Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Магістерська програма Інженерія програмного забезпечення

Механізми захисту протоколу авторизації OAuth 2.0

Текстова частина до курсової роботи
за спеціальністю
Інженерія програмного забезпечення

Керівник курсової роботи
к.т.н.,доцент
Ткаченко_О.М._____
(прізвище та ініціали)
_____
(підпис)
“____” _____ 2020 р.
Виконав студент
Ханін_М.Ю._____
(прізвище та ініціали)
“____” _____ 2020 р.

Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра інформатики факультету інформатики

ЗАТВЕРДЖУЮ
Зав.кафедри інформатики,
_____С. С. Гороховський
(підпис)
„____”_____2020 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ
на курсову роботу

студенту Ханіну_М.Ю._ факультету _інформатики__ _____5_____курсу
ТЕМА _ Механізми захисту протоколу авторизації OAuth 2.0_____
Вихідні дані:

- Додаток, який реалізує протокол з внесеними рекомендаціями
- Текстова частина курсової роботи

Зміст ТЧ до курсової роботи:

Індивідуальне завдання
Вступ
1 Порвняльна характеристика існуючих протоколів авторизації
2 Формування рекомендацій щодо протоколу OAuth 2.0
3 Розробка сервісу на основі протоколу OAuth 2.0 з урахуванням винесених рекомендацій
Висновки
Список літератури
Додатки

Дата видачі „___” _____ 2020 р. Керівник _____
(підпис)
Завдання отримав _____
(підпис)

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

CSS – Cascading Style Sheets;
HTML – HyperText Markup Language;
HTTP – HyperText Transfer Protocol;
HTTPS – HyperText Transfer Protocol Secure;
REST – Representational State Transfer;
SOAP – Simple Object Access Protocol;
TLS – Transport Layer Security;
URL – Uniform Resource Locator;
XML – eXtensible Markup Language;
XSS – Cross Site Scripting;
ІБ – інформаційна безпека;
ІС – інформаційна система;
СУІБ – система управління інформаційною безпекою;

ВСТУП

Об’єкт дослідження – процес роботи протоколу авторизації OAuth 2.0.

Мета роботи – удосконалення механізмів захисту протоколу OAuth 2.0 за рахунок підписання даних ЕЦП та шифрування.

Предмет дослідження – механізми захисту протоколу авторизації OAuth2.0.

Метод дослідження – аналіз відповідності рівня захищеності критеріям стандартів.

Практична цінність отриманих результатів полягає у розробці сервісу на основі протоколу авторизації OAuth 2.0.

Практика використання протоколів авторизації у веб-додатках значно поширилася за останні десять років. Більшість сервісів вимагають ідентифікацію користувачів у своїх системах для розділення прав доступу мід приватними ресурсами, а також для отримання даних про користувача. Збільшення використання функції авторизації в різних сервісах, порталах спричинило пожвавлення інтересу до таких протоколів авторизації як OAuth 1.0, OAuth 2.0, OpenId, які дозволяють икористовувати, так звану, віддалену авторизацію. При цьому акредитовані сервіси, які мають реалізацію цих протоколів надають доступ до своїх ресурсів та надають ідентифікацію своїх користувачів для інших систем. Наприклад, для реєстрації на форумі вже не виникає необхідності проходити процедуру авторизації саме на цьому форумі й надавати сервісу для зберігання в базі даних свої логін та пароль. Використовуючи один з наведених вище протоколів можна вказати персональні дані, наприклад, від сервісу Google або Facebook й користуватися форумом.

Незалежно від того, розглядається веб-додаток чи веб-сайт, можливі різні підходи для реалізації процедури авторизації. Деякі протоколи направлені на застосуввання у веб-порталах, інші – у мобільних застосунках. Основною метою дослдження способів авторизації є проведення оцінки рівня безпеки інформаційної системи організації для управління нею в цілому з урахуванням перспектив її розвитку.

Для аналізу алгоритмів авторизації веб-ресурсів необхідно керуватися певними стандартами та методологіями. За кордоном найвідомішими з них RFC 6000, а саме RFC 6749, RFC 6750, RFC 6819. До того ж, останнім часом спостерігається поступова зміна напрямку державних нормативних документів України у сфері інформаційної безпеки у напрямку процесного підходу і цінностей RFC 6000.

Застосування перерахованих стандартів має практичний сенс під час аналізу підходів авторизації й розробки нових рішень, адже для організацій в пріоритеті є репутаційна складова у веб-просторі. Репутація закладу забезпечується надійністю та мінімальними вимогами до рівня захищеності веб-ресурсу, в залежності від складності його побудови. А оскільки певні міжнародні стандарти стрімко задають вектори управління інформаційною безпекою навіть на державному рівні, оцінювання веб-ресурсу за їх критеріями буде найбільш об’єктивним способом аналізу. Підґрунтям до детального вивчення аналізу інформаційної безпеки об’єкта інформаційної діяльності (ОІД) за міжнародними стандартами у даній роботі були напрацювання та дослідження Ріхтера Д., [1], Лакшмірагавана Б. [2], Шилдта Г.[3].

Відповідно, метою даної роботи є удосконалення механізмів захисту протоколу OAuth 2.0 за рахунок підписання даних електронно-цифровим підписом та застосування асиметричного шифрування даних користувачів. Розробка сервісу з використанням удосконаленого алгоритму авторизації було обрано в якості практичної частини роботи та буде здійснено за міжнародним стандартами з тестування веб-ресурсів.

Для виконання роботи були поставлені такі задачі:

– з'ясування специфічних для сфери зберігання даних на веб-ресурсах вимог;

– дослідження існуючих рішень і підходів до зберігання корпоративних даних;

– аналіз підходів до проектування веб-додатків з використанням процедури авторизації;

– постановка прикладної задачі, яка б надала можливість для детального дослідження;

– безпосереднє використання засобів дослідження для поставленої задачі;

– аналіз результатів, оцінка використаних та досліджених підходів, визначення переваг і недоліків створеного рішення;

– розробка рекомендацій для удосконалення алгоритсу роботи існуючих рішень;

– застосування винесених рекомендацій щодо процесу роботи алгоритсів для використання в практичній частині роботи;

– аналіз отриманих результатів й порівняння з попередніми даними.

Об’єктом дослідження є процес роботи протоколу авторизації користувачів OAuth 2.0.

Предметом дослідження є механізми захисту протоколу авторизації OAuth 2.0.

У процесі виконання та збору необхідної інформації для курсової роботи було використано наступні методи:

- порівняння;

- аналіз;

- системний підхід;

- метод індукції.

Отже, в сучасних реаліях необхідно постійно перевіряти та удосконалювати існуючі рішення авторизації користувачів для реальних загроз, адже іноді перевірка та вдосконалення продукту може зіграти ключову роль у створенні бізнесу або діяльності організації [16].

РОЗДІЛ 1
ПРОТОКОЛИ АВТОРИЗАЦІЇ, ЇХ ПОРІВНЯЛЬНА ХАРАКТЕРИСТИКА.

1.1 Класифікація протоколів авторизації.

1.1.1 OpenId

OpenID – це відкритий стандарт децентралізованої системи аутентифікації, який надає користувачу можливість створити єдиний обліковий запис для авторизації на великій кількості веб-ресурсів. На сайтах з підтримкою OpenID, користувачам не доводиться пам'ятати дані для аутентифікації. Натомість, їм достатньо бути зареєстрованими на сайті «провайдера ідентифікації» OpenID (який надає ідентифікатор). Оскільки технологія OpenID децентралізована, то будь-який сайт може використовувати програмне забезпечення OpenID як засіб аутентифікації користувачів; OpenID розв'язує проблему не покладаючись на централізований веб-ресурс для підтвердження істинності користувача [16].

Система OpenID - децентралізована система. Це означає, що не потрібно ні центральної служби, ні організації, яка б дозволяла використовувати систему або зареєстровувала б запрошену аутентифікацію OpenID інтернет-ресурсів або провайдерів OpenID. Кінцевий користувач може вільно вибирати провайдера OpenID і зберігати ідентифікатор у разі зміни провайдера [12, 18, 16].

Стандарт не вимагає JavaScript або сучасних браузерів, однак схема аутентифікації добре сумісна з підходом AJAX. Це означає, що кінцевий користувач може пройти аутентифікацію на сайті, не залишаючи поточну сторінку. У цьому випадку комунікація інтернет-ресурсу з OpenID-провайдером буде проходити в фоновому режимі. OpenID-аутентифікація використовує тільки стандартні HTTP (S) запити і відповіді, тому стандарт не вимагає від користувача встановлення додаткового програмного забезпечення. Для роботи OpenID не потрібно використовувати cookie або

які-небудь інші механізми управління сесією. Різноманітні розширення можуть використовуватися OpenID, але не є обов'язковими для використання стандарту.

Алгоритм роботи OpenID:

1. Кінцевий користувач ініціює процес аутентифікації на інтернет-сервісі. Для цього він вводить попередній ідентифікатор на формі.

2. З попереднього ідентифікатора інтернет-сервісу визначається URL кінцевої точки OpenID-провайдера, що використовує кінцевий користувач. Ідентифікатор може містити тільки ідентифікатор провайдера. У цьому випадку кінцевий користувач вказує свій заявлений ідентифікатор, взаємодіючи з провайдером.

3. Опціонально, інтернет-сервіс і OpenID-провайдер створюють загальний секретний ключ для коду аутентифікації повідомлень по протоколу Діффі-Хеллмана. За допомогою коду аутентифікації повідомлень інтернет-сервіс авторизує повідомлення від провайдера без додаткових запитів до нього для перевірки дійсності.

4. У режимі checkid_setup інтернет-сервіс переводить користувача на сайт провайдера для подальшої аутентифікації. В режимі checkid_immediate комунікація браузера з провайдером відбувається непомітно для користувача.

5. Провайдер перевіряє, чи авторизований користувач на сервері і чи хоче він пройти авторизацію на інтернет-сервісі. Специфікація OpenID не описує процес аутентифікації користувача на стороні провайдера.

6. Провайдер перенаправляє браузер користувача назад в інтернет-сервіс, передаючи сервісу результат аутентифікації.

7. Інтернет-сервіс перевіряє підрахунки інформації, отриманої від провайдера, враховуючи повернений URL, інформацію про користувача, одноразову мітку (nonce) і підпис повідомлення. Якщо на кроці 3 був створений спільний секретний ключ, то перевірка відбувається за допомогою нього. Якщо ключ створений не був, то інтернет-сервіс відправляє провайдеру додатковий запит (check_authentication) для перевірки дійсності. У першому випадку інтернет-сервіс називається залежною стороною без пам'яті (stateless), у другому випадку сервіс називається німим (dumb) [7].

1.1.2 BrowserId

BrowserID - це відкритий стандарт децентралізованої системи авторизації користувачів у веб-ресурсах та мобільних додатках, який дозволяє користувачам входити на веб-сайти за допомогою своєї адреси електронної пошти. BrowserID призначений для використання адрес електронної пошти як унікального ідентифікатора для користувачів. Для цього BrowserID вимагає, щоб імена користувачів були адресами електронної пошти. Є, звичайно, плюси і мінуси цього підходу, і багато випадків, коли даний підхід має недоліки.

BrowserID повністю позбавляє користувачів веб-ресурсу від необхідності локального паролю для входу на сайт, звільняючи користувачів і сайти від робіт по створенню, керуванню та безпечному зберіганню паролів. Від користувача вимагається лише зареєструватися один раз в системі BrowserID, і протокол дозволить зайти на будь-які спільні сайти без витрат часу на реєстрацію.

BrowserID передбачає три основних елемента процесу авторизації:

1. Користувач – особа, яка здійснює вхід на сайт за допомогою протоколу BrowserID.

2. Перевіряюча сторона – сайт, який надає модливість входу за допомогою протоколу авторизації BrowserID.

3. Постачальник ідентифікації – домен, який видає своїм користувачам Persona-сумісні ідентифікаційні сертифікати.

Persona і протокол авторизації користувачів BrowserID використовують в ролі ідентифікатора користувача його адресу електронної пошти, тому, зазвичай, саме сервіси електронної пошти являються постачальниками ідентифікції.

В алгоритмі роботи протоколу BrowserID можна виділити три основних етапи:

1. Надання сертифіката користувачу.

2. Генерація ствердження.

3. Верифікація ствердження.

Для того, щоб адреса електронної пошти могла однозначно ідентифікувати користувача, необхідно встановити верифікований зв'язок між браузером і адресою електронної пошти користувача. Для цього необхідно отримати у постачальника ідентифікації криптографічно підписаний сертифікат.

BrowserID використовує асиметричну схему для підпису, тому сертифікат підписується закритим ключем постачальника ідентифікації. Сертифікат містить в собі наступну інформацію:

1. Адресу електронної пошти користувача.
2. Відкритий ключ
3. Мітку часу
4. Доменне ім’я
5. Термін дії сертифіката

Для кожної адреси електронної пошти браузер користувача генерує свою пару відкритого і закритого ключів, і ці пари ніяк не передаються між різними браузерами користувача, тому користувач змушений отримувати новий сертифікат кожен раз, коли закінчується час його дії або коли користувач використовує новий браузер або комп'ютер [25].

У момент, коли користувач ідентифікується за допомогою обраної поштової адреси, браузер у фоновому режимі перевіряє у себе наявність сертифіката для цієї адреси. У тому випадку, якщо у браузера немає відповідного сертифіката, він запитує новий сертифікат у «постачальника ідентифікації».

1.1.3 OAuth 1.0

OAuth — це відкритий стандарт авторизації, який дозволяє користувачам відкривати доступ до своїх приватних даних (фотографії, відео, списки контактів), що зберігаються на одному сайті, іншому сайту, без необхідності вводу імені користувача та паролю.

OAuth дозволяє користувачам роздавати сайтам маркери доступу, до даних що розміщуються на сайтах-сервісах. Кожен маркер доступу надає доступ конкретному сайту (наприклад, сайту редагування відео) до конкретних ресурсів (наприклад, тільки відео від конкретного альбому) та на визначений термін (наприклад, на наступні дві години). Це дозволяє користувачам надавати доступ третім сайтам до їх інформації, що зберігається на інших сайтах — постачальниках послуг, не передаючи повною мірою самих даних та без застосування імені/паролю. OAuth доповнює, проте, за своїм призначенням, відрізняється від OpenID та BrowserID [3].

OAuth з'явився в листопаді 2006 року, під час розробки Блейном Куком протоколу OpenID для сервісу мікроблогів Twitter. Спільно з Крісом Мессіною він шукав спосіб використання OpenID для доступу до Twitter API без надання сервісу пароля. У співпраці з одним з творців OpenID Девідом Рекордоном вони провели аналіз функціональності OpenID, а також власницьких протоколів авторизації, таких як Flickr Auth, Google AuthSub і Yahoo BBAuth, після чого прийшли до висновку, що існує необхідність у новому відкритому протоколі.

У квітні 2007 року утворилася група інженерів, що працювали над його створенням. В її роботі взяли участь співробітники компаній Google і AOL (яка в цей же час представила свій власний протокол OpenAuth). Фінальна версія ядра протоколу OAuth 1.0 була представлена 4 грудня 2007 року. У 2008 році проводилась робота зі стандартизації протоколу в Інженерній раді Інтернету [14].

15 квітня 2009 року Twitter запропонував своїм користувачам рішення, що дозволяє делегувати стороннім сайтам і сервісам доступ до своїх акаунтів. Воно було названо «Увійти через Твіттер» і було засноване на OAuth. Ця подія стала приводом для першого широкого дослідження протоколу на вразливість, і через кілька днів була виявлена потенційна вразливість, яка зачіпає всі існуючі реалізації OAuth. Після цього, 23 квітня спільнотою розробників було випущено перший додаток безпеки до протоколу, який увійшов у оновлену специфікацію OAuth Core 1.0 Revision A, яка

опублікована 24 червня. У квітні 2010 року був випущений інформаційний документ RFC 5849, присвячений стандарту OAuth,

OAuth 2.0 — наступне покоління протоколу OAuth, зворотно не сумісне з OAuth 1.0. OAuth 2.0 фокусується на простоті розробки клієнтської частини, забезпечуючи спеціальні потоки дозволу для веб-застосунків, настільних застосунків, мобільних телефонів. Специфікація розробляється в рамках робочої групи IETF OAuth.

Нове Graph API від Facebook підтримує тільки OAuth 2.0 і є найбільшою реалізацією нового стандарту. З 2011 року Google додав експериментальну підтримку OAuth 2.0 в своє API.

1.1.4 OAuth 2.0

OAuth 2.0 – це відкритий протокол авторизації користувачів у веб-додатках та мобільних застосунках, який дозволяє отримати третій стороні обмежений доступ до захищених ресурсів користувача без необхідності передавати їй (третій стороні) логін та пароль. OAuth 2.0 дозволяє отримати доступ до ресурсів власника ресурсу шляхом включення клієнтських додатків на HTTP-сервісах, таких як Facebook, GitHub і т. д. Він дозволяє спільно використовувати ресурси, що зберігаються в одній системі, на іншому сайті без використання їх облікових даних. Замість цього використовуються маркери імені користувача та пароля [1].

Найбільшою перевагою OAuth 2.0 над іншими протоколами полягає в тому, що він надає можливість отримати доступ до ресурсів користувача без спільного використання паролів. OAuth 2.0 надає потоки користувацьких агентів для запуску клієнтської програми з використанням мови сценаріїв, такої як JavaScript. Як правило, браузер - це призначений для користувача агент. Він отримує доступ до даних за допомогою токенів замість використання їх облікових даних і зберігає дані онлайн в файловій системі користувача, такій як Google Docs або аккаунт Dropbox[35].

1.2 Нормативно-правове забезпечення

1.2.1 RFC 6749. OAuth authorization framework

Стандарт RFC серії 6000 містить основні підходи та рекомендації до використання протоколів авторизації у веб-додатках. Стандарт RFC 6749 описує відкритий протокол авторизації OAuth, який дозволяє отримати третій стороні обмежений доступ до захищених ресурсів користувача без необхідності передавати їй (третій стороні) логін та пароль. Згідно стандарту, OAuth являє собою фреймворк для авторизації, що дозволяє додаткам здійснювати обмежений доступ до призначених для користувача аккаунтів на HTTP сервісах, наприклад, на Facebook, GitHub і DigitalOcean. Він працює за принципом делегування аутентифікації користувача сервісу, на якому знаходиться аккаунт користувача, дозволяючи сторонньому додатку отримувати доступ до аккаунту користувача. OAuth працює в інтернеті, на десктопних і мобільних додатках [5].

OAuth визначає наступні ролі:

• Власник ресурсу

• Клієнт

• Сервер ресурсів

• Авторизаційний сервер [8]

1.2.2 RFC 6750. Bearer tokens

Згідно зі специфікацією стандарту RFC 6750, протокол авторизації OAuth 2.0 в роботі алгоритму використовує наступні типи токенів:

- токени (маркери) доступу;
- токени (маркери) оновлення;

Токени доступу - це те, що додатки використовують для створення запитів API від імені користувача. Маркер доступу являє собою авторизацію конкретної програми для доступу до певних частин даних користувача.

Токени доступу повинні зберігатися конфіденційно в процесі транспортування та зберігання. Єдиними сторонами, які повинні бачити маркер доступу, є сама програма, сервер авторизації та сервер ресурсів. Програма повинна забезпечити, щоб зберігання маркера доступу не було доступним для інших програм на одному пристрої. Токен доступу може використовуватися тільки через з'єднання https, оскільки передача його через незашифрований канал зробить її тривіальною для третіх сторін для перехоплення. Кінцевою точкою токена є те, де програми подають запит на отримання маркера доступу для користувача[16, 36].

Токени оновлення несуть інформацію, необхідну для отримання нового маркера доступу. Іншими словами, всякий раз, коли маркер доступу необхідний для доступу до конкретного ресурсу, клієнт може використовувати маркер оновлення для отримання нового маркера доступу, виданого сервером аутентифікації. Загальні випадки використання включають отримання нових маркерів доступу після закінчення старих або отримання першого доступу до нового ресурсу. Токени оновлення також можуть закінчитися, але досить довговічні. Оновлення маркерів зазвичай підлягають жорстким вимогам до зберігання, щоб переконатися, що вони не просочилися. Вони також можуть бути включені до чорного списку сервером авторизації.

1.2.3 RFC 6819. Threat model and security considerations

У цьому розділі стандартів RFC серії 6000 наведено комплексну модель загрози OAuth 2.0. За цим нормативним документом, загрози групуються за спрямуванням проти певних OAuth компонентів, наприклад, атаки на клієнта, атаки на сервери авторизації та атаки на сервери ресурсів. Також вони згруповані по цільовому потоку, наприклад, атаки направлені на отримання маркер (токен) доступу або атаки направлені

на отримання маркер (токен) оновлення. Також у цьому стандарті наведені загрози, спрямовані на отримання секрет клієнта, ідентифікатор ключа, ідентифікатор сесії, фішинг та інші види загроз [38].

1.3 Порівняльна характеристика протоколів авторизації

Згідно міжнародного стандарту RFC 5748, який описує загальні вимоги до процесу авторизації та, насамперед, протоколів авторизації користувачів у веб-додатках та мобільних застосунках, протоколи авторизації повинні відповідати ряду критеріїв, таких як підтримка мобільних додатків, рівень захищеності, підтримка серед сервісів. Також світова спільнота виокремлює й четвертий критерій оцінки ефективності протоколів: зручність, який передбачає легкість реалізації даного протоколу у сервісах та додатках. Саме тому розглянуті раніше протоколи авторизації, а саме OpenId, BrowserId, OAuth 1.0, PAuth 2.0 було детально проаналізовано, визначено їх переваги й недоліки, а також оцінено за п'ятибальною шкалою, де п’ять – максимальний бал, а один – мінімальний бал. Результати порівняння наведені в таблиці 1.1 [20].

Таблиця 1.1

Порівняння найпоширеніших протоколів авторизації

	OpenId	Persona	OAuth 1.0	OAuth 2.0
Підтримка мобільних додатків	5	4	1	5
Зручність	3	4	3	5
Рівень захищеності	4	4	4	4
Підтримка серед сервісів	2	3	4	5

Згідно проведеного аналізу, який складався з декількох етапів: ознайомлення, інтеграція протоколу авторизації до веб-додатку написаного на мові C# ASP.NET WEB API фреймворку (серверна частина), JavaScript AngularJS (клієнтська частина) та PL-SQL (база даних); тестування додатку, аудит об’єкту інформаційної діяльності на наявні загрози, найкращі показники виявилися у протоколу авторизації користувачів OAuth 2.0. Цей протокол має найбільшу підтримку серед існуючих сервісів, підтримує мобільні додатки, зручно інтегрується й показує доволі високі показники захищеності. Також непогані показники виявилися у протоколів авторизації OpenId та OAuth 1.0. Проте протокол авторизації користувачів OAuth 1.0 зовсім не має підтримки мобільних додатків і постає необхідність все одно використовувати браузер. А протокол авторизації користувачів має низький рівень підтримки серед сервісів, що ускладнює процес його інтеграції до веб-додатку [16].

Висновки за розділом 1

У розділі 1 було проаналізовано існуючі протоколи авторизації й проведено їх порівняльну характеристику. Теоретичні відомості щодо концепції роботи протоколів авторизації користувачів у веб-додатках та інших програмних звстосунках сприяли формуванню основних принципів та методів проведення даної курсової роботи. Так, основними цілями аналізу існуючих рішень в напрямі авторизації користувацьких сесій та надання рекомендацій і розробки нових підходів та алгоритмів є оцінка поточного рівня захищеності інформаційної системи, оцінка відповідності ІС існуючим стандартам в області інформаційної безпеки, аналіз ризиків, які зв`язані з можливістю реалізації загроз безпеки ресурсів та розробка рекомендацій по впровадженню нових і підвищенню ефективності існуючих механізмів безпеки інформаційної системи. Також варто зазначити, що у роботі буде проведена розробка рекомендацій та буде впроваджено ці екомендації до сервісу. Серед трьох основних методів аналізу

інформаційної безпеки алгоритмів найдоцільнішим є комбінований метод, застосування якого дозволяє найбільш повно відобразити рівень захищеності веб-ресурсу.

В якості основоположних стандартів для проведення аналізу роботи протоколів авторизації користувачів у веб-додатках є міжнародні стандарти серії RFC 6000. Наведені стандарти базуються на процесному підході, при цьому стандарт RFC 6789 виступає в якості більш консолідуючого.

Розглянувши нормативно-правову базу роботи протоколів авторизації та проведши порівняльний аналіз існуючих протоколів за рядом критеріїв, описаних у міжнародному стандарті RFC 5748, був отриманий результат, який свідчить, що за всіма критеріями переважає протокол авторизації OAuth 2.0, який і буде виступати об'єктом подальших досліджень як протокол, що має найвищий показник. Для подальшого дослідження основних загроз ІБ веб-ресурсу були досліджені клієнт-серверна архітектура та основні принципи роботи OAuth 2.0 протоколу. Отримана інформація буде використовуватися для проведення тестування існуючої конфігурації протоколу та наведення рекомендацій щодо покращення механізмів захисту даного алгоритму.

РОЗДІЛ 2
МЕХАНІЗМИ ЗАХИСТУ ПРОТОКОЛУ АВТОРИЗАЦІЇ OAUTH 2.0. ФОРМУВАННЯ РЕКОМЕНДАЦІЙ ЩОДО ПОКРАЩЕННЯ РОБОТИ ПРОТОКОЛУ OAUTH 2.0.

2.1 Характеристика роботи протоколу аторизації OAuth 2.0

2.1.1 Мета використання протоколу аторизації OAuth 2.0

Основною метою використання протоколу авторизації користувачів у веб-додатках та інших програмних застосунках є забезпечення надійної та зручної верифікації користувача через іншу систему, таку як Google, Facebook або будь-який інший сервіс, який передбачає дистанційну автентифікацію та авторизацію користувача. OAuth 2.0 – відкритий протокол авторизації, який дозволяє отримати третій стороні обмежений доступ до захищених ресурсів користувача без необхідності передавати їй (третій стороні) логін та пароль. OAuth дозволяє отримати доступ до ресурсів власника ресурсу шляхом включення клієнтських додатків на HTTP-сервісах. Він дозволяє спільно використовувати ресурси, що зберігаються на одному сайті, на іншому сайті без використання їх облікових даних. Замість цього використовуються маркери імені користувача та пароля. OAuth 2.0 може використовуватися для читання даних користувача з іншої програми. Він забезпечує робочий процес авторизації для веб-додатків, настільних додатків і мобільних пристроїв. OAuth 2.0 використовує код авторизації і не взаємодіє з обліковими даними користувача. Усе перераховане вище доводить, чому протокол авторизації користувачів OAuth 2.0 є могутнім механізмом авторизації у веб-додатках [9].

2.1.2 Особливості протоколу OAuth 2.0

OAuth 2.0 — наступне покоління протоколу OAuth, який має багато спільного з протоколом авторизації користувачів OpenID, однак має деякі особливості, притаманні лише даній версії протоколу. OAuth 2.0 фокусується на простоті розробки клієнтської частини, забезпечуючи спеціальні потоки дозволу для веб-застосунків, настільних застосунків, мобільних телефонів. OAuth 2.0, на відміну від протоколу OpenID, дозволяє отримати доступ до ресурсів користувача без спільного використання паролів. Він надає потоки користувацьких агентів для запуску клієнтської програми з використанням мови сценаріїв, такої як JavaScript. Як правило, браузер - це призначений для користувача агент. Протокол OAuth 2.0 надає доступ до даних за допомогою токенів замість використання їх облікових даних і зберігає дані онлайн у файловій системі користувача, такій як Google Docs або аккаунт Dropbox [7].

2.2 Принцип роботи протоколу авторизації OAuth 2.0

2.2.1 Ролі протоколу авторизації OAuth 2.0

OAuth 2 являє собою фреймворк для авторизації, що дозволяє додаткам здійснювати обмежений доступ до призначених для користувача аккаунтів на HTTP сервісах, наприклад, на Facebook, GitHub і DigitalOcean. Він працює за принципом делегування аутентифікації користувача сервісу, на якому знаходиться аккаунт користувача, дозволяючи сторонньому додатку отримувати доступ до аккаунту користувача. OAuth 2 працює в інтернеті, на десктопних і мобільних додатках. OAuth визначає чотири ролі для елементів роботи свого алгоритму:

- Власник ресурсу
- Клієнт
- Сервер ресурсів

• Авторизаційний сервер

Далі наводиться детальна характеристика кожної з ролей та додається список можливих підролей, які залежать від конкретної конфігурації протоколу та веб-додатку вцілому [5].

2.2.1.1 Власник ресурсу: Користувач

Власником ресурсу є користувач, який авторизує додаток для доступу до свого облікового запису. Доступ до програми призначеного для користувача аккаунту обмежений "областю видимості" (scope) наданих прав авторизації (наприклад, доступ на читання або запис) [6].

2.2.1.2 Сервер ресурсів / авторизації: API

Сервер ресурсів безпосередньо зберігає захищені дані акаунтів користувачів, а авторизаційний сервер перевіряє справжність інформації, наданої користувачем, а потім створює авторизовані токени для програми, за допомогою яких додаток буде здійснювати доступ до призначених для користувача даних.

З точки зору розробника додатка API сервісу одночасно виконує і роль сервера ресурсів, і роль сервера авторизації. Далі ми будемо вважати ці дві ролі однією, і називати її Сервіс або API [8].

2.2.1.3 Клієнт: Додаток

Клієнтом є додаток, який хоче здійснити доступ до аккаунту користувача. Перед здійсненням доступу додаток повинен бути авторизований користувачем, а авторизація повинна бути схвалена з боку API. При цьому посилається запит на отримання дозволу на авторизацію від додатка до сервера авторизації. Сервер авторизації надсилає

відповідь. Якщо відповідь схвальна, лише тоді розпочинається алгоритм роботи протоколу. Якщо відповідь містить відмову в наданні авторизації для додатку, можна надіслати повторний запит через деякий проміжок часу [11].

2.2.2 Схема роботи протоколу авторизації OAuth 2.0

Проаналізувавши ролі авторизації у протоколі авторизації OAuth 2.0 доцільно розглянути схему взаємодії елементів один з одним. Абстрактна схема роботи протоколу представлена на рисунку 1.1.

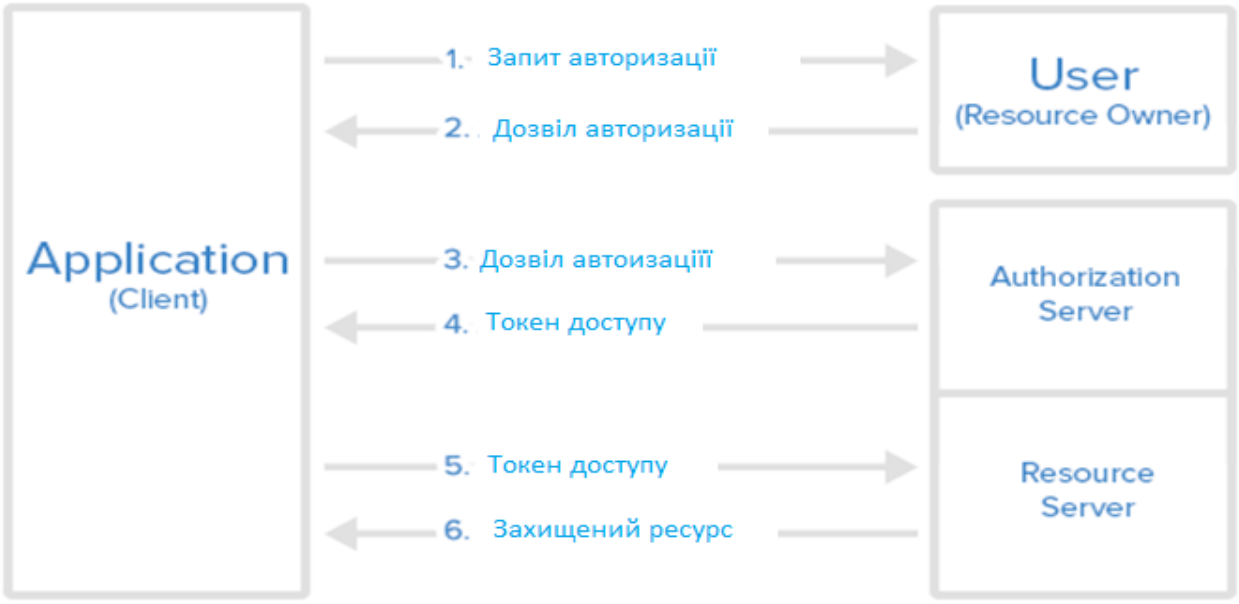

Рисунок 1.1 – Абстрактна схема роботи протоколу OAuth 2.0

Під час проходження алгоритму авторизації у веб-додатках з використанням протоколу авторизації OAuth 2.0 має місце наступна послідовність кроків:

1. Додаток запитує у користувача авторизацію на доступ до сервера ресурсів.

2. Якщо користувач авторизує запит, додаток отримує дозвіл на авторизацію (authorization grant).

3. Додаток запитує авторизаційний токен у сервера авторизації (API) шляхом надання інформації про самого себе і дозвіл на авторизацію від користувача.

4. Якщо справжність додатків підтверджена і дозвіл на авторизацію дійсно, сервер авторизації (API) створює токен доступу для програми. Процес авторизації завершений.

5. Додаток запитує ресурс у сервера ресурсів (API), надаючи при цьому токен доступу для аутентифікації.

6. Якщо токен дійсний, сервер ресурсів (API) надає запитуваний ресурс з додатком.

Фактичний порядок кроків описаного процесу може відрізнятися в залежності від використовуваного типу дозволу на авторизацію, але в цілому процес буде виглядати описаним чином. Далі буде розглянуто різні типи дозволів на авторизацію, притаманних протоколу авторизації OAuth 2.0 в залежності від конфігурації додатку [12].

2.2.2.1 Реєстрація додатку

Наступним кроком, перед використанням протоколу авторизації OAuth необхідно зареєструвати додаток в сервісі. Для цього необхідно в розділі "developer" або "API" сайту сервісу, де є необхідність надати наступну інформацію:

• назву додатка;

• сайт додатка;

• redirect URL або callback URL.

Redirect URL - це URL, на який сервіс буде перенаправляти користувача після авторизації (або відмови в авторизації) прикладної програми. Лише після проходження процедури реєстрації веб-додатку, мобільного додатку або іншого роду рішення сервіс зможе використовувати алгоритми авторизації протоколу OAuth 2.0 [13].

2.2.2.2 Ідентифікатор клієнта і секрет клієнта

Після реєстрації додатка сервіс створює облікові дані клієнта - ідентифікатор клієнта (client ID) і секрет клієнта (client secret). Ідентифікатор клієнта є публічно-доступним записом, який використовує API сервісу для ідентифікації додатка, а також для створення авторизаційних URL для користувачів. Секрет клієнта використовується для підтвердження дійсності додатка для API сервісу, коли додаток запитує доступ до аккаунту користувача. Секрет клієнта повинен бути відомий тільки додатку і API сервісу. Варто зазначити, що саме такий підхід робить протокол OAuth водноас найзахищенішим і найзручнішим серед аналогів, оскільки пара ідентифікатор клієнта і секрет клієнта дає змогу однозначно ідентифікувати користувача й швидко, без значних обчислювань, однозначно підтвердити його особистість[14].

2.2.2.3 Дозвіл на авторизацію в протоколі авторизації OAuth 2.0

В абстрактному описі протоколу авторизації OAuth 2.0 вище перераховані перші чотири кроки, що стосуються питань створення дозволу на авторизацію і токена доступу. Тип дозволу на авторизацію залежить від використовуваного додатком методу запиту авторизації, а також від того, які типи дозволу підтримуються з боку API. OAuth 2 визначає чотири різних типи:

• Код авторизації (Authorization Code): використовується з серверними додатками (server-side applications).

• Неявний (Implicit): використовується мобільними або веб-додатками (додатки, що працюють на пристрої користувача) [18].

• Облікові дані власника ресурсу (Resource Owner Password Credentials): використовуються довіреними додатками, наприклад додатками, які є частиною самого сервісу [16].

• Облікові дані клієнта (Client Credentials): використовуються при доступі додатки до API.

2.2.2.4 Тип дозволу на авторизацію: Код авторизації

Код авторизації є одним з найбільш поширених типів дозволу на авторизацію, оскільки він добре підходить для серверних додатків, де вихідний код програми і секрет клієнта не доступні ззовні. Процес в даному випадку будується на перенаправленні (redirection), що означає, що програма має бути в стані взаємодіяти з призначеним для користувача агентом (user-agent), наприклад, веб-браузером, і отримувати коди авторизації API, перенаправляти через призначений для користувача агент [19].

Дана діаграма, зображена на рисунку 2.1, наглядно ілюструє цей процес:

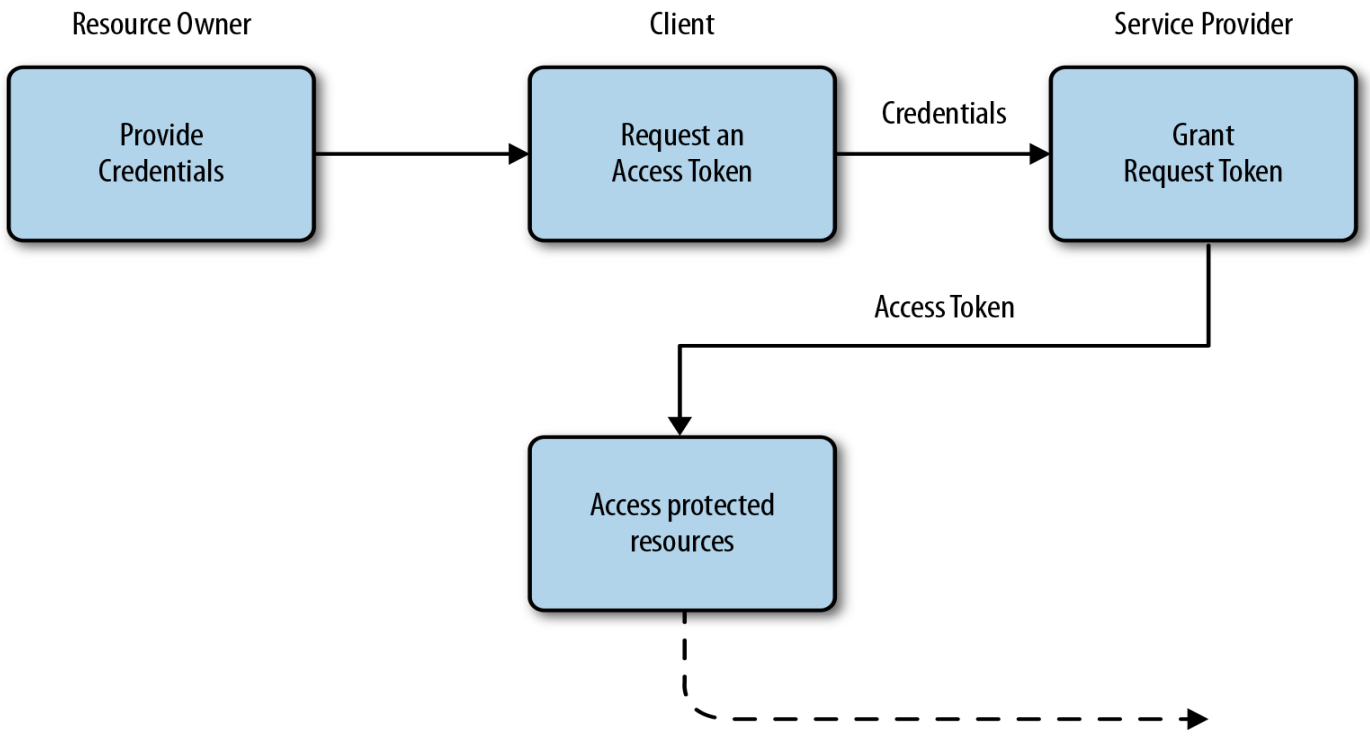

Рисунок 2.1 – Алгоритм роботи протоколу з кодом авторизації

Крок 1: Посилання з кодом авторизації

Спочатку користувачеві надається посилання наступного виду:

https://cloud.digitalocean.com/v1/oauth/authorize?response_type=code&client_id=CLIENT_ID&redirect_uri=CALLBACK_URL&scope=read

Розглянемо компоненти посилання:

• https://cloud.digitalocean.com/v1/oauth/authorize: вхідні точка API авторизації (API authorization endpoint).

• client_id = CLIENT_ID: ідентифікатор клієнта додатка (за допомогою цього ідентифікатора API розуміє, яке додаток запитує доступ).

• redirect_uri = CALLBACK_URL: URL, на який сервіс перенаправить користувача агент (браузер) після видачі авторизационного коду.

• response_type = code: вказує на те, що додаток запитує доступ за допомогою коду авторизації.

• scope = read: задає рівень доступу додатки (в даному випадку - доступ на читання).

Крок 2: Користувач авторизує додаток

Коли користувач натискає на посилання, він повинен спершу здійснити вхід в систему для підтвердження своєї особи. Після цього сервіс запропонує користувачеві авторизувати або відмовити в авторизації додатком для доступу до аккаунту користувача.

Крок 3: Додаток отримує код авторизації

Якщо користувач вибирає "Авторизувати додаток", сервіс перенаправляє користувача агент (браузер) по URL перенаправлення (redirect URL), який був заданий на етапі реєстрації клієнта (разом з кодом авторизації). Посилання буде виглядати схожим чином (в даному прикладі додаток називається "dropletbook.com"):

https://dropletbook.com/callback?code=AUTHORIZATION_CODE

Крок 4: Додаток запитує токен доступу

Додаток запитує токен доступу у API шляхом відправки авторизационного коду і аутентификационной інформації (включаючи секрет клієнта) сервісу. Нижче представлений приклад POST-запиту для отримання токена DigitalOcean:

https://cloud.digitalocean.com/v1/oauth/token?client_id=CLIENT_ID&client_secret=CLIENT_SECRET&grant_type=authorization_code&code=AUTHORIZATION_CODE&redirect_uri=CALLBACK_URL

Крок 5: Додаток отримує токен доступу

Якщо авторизація пройшла успішно, API повертає токен доступу (а також, опціонально, токен для поновлення токена доступу - refresh token). Весь відповідь сервера може виглядати наступним чином:

{"access_token":"ACCESS_TOKEN","token_type":"bearer","expires_in":2592000,"refresh_token":"REFRESH_TOKEN","scope":"read","uid":100101,"info":{"name":"Mark E. Mark","email":"mark@thefunkybunch.com"}}

На завершальному етапі додаток авторизовано. Воно може використовувати токен для доступу до призначеного для користувача аккаунту через API сервісу з заданими обмеженнями доступу до тих пір, поки не закінчиться термін дії токена або токен не буде відкликано. Якщо був створений токен для поновлення токена доступу, він може бути використаний для отримання нових токенів доступу, коли закінчиться термін дії старого токена [21].

2.2.2.5 Тип дозволу на авторизацію: Неявний

Неявний тип дозволу на авторизацію використовується мобільними і веб-додатками (програмами, які працюють в веб-браузері), де конфіденційність секрету клієнта не може бути гарантована. Неявний тип дозволу також заснований на перенаправленні користувача агента, при цьому токен доступу передається призначеному для користувача агенту для подальшої передачі з додатком. Це, в свою чергу, робить токен доступним користувачеві і іншим додаткам на пристрої користувача. Також при цьому типі дозволу на авторизацію не проводиться автентифікація справжності додатку, а сам процес покладається на URL перенаправлення (зареєстрований раніше в сервісі).

Неявний тип дозволу на авторизацію не підтримує токени поновлення токена доступу (refresh tokens).

Процес виглядає наступним чином: додаток просить користувача авторизувати себе, потім сервер авторизації передає токен доступу до призначеного для користувача агенту, який передає токен з додатком. Далі, на рисунку 3.1, наведений опис даного процесу в деталях.

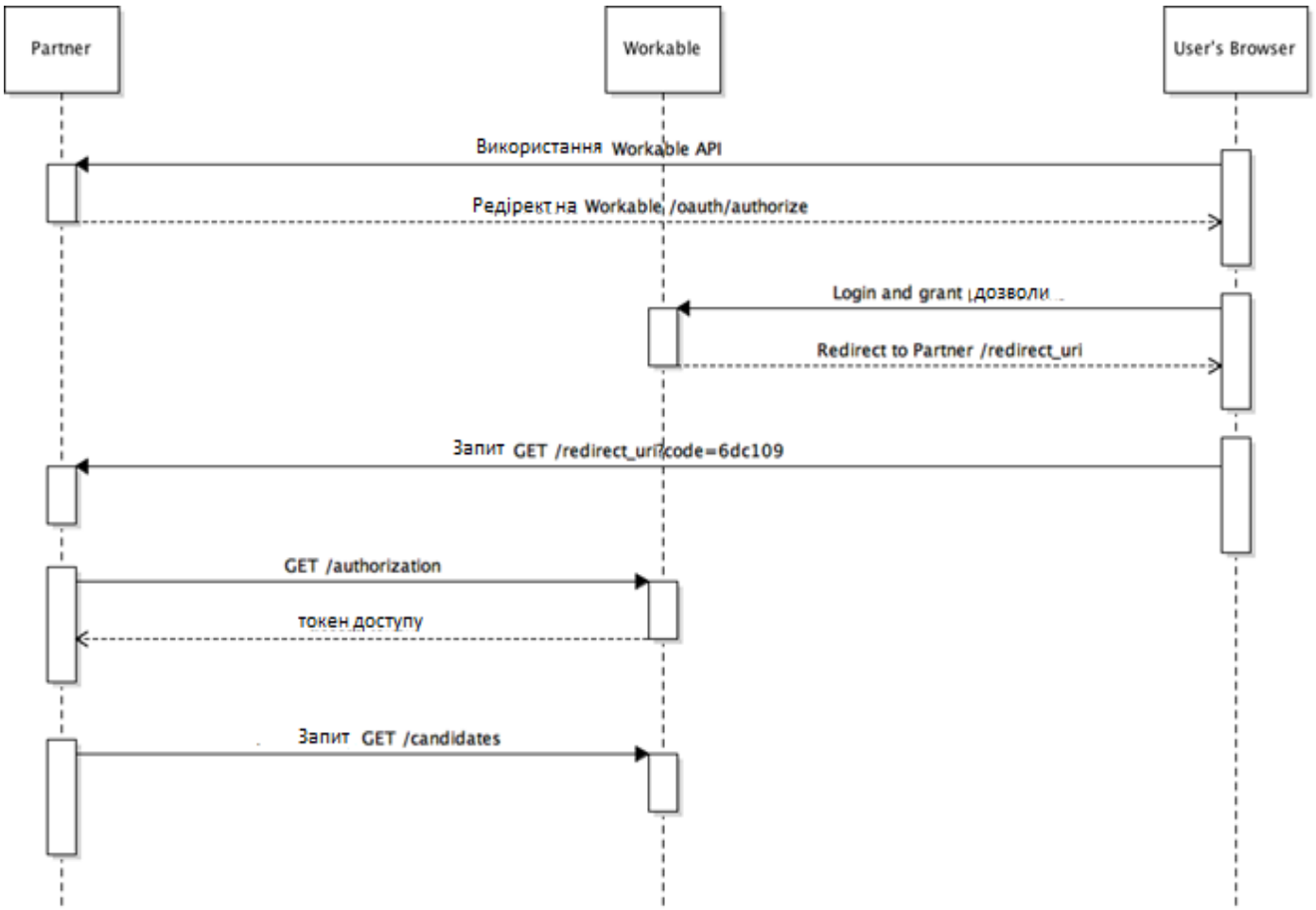

Рисунок 3.1 – Схема неявної авторизації користувача

Крок 1: Посилання для неявної авторизації

При неявному типі дозволу на авторизацію користувачеві надається посилання, яка запитує токен у API. Це посилання виглядає майже так само, як посилання для попереднього способу (з кодом авторизації), за винятком того, що запитується токен замість коду (зверніть увагу на response type "token"):

https://cloud.digitalocean.com/v1/oauth/authorize?response_type=token&client_id=CLIENT_ID&redirect_uri=CALLBACK_URL&scope=read

Крок 2: Користувач авторизує додаток

Коли користувач натискає на посилання, він повинен спершу здійснити вхід в систему для підтвердження своєї особи. Після цього сервіс запропонує користувачеві авторизувати або відмовити в авторизації додатком для доступу до аккаунту користувача.

Крок 3: Призначений для користувача агент отримує токен доступу з URI перенаправлення.

Якщо користувач вибирає "Авторизувати додаток", сервіс перенаправляє користувача агент по URI пренправленія додатки і включає в URI фрагмент, що містить токен доступу. Це виглядає приблизно ось так:

https://dropletbook.com/callback#token=ACCESS_TOKEN

Крок 4: Призначений для користувача агент слід по URI перенаправлення

Призначений для користувача агент слід по URI перенаправлення, зберігаючи при цьому токен доступу.

Крок 5: Додаток виконує скрипт вилучення токена доступу

Додаток повертає веб-сторінку, яка містить скрипт для вилучення токен доступу з повного URI перенаправлення, збереженого призначеним для користувача агентом.

Крок 6: Токен доступу передається з додатком

Призначений для користувача агент запускає скрипт вилучення токена доступу, а потім передає витягнутий токен з додатком.

Тепер додаток авторизовано. Він може використовувати токен для доступу до призначеного для користувача аккаунту через API сервісу з заданими обмеженнями

доступу до тих пір, доки не закінчиться термін дії токена або токен не буде відкликано[16].

2.2.2.6 Тип дозволу на авторизацію: облікові дані власника ресурсу

При цьому типі дозволу на авторизацію користувач надає через додаток безпосередньо свої авторизовані дані в сервісі (ім'я користувача і пароль). Додаток, в свою чергу, використовує отримані облікові дані користувача для отримання токена доступу від сервісу. Цей тип дозволу на авторизацію повинен використовуватися тільки в тому випадку, коли інші варіанти недоступні. Крім того, цей тип дозволу варто використовувати тільки в разі, коли додаток користується довірою користувача (наприклад, є частиною самого сервісу, або операційної системи користувача).

Після того, як користувач передасть свої облікові дані з додатком, додаток запросить токен доступу у авторизаційного сервера. Приклад POST-запиту може виглядати наступним чином:

https://oauth.example.com/token?grant_type=password&username=USERNAME&password=PASSWORD&client_id=CLIENT_ID

Якщо облікові дані коректні - сервер авторизації поверне токен доступу з додатком. Тепер додаток авторизовано.

DigitalOcean в даний час не підтримує тип дозволу на авторизацію з використанням облікових даних власника ресурсу, тому посилання вище перенаправляє на уявний авторизаційний сервер "oauth.example.com".

2.2.2.7 Тип дозволу на авторизацію: Облікові дані клієнта

Тип дозволу на авторизацію з використанням облікових даних клієнта дозволяє додатку здійснювати доступ до свого власного аккаунту сервісу. Це може бути корисно, наприклад, коли додаток хоче оновити власну реєстраційну інформацію на сервісі або

URI перенаправлення, або ж здійснено доступ до іншої інформації, що зберігається в акаунті додатки на сервісі, через API сервісу.

Додаток відправляє запит на отримання токену доступу шляхом відправки своїх облікових даних, ідентифікатора клієнта і секрету клієнта авторизационному серверу. Приклад POST-запиту може виглядати наступним чином:

https://oauth.example.com/token?grant_type=client_credentials&client_id=CLIENT_ID&client_secret=CLIENT_SECRET

Якщо облікові дані клієнта коректні - авторизаційний сервер поверне токен доступу для програми. Тепер додаток авторизовано для використання власного облікового запису!

DigitalOcean в даний час не підтримує тип дозволу на авторизацію з використанням облікових даних клієнта, тому посилання вище веде на уявний авторизаційний сервер "oauth.example.com".

2.2.2.8 Приклад використання токена доступу

Після того, як додаток отримає токен доступу, він може використовувати цей токен для доступу до призначеного для користувача аккаунту через API сервісу з заданими обмеженнями доступу до тих пір, доки не закінчиться термін дії токена або токен не буде відкликано.

Нижче представлений приклад запиту до API з використанням curl. Токен доступу:

curl -X POST -H "Authorization: Bearer ACCESS_TOKEN""https://api.digitalocean.com/v2/$OBJECT"

Якщо токен доступу валідний, API обробить отриманий запит. Якщо термін дії токена доступу закінчився або токен є коректним, API поверне помилку "invalid_request".

2.2.2.9 Оновлення токена доступу

Після закінчення терміну дії токена доступу всі запити до API з його використанням повертатимуть помилку "Invalid Token Error". Якщо при створенні токена доступу був створений і токен для поновлення токена доступу (refresh token), останній може бути використаний для отримання нового токена доступу.

Нижче представлений приклад POST-запиту, що використовує токен для поновлення токена доступу для отримання нового токена доступу:

https://cloud.digitalocean.com/v1/oauth/token?grant_type=refresh_token&client_id=CLIENT_ID&client_secret=CLIENT_SECRET&refresh_token=REFRESH_TOKEN

Нижче, на рисунку 4.1, наведена схема роботи протоколу з токенами доступу.

Рисунок 4.1 – Схема роботи протоколу авторизації OAuth 2.0 з використанням токенів доступу (спрощена схема)

2.3 Переваги та недоліки протоколу авторизації OAuth 2.0

Протокол авторизації користуваяів OAuth 2.0 прийнятий найкращим серед аналогів за рядом критеріїв, однак і він має свої переваги й недоліки, Нижче наведені переваги протоколу.

Легкість. При відправці посилання на малюнок або відео і бажанні залишити коментар єдина проблема полягає в тому, що користувач ніколи не був на цьому сайті раніше, і у нього немає облікового запису. У такому разі протокол OAuth 2.0 надає змогу увійти до свого облікового запису Twitter чи іншого сервісу.

Час. Це не тільки легко, але і неймовірно заощаджує час в довгостроковій перспективі. Будь то пошук нового сайту в Інтернеті або повернення до давно відомого, якщо сайти, які користувач часто відвідує, підтримують OAuth, йому потрібно всього лише створити один обліковий запис для більшості місць в онлайн-чаті.

Networking. OAuth 2.0 необхідний для публікування і коментування фотографій на сайтах, щоб бути соціальними, адже саме мережева привабливість робить сайти гідними відвідування. OAuth дозволяє використовувати один обліковий запис для коментування на кількох різних сайтах, дозволяючи друзям і читачам з усіх сайтів відстежити користувача до сторінки кращого профілю. Більш того, веб-сайти можуть також дозволяти користувачу передавати списки друзів, тому йому не потрібно додавати до списку ті ж 200 друзів кожен раз, коли в мережі з'являється новий популярний мережевий сайт. Точно так же, якщо у користувача є люди, які стежать за ним тільки на одному з цих сайтів, він може включити опції, що дозволяють залишати коментарі на одному сайті і на інших сайтах. Таким чином, йому не потрібно копіювати і вставляти кожен твіт в Google Buzz, і замість цього він може дозволити соціальним мережам працювати в повну силу.

Конфіденційність. При необхідності здійснити придбання, наприклад, в інтернет-магазині, постає потреба вказати номер кредитної картки неперевіреному сервісу. OAuth вирішує цю дилему, дозволяючи користувачу купувати товари онлайн на

багатьох сайтах, котрі дають їм доступ до особистих фінансових даних користувача. Так само, як OAuth дозволяє користувачу увійти на сайт. При здійсненні платежу все, що потрібно зробити, - це використовувати OAuth для входу в свій обліковий запис онлайн-банку і зробити транзакцію, при чому сайт ніколи не дізнається облікові дані. Найкраще те, що кожен зовнішній сайт, на який заходить користувач, не може збирати інформацію про користувачів і продавати. З моменту появи OAuth 2.0 - тепер це стандартна модель - всі передачі даних OAuth повинні здійснюватися по протоколу SSL (Secure Sockets Layer), щоб гарантувати, що найбільш надійні протоколи криптографічної індустрії використовуються для забезпечення максимальної безпеки даних.

Контроль. OAuth не тільки дає користувачам можливість заборонити всім сайтам обмежений доступ до їх даних, але також дозволяє користувачам контролювати, коли закінчується цей тимчасовий доступ. Приємно, що користувачі можуть вибирати, коли термін дії токенів авторизації закінчується.

Витрати. Без необхідності створення повністю надійної системи коментування це звільняє більше часу і грошей для розробників веб-сайтів для роботи над іншими аспектами їх сайту. Крім того, оскільки це відкритий стандарт для авторизації, це також означає, що величезна кількість онлайн-асоціацій може отримати з цього вигоду без шкоди для ліцензійних зборів.

Рух. З усіма зусиллями, пов'язаними з підпискою на новий сайт і приєднанням до товариства, OAuth може зробити речі більш випадковими. Оскільки люди більше ходять по мережі, це означає, що у сайтів більше шансів утримати нових читачів і змусити їх стати постійними відвідувачами, що відмінно підходить для всіх онлайн-підприємств.

Популярність. Більшості людей не подобається стрибати на бета-тестах для нового програмного забезпечення, тому, перш ніж перейти на ще один веб-стандарт, завжди корисно знати, що тенденція вже в русі, і що в майбутньому новий сервіс не зникне. OAuth вже прийнятий Google, Facebook, Twitter і Yahoo, тому, навіть якщо перехід здійснили лише деякі з їх користувачів, це як і раніше дорівнює багатьом мільйонам нинішніх користувачів.

Недоліки протоколу авторизації OAuth 2.0:

Відсутність анонімності. Якщо користувач використовує соціальну мережу Facebook для коментування на іншому сайті, інші можуть не тільки бачити його фотографію в Twitter, а й натискати на нього, щоб побачити, більше інформаці про користувача. Це вибір, який повинен робити користувач, і більшість користувачів Інтернету як і раніше вважають за краще робити анонімні коментарі в Інтернеті, але цей вибір багато ще не готові зробити. Необхідність створення анонімних альтернативних облікових записів може бути такою ж дратівливою, як необхідність щомісяця перемикати десятки налаштувань конфіденційності сайту.

Відсутність насиченості ринку. Незважаючи на те, що мільйони можуть використовувати OAuth через Facebook і Twitter, число інших сайтів, що підтримують OAuth в якості клієнтів, все ще дуже обмежена. Буде потрібно якийсь час, щоб тенденція поширилася, перш ніж користувачі зможуть підключитися до всіх своїх улюблених місцях в мережі і отримати повноцінний досвід роботи в Інтернеті, який ще не пропонував OAuth.

Фішинг. При правильному виконанні OAuth повинен бути безпечним, але сам процес запиту користувачів про необхідність дуже швидкого входу на інший сайт може навчити користувачів бути недбалими і обманювати менш освічених користувачів мережі в тому, що ця практика завжди безпечна. Багато користувачів неминуче стають жертвами переконливо виглядаючих спливаючих вікон, а їх дані піддаються фішингу.

Зловживання даними. Добре, що OAuth обмежує нові сайти в отриманні всіх ваших призначених для користувача даних і продажу їх, але, на жаль, один з найбільших прихильників OAuth вже винен в цих процедурах. У минулому соціальна мережа Facebook безумовно встановлювала погані стандарти для порушення конфіденційності клієнтів, тому турбує те, що тепер у неї буде доступ до ще більшої кількості призначених для користувача даних на декількох сайтах.

2.4 Рекомендації щодо удосконалення рівня захищеності протоколу аторизації OAuth 2.0

Внаслідок інтеграції протоколу авторизації користувачів OAuth 2.0 до новоствореного веб-додатку та проходження процедури тестування додатку за рядом критеріїв були зафіксовані певні потенціальні загрози безеці інйормації, що обробляється в даній системі. В результаті проведених досліджень й тестів було сформовано ряд зауважень та пропозицій для удосконалення механізмів захисту протоколу. Основним кроком для удосконалення захищеності алгоритму є накладання електронно-цифрового підпису на дані з обмеженим доступом при надсиланні їх пост-запитом між елементами системи. Оскільки існує вірогідність здійснення атаки типу “man in the middle” й можливої модифікації даних, електронно-цифровий підпис гарантує перевірку авторства відправника. Постачальником сертифікату рекомендовано використовувати акредитований центр сертифікації ключів «Автор» та кореневі сертифікати центрального засвідчувального органу.

По-друге, після формування сертифікату все одно залишається вірогідність того, що при перехопленні пакетів під загрозою опиниться конфіденційність даних. Для вирішення цієї проблеми запропоновано впровадження другого кроку, а саме шифрування конверту з даними та електронно-цифровим підписом за допомогою асиметричного шифру RSA аба Діффі-Хеллмана. Для процедури шифрування рекомендовано використовувати написані на мові C++ бібліотеки від компанії «Автор». Лише після формування конверту з сертифікатом та шифрування можна передавати дані навіть по незахищеній частині мережі.

Висновки за розділом 2

У розділі 2 було проаналізовано концепції роботи протоколів авторизації користувачів у веб-додатках та інших програмних звстосунках, що сприяли

формуванню рекомендацій щодо удосконалення існуючих рішень. Так, основними цілями аналізу існуючих протоколів в напрямі авторизації користувацьких сесій та надання рекомендацій і розробки нових підходів та алгоритмів є оцінка поточного рівня захищеності інформаційної системи, оцінка відповідності ІС існуючим стандартам в області інформаційної безпеки, аналіз ризиків, які зв`язані з можливістю реалізації загроз безпеки ресурсів та розробка рекомендацій по впровадженню нових і підвищенню ефективності існуючих механізмів безпеки інформаційної системи. Також варто зазначити, що у роботі буде проведена розробка рекомендацій та буде впроваджено ці екомендації до сервісу, який ще не має реалізації. Серед трьох основних методів аналізу інформаційної безпеки алгоритмів найдоцільнішим є комбінований метод, застосування якого дозволяє найбільш повно відобразити рівень захищеності веб-ресурсу.

В якості основоположних стандартів для проведення аналізу роботи протоколів авторизації користувачів у веб-додатках є мжнародні стандарти серії RFC 6000. Наведені стандарти базуються на процесному підході, при цьому стандарт RFC 6789 виступає в якості більш консолідуючого.

Розглянувши механізми захисту протоколу авторизації OAuth 2.0 було виявлено його переваги й недоліки. Внаслідок цього була проведена комплексна робота щодо формуваннння можливостей для удосконалення алгоритму. Після проведеного аналізу було сформовано загальні рекомендації щодо удосконалення роботи цього протоколу.

Для подальшого дослідження основних загроз ІБ веб-ресурсу були досліджені клієнт-серверна архітектура та основні принципи роботи OAuth 2.0 протоколу. Отримана інформація буде використовуватися для проведення тестування існуючої конфігурації протоколу та наведення рекомендацій щодо покращення механізмів захисту даного алгоритму.

Таким чином були виконані поставлені завдання курсової роботи, що стосуються узагальнення основних загроз веб-ресурсів та дослідження методів тестування ІБ.

РОЗДІЛ 3
ПРОЕКТ СТВОРЕННЯ СЕРВІСУ НА ОСНОВІ ПРОТОКОЛУ АВТОРИЗАЦІЇ OAUTH 2.0 З УРАХУВАННЯМ ЗАПРОПОНОВАНИХ РЕКОМЕНДАЦІЙ.

У даному розділі використовується інформація зі специфікації системи BankId, у розробці якої приймав учать автор даної роботи [17].

3.1 Загальні вимоги до проекту

3.1.1 Глосарій

У проекті використано ряд термінів. Ці терміни наведені в таблиці 1.2.

Таблиця 1.2

Терміни та скорочення

Термін, скорочення	Визначення
API	Application Programming Interface. Набір готових класів, процедур, функцій, структур та констант, що надаються програмним комплексом (бібліотекою, сервісом) для використання у зовнішніх програмних продуктах.
OAuth2.0	Відкритий протокол авторизації, який дозволяє отримати третій стороні обмежений доступ до захищених ресурсів користувача без необхідності передавати їй (третій стороні) логін та пароль (http://oauth.net/)
ПАП	Портал, суб'єкт надання адміністративних послуг або інший державний орган, який здійснює дистанційну ідентифікацію користувачів з використанням системи UserId

Продовження до таблиці 1.2

UserId	Система програмно-технічних засобів та організаційно-технологічних заходів для забезпечення інформаційної взаємодії між абонентами в електронній формі. Система UserId призначена для надійного і захищеного передавання, обміну та управління інформаційними потоками між абонентами для проведення дистанційної ідентифікації користувачів системи UserId.
JSON	JavaScript Object Notation. Текстовий формат обміну даними, побудований на JavaScript.
АБС	Автоматизована банківська система - комплекс програмного та технічного забезпечення, спрямований на автоматизацію банківської діяльності. У даному документі використовуються такі приклади АБС: інтернет-банкінг, мобільний банкінг.
ІБ	Веб-портал інтернет-банкінгу. Система дистанційного обслуговування, яка працює в банку (у якій безпосередньо зареєстрований кінцевий користувач, що працює з ПАП).
Мобільний банкінг	Автоматизована система дистанційного обслуговування клієнтів банку щодо управління їх картками та картрахунками за допомогою мобільного телефону. Ключовими параметрами, що ідентифікують клієнта у мобільному банкінгу є номер мобільного телефону та номер платіжної картки клієнта.

Продовження до таблиці 1.2

Авторизація	Процес надання особі/процесу прав на виконання певних дій або доступу до ресурсів, а також процес перевірки (підтвердження) прав при спробі виконання цих дій.
Автентифікація	Електронний процес, що дає змогу підтвердити електронну дистанційну ідентифікацію фізичної або юридичної особи чи походження та цілісність даних в електронній формі.
Посилений сертифікат шифрування	Сертифікат відкритого ключа, виданий акредитованим центром сертифікації ключів, що використовується при обчисленні узгодженого ключа шифрування ключів згідно з Вимогами до форматів криптографічних повідомлень, які затверджені наказом Адміністрації Державної служби спеціального зв'язку та захисту інформації України від 18.12.2012 № 739.

3.1.2 Цілі створення сервісу

Основними цілями створення сервісу на основі протоколу авторизації OAuth 2.0 є забезпечення на порталах адміністративних послуг надійної та зручної верифікації користувача через його власний інтернет-банкінг або будь-який інший банківський сервіс, який передбачає дистанційну автентифікацію та авторизацію користувача як клієнта банку.

3.1.3 Концепція функціонування системи

Функціонально система UserId складається з трьох основних блоків:

1. ПАП (веб-портал), на якому розміщені форми надання адміністративних послуг у електронному вигляді (замовлення довідок, внесення у реєстри тощо). При вході на

портал та/або при замовленні конкретної послуги, користувачу доступна кнопка «авторизація через UserId».

Після натиснення кнопки «авторизація через UserId», користувач перенаправляється на сторінку сервісу «Сервіс UserId» для вибору конкретного банку та проведення процедури автентифікації і авторизації.

В результаті успішного проходу процедури авторизації через UserId формується електронна анкета з персональними даними користувача, після чого користувач автоматично перенаправляється назад на сторінку порталу адміністративних послуг. Одночасно портал отримує від банка через систему UserId анкету з даними користувача для здійснення його (користувача) реєстрації на порталі.

2. UserId (веб-сервер), на якому розміщена сторінка вибору банку та програмні процедури запиту/передачі даних з/до банку або порталу адміністративних послуг.

Після вибору користувачем банку, система перенаправляє його на сторінку сервісу автентифікації та авторизації відповідного банку (наприклад, сторінку логіну відповідного інтернет-банкінгу).

В результаті успішного проходу процедури автентифікації та авторизації клієнта у відповідному сервісі банку, банк формує електронну анкету, підписує її та передає до порталу адміністративних послуг.

3. Банк (АБС: веб-портал інтернет-банкінгу або іншого сервісу банку). У відповідній АБС банку має бути реалізована сторінка для здійснення автентифікації та авторизації клієнта та програмні процедури формування електронної анкети, накладання на неї юридично значимого електронного цифрового підпису (печатки) банку, шифрування даних анкети і передавання цієї анкети до ПАП. Загальна схема функціонування сервісу UserId наведена на рисунку 5.1.

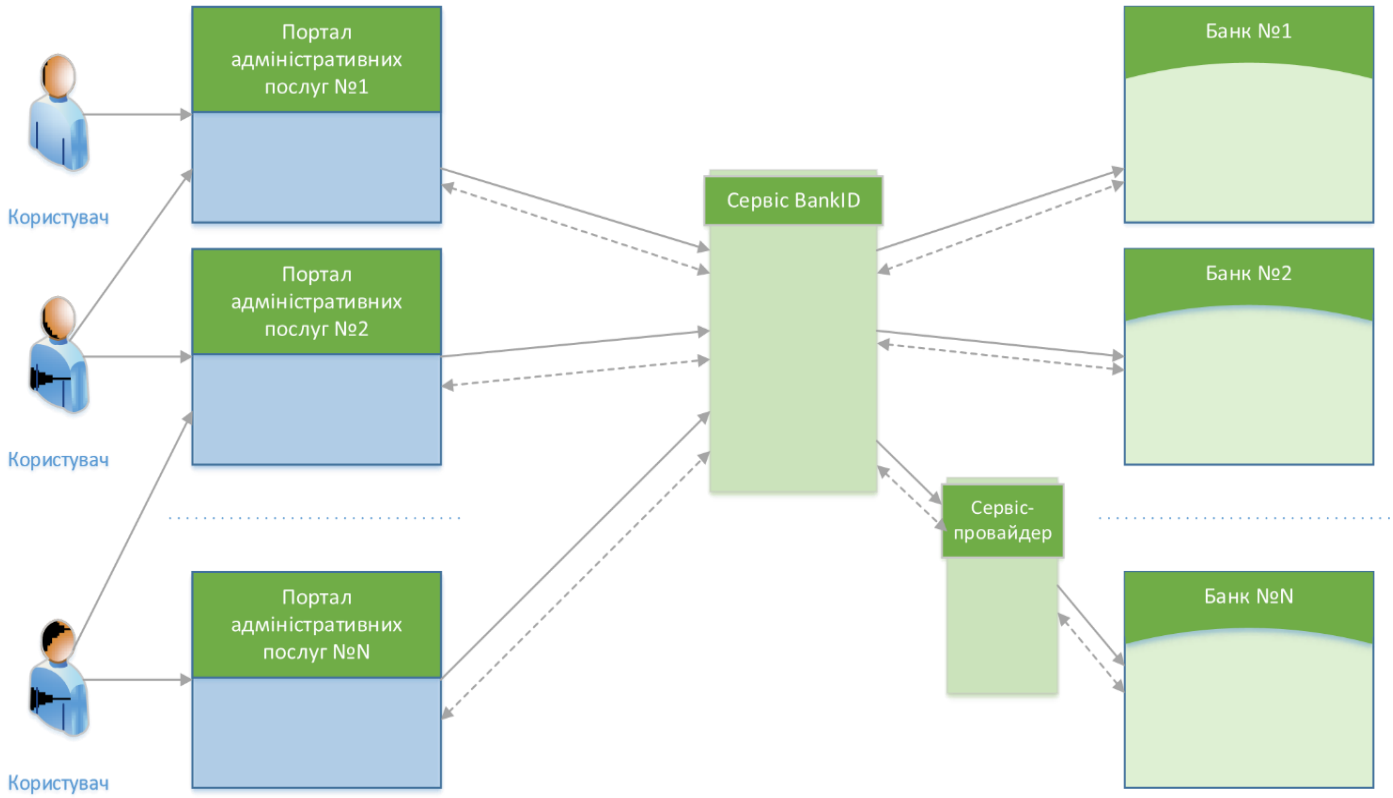

Рисунок 5.1 - Загальна схема функціонування сервісу UserId

Банк може делегувати функцію автентифікації та авторизації користувача певному сервіс-провайдеру, з яким у нього є договірні відносини.

У такому випадку функції автентифікації та авторизації користувача можуть реалізовуватись не безпосередньо в АБС банку, а у відповідній автоматизованій системі сервіс-провайдера.

Сервіс-провайдер забезпечує перевірку автентичності клієнта, генерацію коду авторизації клієнта, надання токену доступу та подальшу перевірку токену доступу, а також взаємодію із БД АБС банку для отримання підписаної анкети з персональними даними клієнта.

У випадку, якщо у схемі отримання персональних даних бере участь сервіс-провайдер, то персональні дані клієнта із БД банку передаються до сервіс-провайдера і

далі в UserId у зашифрованому вигляді таким чином, що сервіс-провайдер не має змоги ознайомитись із змістом анкети.

3.2 Технічна архітектура системи

Взаємодія сервіса UserId з банком відбувається за стандартним протоколом OAuth2.0 згідно зі специфікацією (http://tools.ietf.org/html/rfc6749).

Банк для підключення в систему UserId проходить встановлену процедуру реєстрації, за результатам якої адміністратор центрального вузла UserId видає уповноваженій особі банку параметри з'єднання (Client ID, Client Secret та адреси зворотнього виклику Redirect URI чи Callback URL) та реєструє наступні адреси: DNS адресу банку (clientHost), authorizationUri, accessTokenUri та resourceUri. Зазначені адреси мають бути фіксовані і не повинні містити змінних параметрів (зміни у адресах повинні узгоджуватись з адміністратором центрального вузла UserId). Технічна архітектура системи зображена на рисунку 6.1.

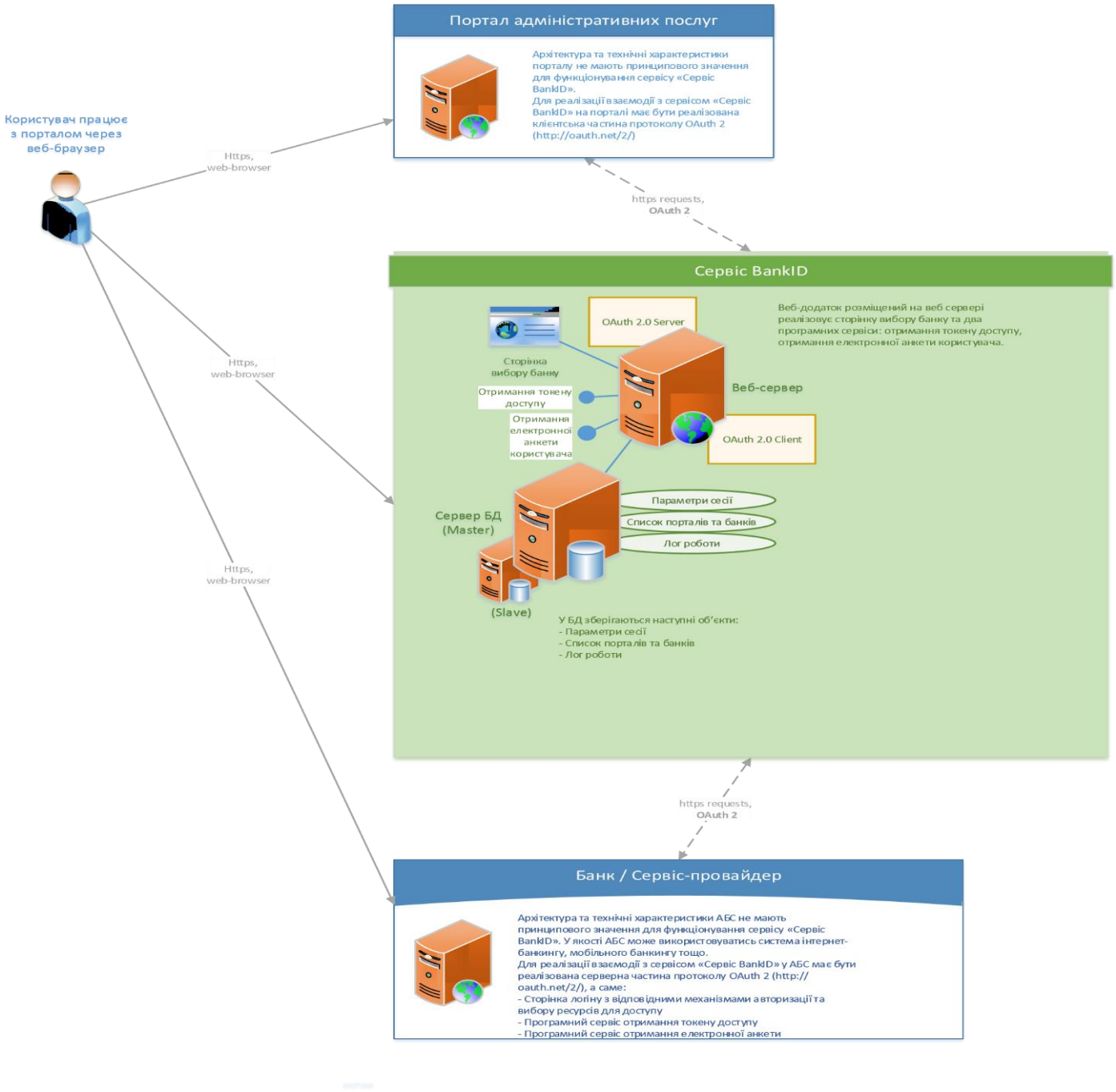

Рисунок 6.1 - Технологічна схема функціонування сервісу UserId

3.2.1 Процедура авторизації

Авторизація згідно стандарту OAuth 2.0 виконується у два етапи (Authorization Code Flow): Перший етап – отримання коду авторизації (запит на адресу authorizationUri);

Другий етап – отримання токену доступу на підставі коду авторизації(запит на адресу accessTokenUri).

Якщо банк делегує функції автентифікації та авторизації сервіс-провайдеру, то описані нижче методи повинні бути реалізовані на стороні сервіс-провайдера. При цьому сторінка проходження користувачем процедури автентифікації та погодження на передачу персональних даних повинна бути брендована під кожен конкретний банк.

Перехід на сторінку АБС методом GET (перший етап)

Запит

```
GET
/v1/bank/oauth2/authorize?response_type=code&client_id=client_id&redirect_uri=callback_url&state=state  HTTP/1.1
```

Параметри цього запиту наведені в таблиці 1.3.

Таблиця 1.3

Парметри запиту

Параметр	Обов'язковість	Опис
client_id	Так	Унікальний ідентифікатор ПАП (видається стороною BankID при реєстрації одноразово)

Продовження до таблиці 1.3

callback_url	Так	Узгоджена заздалегідь адреса сайту ПАП, на яку буде виконана переадресація з кодом авторизації клієнта (authorization code). Адреса фіксована, не повинна містити змінних параметрів (зміни у адресі повинні узгоджуватись зі стороною BankID).
state	Ні, але бажано	Довільний параметр, який буде повернуто при переадресації на адресу, вказану у **callback_url.** Як правило використовується для уникнення CSRF атак), використання рекомендується.

Відповідь на запит

HTTP/1.1 200 OK

Переадресація на сторінку аутентифікації/авторизації користувача АБС (наприклад, логіну у ІБ).

По завершенню авторизації користувача буде виконано переадресацію на вказаний у параметрі redirect_url з поверненням параметру code

Якщо при вказаному запиті виникають помилки, то можливі 2 ситуації:

- Клієнта не вдалося автентифікувати (зокрема не зареєстрований на стороні BankID) та невірно вказано адресу callback_url. В такому випадку опис помилки буде відображено у вікні сайту BankID.

- Клієнта вдалося аутентифікувати і переданий коректний callback_url, проте сталася якась інша помилка – то буде виконано переадресацію на адресу callback_url з наступними параметрами. Типи помилок наведені в таблиці 1.4.

Таблиця 1.4

Типи помилок

Параметр	**Обов'язковість**	**Опис**
error	Так	Один з визначених кодів помилки. Повний перелік стандартних кодів можна знайти за посиланням https://tools.ietf.org/html/rfc6749#section-4.1.2.1
error_description	Ні	Текстовий опис помилки (у UTF-8 кодуванні). Як правило деталізація для розробників.
error_uri	Ні	Адреса, за якою розміщено сторінку з детальним описом помилки, що сталася
state	Ні	Буде передано те ж значення, що використовувалось при початковому виклику (якщо було вказано).

Запит

```
GET
https://bank.com/v1/bank/oauth2/authorize?response_type=code
&client_id=a18a1a63-7ace-434d-bd36-1538ad31d74d
&redirect_uri=https://id.bank.gov.ua/callback.htm&state=2364fc5d-c6b6-4f99-
87a911d2baa32484
```

Відповідь (переадресація з сайта ІБ)

```
GET
https://id.bank.gov.ua/callback.htm?code=2d6f2318cb06cc2c97d948deb9799d608f1d5c97
&state=2364fc5d-c6b6-4f99-87a9-11d2baa32484
```

Запит на отримання токену доступу (access_token) методом POST (другий етап)

```
POST /v1/bank/oauth2/token HTTP/1.1
Content-Type: application/x-www-form-urlencoded
code=code client_id=client_id& client_secret=client_secret& redirect_uri=callback_url
```

- Парметри даного запиту наведені в таблиці 1.5.

Таблиця 1.5

Параметри запиту

Параметр	Обов'язковість	Опис
code	Так	Код авторизації клієнта (authorization code), отриманий на попередньому кроці
client_id	Так	Унікальний ідентифікатор ПАП (видається стороною BankID при реєстрації одноразово)
client_secret	Так	Таємний код ПАП (видається стороною BankID при реєстрації одноразово)
callback_url	Так	Узгоджена заздалегідь адреса сайту ПАП, у даному випадку використовується для переадресації у разі виникнення помилок при отриманні токену доступу. Адреса фіксована, не повинна містити змінних параметрів (зміни у адресі повинні узгоджуватись зі стороною BankID).

Відповідь на запит:

```
HTTP/1.1 200 OK
Content-Type: application/json
{
```

"token_type":"bearer",
"access_token":"db8814c6d97cbad2a02db80e17d4676fab5914c6", "expires_in":3600,
"refresh_token":"91ee282ccda4e1af6dbd0da4920b206866278f5e" }

Запит на продовження дії токену доступу(access_token) методом POST

Для можливості продовжити дію вже отриманого токену доступу (без необхідності виконання всіх попередній викликів) необхідно виконати наступний запит.

```
POST /v1/bank/oauth2/token HTTP/1.1
Content-Type: application/x-www-form-urlencoded

client_id=client_id& client_secret=client_secret& refresh_token=refresh_token&
grant_type=refresh_token
```

- Парметри даного запиту наведені в таблиці 1.6.

Таблиця 1.6

Параметри запиту

Параметр	Обов'язковість	Опис
client_id	Так	Унікальний ідентифікатор ПАП (видається стороною BankID при реєстрації одноразово)
client_secret	Так	Таємний код ПАП (видається стороною BankID при реєстрації одноразово)
refresh_token	Так	Значення токену доступу (refresh_token), запит на продовження дії якого подається

Відповідь на запит:

```
HTTP/1.1 200 OK
Content-Type: application/json
```

```
{
 "token_type":"bearer",
 "access_token":"db8814c6d97cbad2a02db80e17d4676fab5914c6",  "expires_in":3600
}
```


3.2.2 Процедура отримання даних по клієнту

Отримання даних відбувається на основі токену доступу (access_token), отриманого у ході авторизації (згідно попереднього пункту). Токен доступу необхідно передавати в заголовку запиту (headers) у вигляді:

Authorization: “Bearer access_token”

Центральний вузол UserId для шифрування відповіді від банку (анкети з даними про клієнта) повинен передати посилений сертифікат шифрування того ПАП, для якого здійснюється автентифікація клієнта. Сертифікат передається в атрибуті "cert" у форматі base64. Порядок шифрування анкети визначено в пункті 2.3.3. цієї Специфікації. Також сторона UserId в запиті повинна вказати, який саме набір полів по клієнту потрібно повернути після виклику. Опис усіх допустимих полів задається у вигляді JSON об’єкту:


```
{
 "type":"physical",
"cert":"dEBIGx1pdL6dt1ngaOEfPvt/ik9eUpDuz90rm/Vk23v+FtiKJEvGnuea9FGjvBMG
lFZ
S4zhg2IYIHGOhlBVYrZcez6udotjZlCLGZ7zwPuFo0XypKDQj5qpR7w0rFFZNjcPH3J
W2I zEUv",
    "fields":["firstName","middleName","lastName","phone","inn","birthDay"],
"scans":[
      {"type":"passport","fields":["link","dateCreate","extension"]},
```

```
        {"type":"zpassport","fields":["link","dateCreate","extension"]}
      ],
      "addresses":[

{"type":"factual","fields":["country","state","area","city","street","houseNo","flatNo"
]},
{"type":"birth","fields":["country","state","area","city","street","houseNo","flatNo"]}
    ],
    "documents":[

{"type":"passport","fields":["series","number","issue","dateIssue","dateExpiration","i
ssueCou ntryIso2"]},

{"type":"zpassport","fields":["series","number","issue","dateIssue","dateExpiration",
"issueCo untryIso2"]}
    ]
}
```


- Тобто для отримання необхідних даних про клієнта потрібно передати усі поля структури або підмножину полів (відсутність поля вказує на те, що дані по цьому полю не потрібні). Опис допустимих полів наведений в таблиці 1.7 [17].

Таблиця 1.7

Опис допустимих полів

Блок	Поле	Опис
cert	cert	Посилений сертифікат шифрування (у форматі base64), обов’язково для заповнення
fields	lastName	Прізвище
	firstName	Ім’я
	middleName	По батькові
	phone	Номер мобільного телефону
	inn	Ідентифікаційний код
	birthDay	Дата народження
	sex	Стать
	email	Електронна адреса
addresses	type	Тип адреси, допустимі значення factual і birth (фактична адреса та адреса народження відповідно)
	country	Країна
	state	Область
	area	Район
	city	Місто
	street	Вулиця
	houseNo	Номер будинку
	flatNo	Номер квартири

Продовження до таблиці 1.7

documents	type	Тип документу, допустимі значення passport, zpassport і ident (паспорт, закордонний паспорт, та посвідчення особи відповідно).
	typeName	Назва документу
	series	Серія документу
	number	Номер документу
	issue	Ким виданий документ
	dateIssue	Коли виданий
	dateExpiration	Термін дії
	issueCountryIso2	Країна видачі документу
scans	type	Копія документів, допустимі значення passport і zpassport (паспорт та закордонний паспорт відповідно)
	scanFile	файл сканкопії (у форматі base64)
	dateCreate	Дата створення документу
	extension	Розширення файлу

2.2.1 Запит даних по клієнту

Для отримання даних по клієнту відбувається запит на адресу resourceUri.

Запит

POST /v1/bank/resource/client HTTP/1.1

```
Authorization: Bearer access_token
Content-Type: application/json
```

```
{
        "type":"physical",
        "cert":"dEBIGx1pdL6dt1ngaOEfPvt/ik9eUpDuz90rm/Vk23v+FtiKJEvGnuea9FGjvB
MGlFZ
S4zhg2IYIHGOhlBVYrZcez6udotjZlCLGZ7zwPuFo0XypKDQj5qpR7w0rFFZNjcPH3JW2
I zEUv",
        "fields":["firstName","middleName","lastName","phone","inn","clId","clIdText","birt
hDay"],
          "scans":[{"type":"passport","fields":["link","dateCreate","extension"]}],

        "addresses":[{"type":"factual","fields":["country","state","area","city","street","house
No","flat No"]}],
          "documents":[{"type":"passport","fields":
           ["series","number","issue","dateIssue","dateExpiration","issueCountryIso2"]}] }
```


Інформація про поле токену доступу наведена в таблиці 1.8.

Таблиця 1.8

Поле токен доступу

Параметр	Обов’язковість	Опис
access_token	Так	Токен доступу, отриманий в процесі авторизації.

Відповідь на запит:

Json-об’єкт, в якому банк надає свій посилений сертифікат шифрування та анкету з даними про клієнта у зашифрованому вигляді. Посилений сертифікат шифрування, який буде використовуватись стороною ПАП для розшифрування даних, міститься в атрибуті "cert" у форматі base64. Дані про клієнта у зашифрованому вигляді (base64-encoded) містяться в атрибуті "customerCrypto".

```
HTTP/1.1 200 OK
        Content-Type: application/json

        {
           "state": "ok",
        "cert":"dEBIGx1pdL6dt1ngaOEfPvt/ik9eUpDuz90rm/Vk23v+FtiKJEvGnuea9FGjvB
MGlFZ
S4zhg2IYIHGOhlBVYrZcez6udotjZlCLGZ7zwPuFo0XypKDQj5qpR7w0rFFZNjcPH3JW2
I zEUv",
           "customerCrypto":
        “dEBIGx1pdL6dt1ngaOEfPvt/ik9eUpDuz90rm/Vk23v+FtiKJEvGnuea9FGjvBMGlF
ZS4zhg2
IYIHGOhlBVYrZcez6udotjZlCLGZ7zwPuFo0XypKDQj5qpR7w0rFFZNjcPH3JW2IzEUv/
4
bXQWqYCccma03b3lbva+YJ/Txox1CMfyV4jJ5fXeCMOjEWxwctEc7mXNzPfcbKMoqr0
4
8uvW9HTiPkjsLIU5jgTKJVdgoanZI4712dmQev5UzMKqNYvOwfJ+hU872kCSD1wfgVaJ
U0qP6yURcK80ys2K5OvUpa9uIHwmL7KmnxMDhB4hLr5CQPl1XZ09RnNykgS/4cQ”
        }
```

Після розшифрування отримана інформація має бути у вигляді наведеного Json-об’єкта "customer"

```
   "customer": {
      "type": "physical",
      "inn": "112233445566",
      "sex": "м",
      "email": "geraschenko@gmail.com",
      "birthDay": "20.01.1973",
      "firstName": "ПЕТРО",
```

```
    "lastName": "ГЕРАЩЕНКО",
    "middleName": "ІВАНОВИЧ",
    "phone": "+380961234511",
    "addresses": [
      {
        "type": "factual",
        "country": "UA",
        "state": "ВОЛИНСЬКА",
        "city": "Ківерці",
        "street": "Незалежності",
        "houseNo": "62",
        "flatNo": "12"
      },
    ],
    "documents": [
      {
        "type": "passport",
        "series": "AA",
        "number": "222333",
        "issue": "Ківерцівським РО УМВД",
        "dateIssue": "15.03.1999",
        "issueCountryIso2": "UA"
      }
    ],
    "scans": [
      {
        "type": "passport",          "scanFile":
"H4sIAAAAAAAAEAOS8dVhczZYuHtxdQiBA4wRraNwdgru7NO7uFpwEd3cLwT...",
```

```
            "dateCreate": "09.04.2015",
            "extension": "zip"
        }
    ]
}
```

3.2.3 Захист інформації в системі UserId

3.2.3.1 Загальні положення

Передавання інформації між абонентами системи UserId повинно здійснюватися із забезпеченням конфіденційності та контролю цілісності.

Абоненти-надавачі послуг, абоненти-ідентифікатори, центральний вузол системи UserId забезпечують ідентифікацію й автентифікацію у своїх інформаційнотелекомунікаційних системах з використанням криптографічного протоколу TLS (Transport Layer Security), вимоги до якого наведено нижче.

В інформаційно-телекомунікаційних системах абонентів-надавачів послуг, абонентів ідентифікаторів, центрального вузла системи UserId здійснюється реєстрація подій шляхом ведення журналу аудиту.

Журнал аудиту абонента-надавача послуг повинен містити відомості про факт отримання анкети від центрального вузла системи UserId, результат розшифрування анкети, результат перевірки ЕЦП, накладеного абонентом- ідентифікатором.

Журнал аудиту абонента- ідентифікатора повинен містити відомості про факт звернення користувача системи UserId, результат опрацювання звернення користувача системи UserId, факт відправлення анкети центральному вузлу системи UserId.

Журнал аудиту центрального вузла системи UserId повинен містити відомості про факт отримання анкети від абонента-ідентифікатора, факт відправлення анкети абоненту надавачу послуг.

Абоненти-надавачі послуг, абоненти-ідентифікатори, адміністратор центрального вузла системи UserId мають право самостійно визначати додаткові події, що фіксуються у відповідних журналах аудиту.

Усі записи в журналах аудиту повинні містити опис події, дату та час події, а також забезпечувати ідентифікацію суб'єкта, що ініціював подію.

Журнали аудиту повинні мати захист від несанкціонованого доступу, модифікації та знищення (руйнування).

Взаємодія інформаційно-телекомунікаційної системи центрального вузла системи UserId з інформаційно-телекомунікаційними системами абонентів-надавачів послуг та абонентів-ідентифікаторів здійснюється виключно з використанням параметрів та адрес, які узгоджені з адміністратором центрального вузла системи UserId [17].

3.2.3.2 Вимоги до використання криптографічного протоколу TLS та відповідних сертифікатів відкритих ключів

Абоненти-надавачі послуг, абоненти-ідентифікатори, адміністратор центрального вузла системи UserId для встановлення безпечного з'єднання між собою та з користувачами системи UserId повинні використовувати криптографічний протокол TLS не нижче версії 1.2, а також відповідні особисті ключі та сертифікати відкритих ключів.

У протоколі TLS допускаються різні криптографічні набори.

Криптографічний набір узгоджується між клієнтом та сервером під час встановлення з'єднання. Клієнт передає серверу список підтримуваних криптографічних наборів, а сервер вибирає один з них для захисту інформації.

Сервери не повинні використовувати криптографічні набори, які не використовують шифрування або для шифрування використовується алгоритм RC4 (в якості EncryptionAlg встановлено NULL або RC4).

Для шифрування інформації повинні використовуватись симетричні криптографічні алгоритми з довжиною ключа не менше 128 біт.

Не рекомендується використовувати криптографічні набори, які для обміну ключами використовують статичний RSA. Довжина відкритого ключа RSA повинна бути не меншою 2048 біт. Заборонено використовувати криптографічні набори які використовують попередньо узгоджений загальний секретний ключ (PSK).

Для узгодження сеансових ключів використовуються протоколи DHE та ECDHE. Довжина відкритого ключа для протоколу DH повинна бути не меншою 2048 біт. Довжина відкритого ключа для протоколу ECDHE повинна бути не меншою 256 біт.

Рекомендується використовувати такі криптографічні набори:

TLS_DHE_RSA_WITH_AES_128_GCM_SHA256;

TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256;

TLS_DHE_RSA_WITH_AES_256_GCM_SHA384;

TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384.

Абоненти-надавачі послуг, абоненти-ідентифікатори, центральний вузол системи UserId використовують сертифікати відкритих ключів розширеної перевірки (Extended Validation Certificates, далі – EV SSL сертифікат) у форматі X.509 версії 3.

Рекомендується використовувати браузери провідних розробників (таких як Apple, Google Inc., Microsoft Corporation, Mozilla Foundation, Opera Software ASA) та отримувати

EV SSL сертифікати від центрів сертифікації ключів (certificate authority/CA), довірених для відповідних браузерів.

EV SSL сертифікат не повинен мати тип Wildcard. В розширенні "Додаткові дані підписувача" ("subjectAlternativeName") EV SSL сертифіката не допускається

використання URL, відмінного від URL, зазначеного в "реквізиті підписувача" ("commonName") поля "Підписувач" ("subject").

3.2.3.3 Вимоги до забезпечення конфіденційності та контролю цілісності електронної анкети

Абонент-ідентифікатор перед передаванням електронної анкети з інформацією про користувача через систему UserId:

− накладає на цю електронну анкету електронний цифровий підпис, що відповідно до вимог Закону України “Про електронний цифровий підпис” прирівнюється до печатки;

− шифрує електронну анкету з використанням посиленого сертифікату шифрування того абонента-надавача послуг, якому передає електронну анкету.

Шифрування/розшифрування електронної анкети відбувається згідно з Вимогами до форматів криптографічних повідомлень, які затверджені наказом Адміністрації Державної служби спеціального зв’язку та захисту інформації України від 18.12.2012 № 739.

Абонент-ідентифікатор накладає на електронну анкету електронний цифровий підпис з використанням формату “ЕЦП з повним набором даних перевірки” (CAdES-X Long відповідно до ДСТУ ETSI TS 101 733:2009).

Вимоги до формату підписаних даних затверджені наказом Міністерства юстиції України, Адміністрації Державної служби спеціального зв’язку та захисту інформації України від 20.08.2012 № 1236/5/453 та зареєстровані в Міністерстві юстиції України 20.08.2012 за № 1401/21713 (зі змінами) [17].

Висновки за розділом 3

У розділі 3 було побудовано сервіс на основі критеріїв стандартів серії RFC 6000 та протоколу авторизації OAuth 2.0 з запропонованими удоскаоналеннями механізмів захисту інформації, а саме асиметричним шифруванням та використанням електронно-цифрових підписів. Проект було успішно виконано, протестовано, а також розгорнуто на серверах з подальшим тестуванням, налаштуванням конфігураційних файлів, веб серверів, токенів. Внаслідок тестування був отриманий позитивний результат. Мети досягнуто. Варто зазначити, що лише після введення удосконалень вдалося досягти передбачених міжнародними стандартами норм.

ВИСНОВКИ

У курсовій роботі було проаналізовано основні положення нормативно-методичної бази з роботи протоколів авторизації, досліджено принципи побудови веб-ресурсу з авторизацію користувачів, найпоширеніші загрози ІБ веб-ресурсу під час проаедення авторизації користувачів. У результаті виконаної роботи був проведений аналіз роботи протоколу авторизації OAuth 2.0. Було розглянуто його переваги і недоліки, а також можливості покращити роботу протоколу шляхом покращення показників безпеки. Також був проведений аналіз інших протоколів авторизації, які існують на даний момент, було розглянуто їх переваги та недоліки та обрано найкращий серед них. Найкращі показники виявилися у протоколу OAuth 2.0.

На основі досліджень було сформовано рекомендації щодо покращення роботи протоколу авторизації OAuth 2.0. Для досягнення поставленої мети роботи було використано метод аналізу відповідності рівня захищеності критеріям стандартів RFC 6000.

За результатами сформованих рекомендацій була проведена робота над розробкою сервісу, який побудований на основі протоколу OAuth 2.0 з запропонованими покращеннями, а саме асиметричним шифруванням та використанням електронно-цифрових підписів. Проект було успішно виконано, протестовано, а також розгорнуто на серверах з подальшим тестуванням, налаштуванням конфігураційних файлів, веб серверів, токенів..

Практичне значення роботи полягає у формуванні висновків та рекомендацій щодо рівня захищеності веб-ресурсу, який використовує протоколи авторизації користувачів та побудові сервісу на основі удосконалених механізмів. Розроблений сервіс та виведені рекомендації можуть слугувати прикладом для подальшої побудови інших, більш захищених систем.

Мету роботи досягнуто, поставлені задачі виконано.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1) Аверченков В.И. Аудит информационной безопасности. Учебное пособие для вузов.– М.: ФЛИНТА, 2016 – 269 с.

2) Усач Б.Ф. Аудит: Навч. посіб. – К.: Знання-Прес, 2002.

3) Скабцов Н. Аудит безопасности информационных систем. — СПб.: Питер, 2018. — 272 с.

4) Курило А.П., Зефиров С.Л., Голованов В.Б. Аудит информационной безопасности.– М.:Издательская группа «БДЦ-пресс», 2006.-304с.,с вкл

5) Стандарты информационной безопасности [Електронний ресурс]. – Режим доступу: http://ypn.ru/177/international-standards-of-information-technologies-security/

6) COBIT 5 [Електронний ресурс]. – Режим доступу: http://www.isaca.org/cobit/pages/default.aspx

7) COBIT 5 (Control Objectives for Information and related Technology) [Електронний ресурс]. – Режим доступу: https://managementmania.com/en/cobit-control-objectives-for-information-and-related-technology

8) COBIT 5[Електронний ресурс] [Електронний ресурс]. – Режим доступу: Режим доступу:http://www.wikiitil.ru/books/Cobit-5_frm_rus_0813.pdf

9) ГОСТ Р ИСО/МЭК 27001-2006. Информационная технология. Методы средства обеспечения безопасности. Системы менеджмента информационной безопасности [Електронний ресурс]. – Режим доступу: http://docs.cntd.ru/document/gost-r-iso-mek-27001-2006

10) ISO/IEC 27001 [Електронний ресурс]. – Режим доступу: https://ru.wikipedia.org/wiki/ISO/IEC_27001

11) ISO/IEC 27002 [Електронний ресурс]. – Режим доступу: https://ru.wikipedia.org/wiki/ISO/IEC_27002

12) Симонов С. Аудит безопасности информационных систем [Електронний ресурс]. – Режим доступу: www.jetinfo.ru/Sites/new/.../1999_9.9DB939D4F5F84A4AB9848C1D5A04E0CB.pdf

13) OWASP Methodology [Електронний ресурс]. – Режим доступу:https://www.owasp.org/index.php/Main_Page

14) OWASP Application Security Verification Standard 3.0.1 [Електронний ресурс]. – Режим доступу:https://www.owasp.org/index.php/Main_Page

15) НДТЗІ 2.5-010-03 «Вимоги до захисту інформації WEB-сторінки від несанкціонованого доступу»

16) Ханін М. Вдосконалення механізмів захисту протоколу OAuth 2.0, 2019

17) Специфікація BankID [Електронний ресурс]. – Режим доступу: https://github.com/UNITY-BARS/BankID

18) The Web Application Hacker’s Handbook. Dafydd Stuttard, Marcus Pinto.-2011

19) URL (Единый указатель ресурса) [Електронний ресурс]. – Режим доступу: https://developer.mozilla.org/ru/docs/%D0%A1%D0%BB%D0%BE%D0%B2%D0%B0%D1%80%D1%8C/URL

20) Наиболее распространенные угрозы [Електронний ресурс]. – Режим доступу: https://www.intuit.ru/studies/courses/10/10/lecture/300

21) Основные угрозы безопасности сайта [Електронний ресурс]. – Режим доступу: https://habr.com/company/pentestit/blog/279787/

22) Утечки данных [Електронний ресурс]. – Режим доступу: http://www.tadviser.ru/index.php/%D0%A1%D1%82%D0%B0%D1%82%D1%8C%D1%8F%3A%D0%A3%D1%82%D0%B5%D1%87%D0%BA%D0%B8_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85

23) OWASP Top 10 – 2017.The Ten Most Critical Web Application Security Risks

24) D5.1 Security Testing Methodology

25) Gray Box VS Black Box VS White Box Testing: Briefly Explained Difference [Електронний ресурс]. – Режим доступу: http://testorigen.com/gray-box-vs-black-box-vs-white-box-testing-briefly-explained-difference/

26) Козлов Д. Д., Петухов А. А.Методы обнаружения уязвимостей в web-приложениях

27) Fuzzing: исследование уязвимостей методом грубой силы. – Пер. с англ. – СПб.: Символ$Плюс, 2009. – 560 с.

28) Ильенко Ф.В., Приходько Т.А. Проблемы уязвимости web и средства для анализа безопасности web-приложений. Інформаційні управляючі системи та комп'ютерний моніторинг (ІУС КМ - 2013) - 2013 / Матеріали ІІІ міжнародної науково-технічної конференції студентів, аспірантів та молодих вчених. —Донецьк, ДонНТУ — 2013

29) Анализ исходного кода [Електронний ресурс]. – Режим доступу: https://www.viva64.com/ru/t/0075/

30) Наказ «Про затвердження Положення про контроль за функціонуванням системи технічного захисту інформації» 11 січня 2000 р.

31) Наказ «Про затвердження Зводу відомостей, що становлять державну таємницю» від 17 серпня 2005 р.

32) Application Threat Modeling [Електронний ресурс]. – Режим доступу: https://www.owasp.org/index.php/Application_Threat_Modeling

33) ISO/IEC 15408:2000 – Information technology – Security techniques – Evaluation criteria for IT security. – Part 1: Introduction and general model. ISO/IEC 15408:2000 – Information technology – Security techniques – Evaluation criteria for IT security. – Part 2: Security functional requirements. ISO/IEC 15408:2000 – Information technology – Security techniques – Evaluation criteria for IT security. – Part 3: Security assurance requirements. ISO/IEC WD 15292 – Information Technology – Security techniques – Protection Profile registration procedures – 1999.

34) CEM-97/017 – Common Evaluation Methodology for Information Technology security. – Part 1: Introduction and general model. – Ver. 0.6 – 1997.

35) CEM-99/008 – Common Evaluation Methodology for Information Technology security. – Part 2: Evaluation Methodology. – Ver. 0.6 – 1999.

36) Горбенко И. Д., Потий А. В, Терещенко П. И. Критерии и методология оценки безопасности информационных технологий // Радиотехника. Всеукраинский межвед. научн.-техн. сб. – 2000. – Вып. 114. – С. 25-38.

37) Скрыпник Л. В, Потий А. В. Гибкость и специализация профиля защиты автоматизированной системы // Радиотехника. Всеукраинский межвед. научн.-техн. сб. – 2001. – Вып. 119. – С. 17-21.

38) Горбенко И. Д., Потій А. В., Терещенко П. И. Рекомендації міжнародних стандартів по оцінці безпеки інформаційних технологій // Матеріали третьої міжнародної науково-практичної конференції «Безпека інформації в інформаційно-телекомунікаціних системах». – Київ 2000.– С. 150-160.

39) Бондаренко М. Ф., Черних С. П., Горбенко И. Д., Замула А. А., Ткач А. А. Методологічні основи концепції і політики безпеки інформаційних технологій // Радіотехніка. Всеукраїнський міжнар. наук.-техн. сб. – 2001. – Вип. 119. – С. 5-17.

40) ISO 7498-2:1989 – Information technology – Open Systems Interconnection – Basic reference model – Part 2: Security architecture.

41) ISO/IEC 10181-(1-7):1996 – Information technology – Open Systems Interconnection – Security frameworks for open systems.

42) Закон України «Про Національну програму інформатизації» № 74/98-ВР – 1998 – № 27-28.

43) Щербо В. К. Стандарты вычислительных сетей. Взаимосвязи сетей. Справочник. – М.: Кудиц-образ, 2000. – 198 с.

44) Gary Stoneburner. Underlying Technical Models for Information Technology Security – NIST Special Publications – 2001.

45) ISO/IEC 13335-(1-3): 1998 – Information technology –Guidelines of IT Security.

46) ISO/IEC 17799:2001 – Information technology. – Information Security Management – Code of Practice for Information Security Management.

47) BS 7799: 1995 – Code of Practice for Information Security Management.

48) Bundesmat fur Sicherheit in der Informationstechnik. IT Baseline Protection Manual. – 1998.

49) Анищенко В. В. Оценка информационной безопасности // PC Magazine/RE – № 2. – 2000. – С. 7–11.

50) НД ТЗІ 1.1 – 002 – 99. Основні положення по захисту інформації в комп'ютерних системах від несанкціонованого доступу.

51) НД ТЗІ 1.1 – 003 –99. Термінологія в області захисту інформації в комп'ютерних системах від несанкціонованого доступу.

52) НД ТЗІ 2.5 – 004 – 99. Критерії оцінки захищеності інформації в комп'ютерних системах від несанкціонованого доступу.

53) НД ТЗІ 3.7. – 001 – 99. Методичні вказівки по розробці технічного завдання на створення комплексної системи захисту інформації в автоматизованій системі.

54) НД ТЗІ 2.5. – 005 – 99. Класифікація автоматизованих систем і стандартні профілі захищеності оброблювальної інформації від несанкціонованого доступу.

КАЛЕНДАРНИЙ ПЛАН ВИКОНАННЯ КУРСОВОЇ РОБОТИ

Тема: __ Механізми захисту протоколу авторизації OAuth 2.0

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Найменування етапів курсової роботи	Термін виконання робіт (початок-кінець)	Відмітка про виконання
1	Уточнення постановки задачі	19.01.2020 – 20.01.2020	*виконано*
2	Аналіз літератури	21.01.2020- 23.01.2020	*виконано*
3	Формування його переваг і недоліків	23.01.2020- 28.01.2020	*виконано*
4	Формування рекомендацій для покращення роботи протоколу	28.01.2020- 09.02.2020	*виконано*
5	Створення сервісу на основі протоколу з урахуванням запропонованих рекомендацій. Тестування	10.02.2020- 20.04.2020	*виконано*
6	Формування висновків і рекомендацій для вдосконалення системи авторизації користувачів	21.04.2020- 07.05.2020	*виконано*
7	Оформлення пояснювальної записки	8.05.2020	*виконано*
8	Підготовка до захисту курсової роботи	9.05.2020	*виконано*

Студент **_____Ханін М. Ю._____**

Керівник **___Ткаченко О. М._____**

“_____”_____