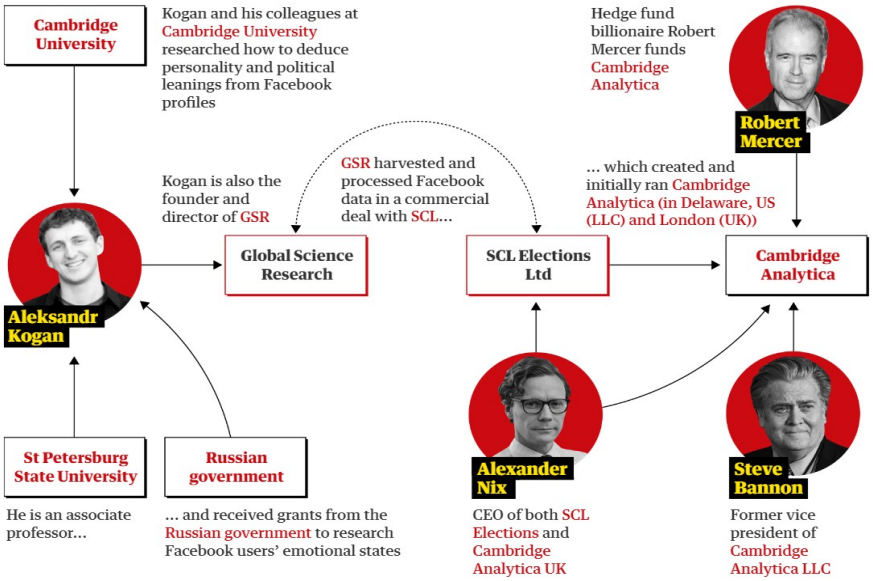

Рисунок 3.1 – Схема, як пов’язані між собою ключові учасники інциденту

Ця схема ілюструє зв'язки між різними особами та організаціями, які були залучені до скандалу з Cambridge Analytica та її ролі у зборі та використанні даних користувачів Facebook для політичних цілей. Вона показує, що Олександр Коган з Кембриджського університету розробив застосунок, який збирав дані профілів Facebook, а потім надав їх компанії GSR, яку він також заснував. GSR, у свою чергу, передала ці дані Cambridge Analytica через фірму SCL Elections. Cambridge Analytica використовувала ці дані для надання консультацій у виборчих кампаніях. На схемі також згадується, що GSR мала контракт із російським урядом для дослідження даних Facebook, а один з керівників SCL Elections має зв'язки з російським університетом. Схема висвітлює складну мережу зв'язків та обміну даними між цими організаціями та персонами.

Незважаючи на те, що тільки 270 тисяч людей погодилися на участь в опитуванні, додаток також зміг зібрати інформацію про друзів цих людей завдяки слабкому контролю доступу в API Facebook на той час. Таким чином, Cambridge Analytica отримала доступ до значно більшого обсягу даних, ніж було заявлено спочатку [25].

Цей інцидент став одним із найбільш резонансних і широко висвітлених у ЗМІ випадків порушення конфіденційності даних користувачів у соціальних мережах. Він привернув значну увагу громадськості, регуляторних органів та політиків до практик обробки персональних даних у цифровій сфері. Для Facebook це мало серйозні наслідки, зокрема, значні штрафи, падіння акцій, втрату довіри користувачів та репутаційні збитки [26, с. 8]. Інцидент також викликав занепокоєння щодо безпеки систем автентифікації та авторизації на основі OAuth та підштовхнув інші компанії переглянути свої підходи до імплементації цього протоколу.

Детальний розгляд інциденту з витоком даних користувачів Facebook, пов'язаного з Cambridge Analytica, є вкрай важливим для розуміння потенційних вразливостей систем автентифікації та авторизації, побудованих на основі OAuth. Цей випадок наочно продемонстрував, як зловживання

функціями OAuth може призвести до серйозних порушень конфіденційності даних користувачів та негативних наслідків для компаній, які використовують цей протокол.

Ретельний аналіз обставин цього інциденту дозволить виявити слабкі місця в системах безпеки та імплементації OAuth, а також визначити можливі шляхи їх усунення та вдосконалення. Розуміння причин виникнення такої вразливості є ключовим для розробки ефективних заходів щодо захисту даних користувачів у майбутньому та запобігання подібним випадкам.

3.2. Аналіз причин виникнення вразливості та слабких місць у системі безпеки, що розглядається

Процес, через який Cambridge Analytica змогла отримати доступ до даних мільйонів користувачів Facebook, включав декілька ключових етапів та зловживань функціями OAuth. Спочатку, у 2013 році, науковий співробітник Кембриджського університету Олексій Коган створив додаток під назвою "thisisyourdigitallife", який начебто був призначений для проведення академічного дослідження особистості користувачів [25].

Цей додаток, після надання дозволів через OAuth, міг збирати не лише дані про користувачів, які його встановили, а й про їхніх друзів у Facebook. Таким чином, незважаючи на те, що лише близько 270 тисяч користувачів погодилися встановити додаток, він зміг зібрати дані про десятки мільйонів їхніх друзів. Це стало можливим через недоліки в налаштуваннях конфіденційності та контролю доступу в API Facebook на той час [27].

Зібрані дані, включаючи інформацію про інтереси, вподобання та демографічні характеристики користувачів, були передані Когану, який згодом продав їх Cambridge Analytica. Ця компанія, у свою чергу, використовувала ці дані для створення детальних психологічних профілів користувачів та таргетованої політичної реклами під час президентських виборів у США 2016 року.

Роль OAuth у цьому інциденті полягала в тому, що цей протокол був використаний для надання дозволів додатку Когана на збирання даних про користувачів та їхніх друзів. Однак, на відміну від належного використання OAuth, де сторонні додатки отримують лише обмежений доступ до певних даних, у даному випадку додаток Когана мав можливість збирати значно більше інформації, ніж було заявлено та очікувалося користувачами [28].

Таким чином, зловживання функціями OAuth та недостатній контроль з боку Facebook над обсягом даних, до яких мали доступ сторонні додатки, стали ключовими факторами, що дозволили Cambridge Analytica отримати несанкціонований доступ до великої кількості персональних даних користувачів соціальної мережі.

Політика конфіденційності та безпеки Facebook на момент інциденту з Cambridge Analytica мала певні недоліки та прогалини, які могли сприяти виникненню цієї вразливості. Зокрема, компанія дозволяла стороннім розробникам отримувати доступ до значно більшого обсягу даних про користувачів, ніж це було необхідно для функціонування їхніх додатків [25]. Процедури перевірки та авторизації сторонніх додатків також виявилися недостатньо ретельними, що дозволило додатку обійти обмеження та зібрати дані про мільйони користувачів без їхньої безпосередньої згоди.

Рівень обізнаності та розуміння користувачами ризиків, пов'язаних з наданням дозволів сторонньому програмному забезпеченню через OAuth був досить низьким. Багато користувачів не усвідомлювали повною мірою, що їхні дані можуть бути зібрані та використані третіми сторонами, коли вони надавали дозволи додатку. Умови конфіденційності та політики безпеки Facebook на той час були недостатньо чіткими та зрозумілими для пересічного користувача [28].

Крім того, слід відзначити кілька технічних, організаційних та людських чинників, які могли сприяти виникненню цієї вразливості в системі безпеки Facebook

1. Технічні чинники:

- недостатній контроль та обмеження доступу сторонніх додатків до даних користувачів та їхніх друзів в API Facebook;

- відсутність належного моніторингу активності сторонніх додатків та виявлення можливих порушень політики безпеки.

2. Організаційні чинники:

- недостатня увага до питань безпеки та конфіденційності даних користувачів у корпоративній культурі Facebook;

- брак ресурсів та персоналу, відповідального за перевірку та аудит сторонніх додатків.

3. Людські чинники:

- можливі помилки або недбалість з боку співробітників Facebook, які були відповідальні за процеси авторизації та перевірки сторонніх додатків;

- відсутність належної підготовки та обізнаності персоналу щодо ризиків, пов'язаних з OAuth та доступом сторонніх додатків до даних користувачів.

Сукупність цих факторів створила сприятливі умови для зловживань та несанкціонованого доступу до персональних даних користувачів Facebook через недоліки в системі безпеки та імплементації OAuth.

3.3. Наслідки інциденту та вплив на користувачів і компанію

Інцидент з витоком персональних даних мільйонів користувачів Facebook, мав серйозні наслідки як для самих користувачів, так і для компанії Facebook.

Розпочнемо з розгляду наслідків для користувачів Facebook. Найбільшою проблемою для користувачів стало порушення конфіденційності їхніх персональних даних. Близько 87 мільйонів профілів користувачів, включаючи інформацію про їхні інтереси, вподобання, демографічні характеристики та публікації у Facebook, були незаконно зібрані та передані Cambridge Analytica без відома та згоди цих користувачів [25]. Таке несанкціоноване розголошення

приватних даних є серйозним порушенням права на приватність та конфіденційність.

Існувала реальна загроза того, що зібрані дані можуть бути використані для цілей, про які користувачі не були поінформовані та на які вони не давали згоди. У даному випадку, компанія використовувала ці дані для створення психологічних профілів користувачів та таргетованої політичної реклами під час виборів у США [28]. Такі дії можна розцінювати як зловживання персональними даними та маніпулювання свідомістю користувачів.

Як наслідок, інцидент призвів до значної втрати довіри користувачів до Facebook та занепокоєння щодо безпеки їхніх даних у цій соціальній мережі. Багато людей висловлювали розчарування та обурення через неналежний захист їхньої приватності та конфіденційності .

Наслідки для Facebook як компанії. Для самої компанії Facebook інцидент мав руйнівні наслідки, насамперед у плані репутації та іміджу. Розголошення факту витоку персональних даних мільйонів користувачів завдало серйозної шкоди бренду та підірвало довіру користувачів та партнерів до компанії [29].

Facebook зазнав значних фінансових втрат. Компанія була змушена сплатити багатомільйонні штрафи регуляторним органам у різних країнах, а також витрати на юридичний захист від численних позовів з боку користувачів та організацій [30]. Додатково, вартість акцій Facebook різко впала після оприлюднення інформації про інцидент, що призвело до втрати прибутків для акціонерів.

Однією з найважливіших вимог для Facebook стала необхідність повного перегляду та вдосконалення своїх політик конфіденційності та безпеки, а також процесів авторизації та контролю доступу для сторонніх додатків, що використовують OAuth. Компанія була змушена інвестувати значні ресурси в зміцнення систем захисту даних користувачів.

Розглянемо вплив на галузь соціальних мереж та інтернет-компаній. Інцидент значно посилив увагу до питань конфіденційності даних та безпеки в цифровому просторі. Багато компаній, які працюють з персональними даними

користувачів, були змушені переглянути свої практики та підходи до захисту цієї інформації. Відбулося усвідомлення того, що конфіденційність даних є критично важливою для збереження довіри користувачів та уникнення репутаційних ризиків.

Інцидент з Cambridge Analytica став одним із ключових чинників, що прискорили прийняття нових нормативних актів та регулювань у сфері захисту персональних даних. Зокрема, Загальний регламент захисту даних (GDPR), який набув чинності в Європейському Союзі у 2018 році, встановив жорсткі вимоги до обробки та захисту персональних даних громадян ЄС [31]. Це змусило багато компаній переглянути свої політики конфіденційності та безпеки, а також запровадити нові механізми контролю та захисту даних користувачів.

Інтернет-компанії та соціальні мережі також були змушені переглянути свої практики використання та захисту персональних даних користувачів. Після інциденту з Cambridge Analytica компанії, такі як Facebook, Twitter та Google, зіткнулися з посиленим регуляторним тиском та вимогами до більшої прозорості щодо збору та використання даних своїх користувачів.

Суспільний резонанс та реакція громадськості. Інцидент привернув значну увагу ЗМІ та широкого загалу. Він став предметом активного обговорення питань конфіденційності даних, етичних аспектів використання персональної інформації та захисту приватності в цифровому середовищі [25].

Громадськість також висловлювала занепокоєння щодо можливого впливу витоку даних на політичні процеси та вибори. Існували побоювання, що використання персональних даних для таргетованої політичної реклами може призвести до маніпулювання свідомістю виборців та викривлення результатів виборів.

Важливим наслідком інциденту стало підвищення обізнаності користувачів щодо ризиків, пов'язаних з розголошенням їхніх персональних даних, та необхідності захисту приватності в інтернеті. Багато людей почали

критичніше ставитися до надання своїх даних компаніям та уважніше вивчати політики конфіденційності та умови використання різних онлайн-сервісів.

Після того, як інцидент набув широкого розголосу, Facebook був змушений вжити низку заходів для вирішення проблеми та відновлення довіри користувачів і партнерів.

Одним із ключових кроків стало посилення процесів перевірки та авторизації сторонніх додатків, які отримують доступ до даних користувачів через OAuth. Facebook запровадив більш ретельну та багатоетапну процедуру перевірки додатків, включаючи аналіз їхнього коду, цілей використання даних і заявлених можливостей Це мало на меті запобігти повторенню ситуацій, коли сторонні додатки зловживали своїми дозволами та збирали більше даних, ніж було необхідно для їх функціонування.

Компанія впровадила нові інструменти контролю конфіденційності для своїх користувачів. Зокрема, були додані більш чіткі та зрозумілі налаштування приватності, що дозволяли користувачам обмежувати доступ до їхніх даних для сторонніх додатків та контролювати, яка інформація буде доступна. Ці заходи були спрямовані на підвищення прозорості та надання користувачам більшого контролю над їхніми персональними даними.

Facebook також зробив кроки для вдосконалення своїх політик конфіденційності та безпеки, зокрема, зробивши їх більш зрозумілими для користувачів. Були внесені зміни, спрямовані на забезпечення більш прозорого та етичного використання персональних даних, а також посилення захисту цих даних [32].

Компанія тісно співпрацювала з регуляторними органами, такими як Федеральна торгова комісія США та органами нагляду за захистом даних в Європейському Союзі, для проведення ретельного розслідування інциденту та вжиття відповідних заходів [33]. Ця співпраця була важливою для забезпечення дотримання нових нормативних вимог та уникнення подібних порушень у майбутньому.

Щодо того, як можна було уникнути інциденту з Cambridge Analytica, слід зазначити, що ключовим фактором стало недостатнє регулювання та контроль з боку Facebook над обсягом даних, до яких мали доступ сторонні додатки через OAuth. Якби компанія запровадила жорсткіші обмеження та ретельніший моніторинг діяльності таких додатків, ситуація, коли додаток Когана зміг зібрати дані про мільйони користувачів без їхньої згоди, могла бути попереджена.

Крім того, більш прозорі та зрозумілі політики конфіденційності, а також підвищення обізнаності користувачів щодо ризиків надання дозволів сторонньому програмному забезпеченню через OAuth могли б допомогти уникнути цього інциденту або принаймні зменшити його масштаби.

Загалом, цей випадок підкреслює важливість ретельного контролю, регулювання та моніторингу систем, побудованих на основі OAuth, а також забезпечення високого рівня прозорості та обізнаності як з боку компаній, так і з боку користувачів.

3.4. Рекомендації щодо усунення існуючих вразливостей та вдосконалення безпеки систем на основі OAuth

Після розглянутого інциденту стало очевидно, що існуючі процеси перевірки та авторизації сторонніх додатків, які використовували OAuth для доступу до даних користувачів Facebook, мали серйозні прогалини. Тому першочерговою рекомендацією для усунення таких вразливостей є посилення цих процесів.

Таблиця 3.1 – Узагальнення ключових рекомендації

Рекомендація	Опис
Ретельний аналіз коду застосунків	Проводити ретельний аналіз коду сторонніх додатків, що використовують OAuth, для виявлення потенційних загроз безпеці або можливостей зловживання доступом до даних користувачів.
Чіткі політики конфіденційності	Впровадити зрозумілі політики конфіденційності, які пояснюють, як персональні дані користувачів можуть бути використані сторонніми додатками з доступом через OAuth.

Продовження таблиці 3.1 - Узагальнення ключових рекомендації

| Більший контроль для користувачів	Надати користувачам інструменти для обмеження доступу сторонніх додатків до певних категорій їхніх даних або повністю відмовляти в доступі.
Стандартизація практик	Забезпечити, щоб посилені заходи безпеки та прозорості стали загальноприйнятою практикою для всіх систем, що використовують OAuth.
Моніторинг та аудит	Впровадити системи моніторингу для відстеження активності додатків, обсягів запитуваних даних та способів їх використання, а також регулярно проводити аудити.
Обмеження доступу	Встановити чіткі обмеження на обсяг даних, до яких можуть отримувати доступ сторонні додатки, ґрунтуючись на принципі "мінімальних привілеїв".
Дотримання нормативних вимог	Тісно співпрацювати з регуляторними органами та дотримуватися відповідних нормативних вимог щодо захисту персональних даних і конфіденційності користувачів.
Навчання персоналу	Регулярно проводити навчання та підвищувати обізнаність персоналу щодо ризиків, пов'язаних з OAuth та доступом сторонніх додатків до даних користувачів.

Таблиця містить комплекс рекомендацій, спрямованих на підвищення рівня безпеки та конфіденційності при використанні OAuth та доступі сторонніх додатків до персональних даних користувачів. Ці рекомендації охоплюють різні аспекти, починаючи від ретельного аналізу коду додатків та впровадження чітких політик конфіденційності, до надання користувачам більшого контролю над їхніми даними та встановлення обмежень на обсяг доступу сторонніх додатків.

Крім того, таблиця підкреслює важливість постійного моніторингу та аудиту діяльності додатків, дотримання нормативних вимог у сфері захисту даних, а також регулярного навчання персоналу щодо ризиків, пов'язаних з OAuth.

Впровадження цих рекомендацій є критично важливим для відновлення довіри користувачів до систем, що використовують OAuth, та для забезпечення належного рівня безпеки та конфіденційності персональних даних. Лише шляхом комплексного підходу, що охоплює технічні, організаційні та нормативні заходи, можна ефективно протидіяти загрозам, подібним до тих, що виникли під час інциденту з Cambridge Analytica.

ВИСНОВКИ

1. Протокол OAuth став широко використовуваним стандартом для делегування доступу до персональних даних користувачів сторонніми додатками та веб-сервісами. Однак, як показав інцидент з витоком персональних даних мільйонів користувачів Facebook через зловживання функціями OAuth, імплементації цього протоколу можуть мати серйозні вразливості та створювати ризики для конфіденційності та безпеки даних користувачів.

Детальний аналіз цього інциденту виявив декілька ключових недоліків у системі безпеки Facebook, таких як: недостатній контроль доступу сторонніх додатків, відсутність належного моніторингу їх діяльності, нечіткі політики конфіденційності та безпеки, а також організаційні та людські чинники. Наслідки цього інциденту були вкрай серйозними, включаючи порушення конфіденційності даних мільйонів користувачів, репутаційні та фінансові втрати а також значний суспільний резонанс і підвищення уваги до питань захисту персональних даних.

2. Серед ключових рекомендацій слід виділити: посилення процесів перевірки та авторизації сторонніх додатків, впровадження прозорих політик конфіденційності, надання користувачам більшого контролю над їхніми даними, посилення моніторингу та аудиту діяльності додатків, встановлення чітких обмежень на доступ до даних, співпрацю з регуляторними органами та підвищення обізнаності персоналу щодо ризиків, пов'язаних з OAuth.

Лише шляхом впровадження таких заходів та створення надійних і безпечних систем автентифікації та авторизації на основі OAuth можна забезпечити належний захист персональних даних користувачів, відновити довіру до цих систем та уникнути повторення подібних інцидентів у майбутньому.

3. Результати цього дослідження є актуальними та корисними для розробників програмного забезпечення, системних адміністраторів, фахівців з

інформаційної безпеки та всіх зацікавлених сторін, які працюють з системами автентифікації та авторизації на основі OAuth або планують їх впровадження.

Вони допоможуть зрозуміти потенційні ризики та вразливості, а також визначити належні підходи до забезпечення безпеки та конфіденційності даних користувачів у цих системах.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Hardt D. The OAuth 2.0 Authorization Framework : No. RFC 6749. 2012. URL: https://tools.ietf.org/html/rfc6749 (дата звернення: 05.2024).

2. OAuth 2.0 : IETF Tools [Електронний ресурс]. – Режим доступу: https://tools.ietf.org/html/rfc6749 (дата звернення: 01.05.2024).

3. Parecki A. Understanding OAuth 2.0 [Електронний ресурс] / Aaron Parecki // Aaronparecki.com. – 2023. – Режим доступу: https://aaronparecki.com/oauth-2-simplified/ (дата звернення: 01.06.2024).

4. An Introduction to OAuth 2.0 and OpenID Connect [Електронний ресурс] // Okta Developer Blog. – 2019. – Режим доступу: https://developer.okta.com/blog/2019/10/21/illustrated-guide-to-oauth-and-oidc (дата звернення: 01.06.2024).

5. Google Identity Platform [Електронний ресурс] // Google Developers. – Режим доступу: https://developers.google.com/identity/protocols/oauth2 (дата звернення: 05.2024).

6. Authentication Scenarios for Azure App Service [Електронний ресурс] // Microsoft Docs. – Режим доступу: https://docs.microsoft.com/en-us/azure/app-service/overview-authentication-authorization (дата звернення: 01.06.2024).

7. OAuth 2.0 for Browser-Based Apps [Електронний ресурс] // Auth0 Docs. – Режим доступу: https://auth0.com/docs/flows/concepts/auth-code (дата звернення: 01.06.2024).

8. OAuthLib [Електронний ресурс] // GitHub. – 2023. – Режим доступу: https://github.com/oauthlib/oauthlib (дата звернення: 01.06.2024).

9. Bilbie A. Using OAuth 2.0 for Server to Server Applications [Електронний ресурс] / Alex Bilbie // ScottBrady91.com. – 2022. – Режим доступу: https://www.scottbrady91.com/OAuth/Using-OAuth-2.0-for-Server-to-Server-Applications (дата звернення: 01.06.2024).

10. OAuth 2.0 Threat Model and Security Considerations : IETF Tools [Електронний ресурс]. – Режим доступу: https://tools.ietf.org/html/rfc6819 (дата звернення: 05.2024).

11. OpenID Connect [Електронний ресурс] // OpenID. – 2023. – Режим доступу: http://openid.net/specs/openid-connect-core-1_0.html (дата звернення: 05.2024).

12. OAuth 2.0 : IETF Tools [Електронний ресурс]. – Режим доступу: https://tools.ietf.org/html/rfc6749 (дата звернення: 05.2024).

13. Parecki A. Understanding OAuth 2.0 [Електронний ресурс] / Aaron Parecki // Aaronparecki.com. – 2023. – Режим доступу: https://aaronparecki.com/oauth-2-simplified/ (дата звернення: 05.2024).

14. An Introduction to OAuth 2.0 and OpenID Connect [Електронний ресурс] // Okta Developer Blog. – 2019. – Режим доступу: https://developer.okta.com/blog/2019/10/21/illustrated-guide-to-oauth-and-oidc (дата звернення: 05.2024).

15. Google Identity Platform [Електронний ресурс] // Google Developers. – Режим доступу: https://developers.google.com/identity/protocols/oauth2 (дата звернення: 05.2024).

16. Authentication Scenarios for Azure App Service [Електронний ресурс] // Microsoft Docs. – Режим доступу: https://docs.microsoft.com/en-us/azure/app-service/overview-authentication-authorization (дата звернення: 05.2024).

17. Рябко О. В., Бондаренко М. Ф. Отримання токенів доступу та оновлення в процесах авторизації за протоколом OAuth 2.0. Системи обробки інформації. 2021. № 1. С. 65–72.

18. OAuth 2.0 for Browser-Based Apps [Електронний ресурс] // Auth0 Docs. – Режим доступу: https://auth0.com/docs/flows/concepts/auth-code (дата звернення: 05.2024).

19. OAuthLib [Електронний ресурс] // GitHub. – 2023. – Режим доступу: https://github.com/oauthlib/oauthlib (дата звернення: 05.2024).

20. Bilbie A. Using OAuth 2.0 for Server to Server Applications [Електронний ресурс] / Alex Bilbie // ScottBrady91.com. – 2022. – Режим доступу: https://www.scottbrady91.com/OAuth/Using-OAuth-2.0-for-Server-to-Server-Applications (дата звернення: 05.2024).

21. Google Sign-In for Developers. URL: https://developers.google.com/identity/sign-in/web/ (дата звернення: 15.05.2024).

22. Facebook for Developers - Authentication. URL: https://developers.facebook.com/docs/authentication (дата звернення: 15.05.2024).

23. Microsoft identity platform authentication libraries. URL: https://docs.microsoft.com/en-us/azure/active-directory/develop/reference-v2-libraries (дата звернення: .05.2024).

24. Amazon Cognito Security. URL: https://docs.aws.amazon.com/cognito/latest/developerguide/cognito-user-pool-security.html (дата звернення: .05.2024).

25. Granville K. Facebook and Cambridge Analytica: What You Need to Know as Fallout Widens. The New York Times. 2018. URL: https://www.nytimes.com/2018/03/19/technology/facebook-cambridge-analytica-explained.html (дата звернення: .05.2024).

26. Бакалінський О. В., Гавриш О. М. Соціально-економічні аспекти скандалів зі збором персональних даних користувачів. Східна Європа: економіка, бізнес та управління. 2019. № 3(20). С. 4–10.

27. Tiku N. Europe's Tough New Data Protection Laws Have Claws. Wired. 2018. URL: https://www.wired.com/story/europes-tough-new-data-protection-laws-have-claws/ (дата звернення: .05.2024).

28. Cadwalladr C., Graham-Harrison E. Revealed: 50 million Facebook profiles harvested for Cambridge Analytica in major data breach. The Guardian. 2018. URL: https://www.theguardian.com/news/2018/mar/17/cambridge-analytica-facebook-influence-us-election (дата звернення: 11.05.2024).

29. Solon O. Facebook says data on 87 million users was improperly shared with Cambridge Analytica. The Guardian. 2018. URL: https://www.theguardian.com/technology/2018/apr/04/facebook-cambridge-analytica-user-data-latest-more-than-thought (дата звернення: 15.05.2024).

30. Wakabayashi D., Goldstein D. Meta Fined $414 Million by European Regulators Over Data Privacy. The New York Times. 2022. URL:

https://www.nytimes.com/2023/01/04/technology/meta-facebook-privacy-fine-eu.html (дата звернення: 01.05.2024).

31. GDPR Portal: Site Overview. URL: https://gdpr.eu/ (дата звернення: .05.2024).

32. Facebook's Revised Data Policy. 2018. URL: https://www.facebook.com/about/privacy/update (дата звернення: 12.05.2024).

33. FTC Imposes $5 Billion Penalty and Sweeping New Privacy Restrictions on Facebook. 2019. URL: https://www.ftc.gov/news-events/news/press-releases/2019/07/ftc-imposes-5-billion-penalty-sweeping-new-privacy-restrictions-facebook (дата звернення: .05.2024).