



УДК 004.4:336.7

[https://doi.org/10.52058/3041-1254-2025-4\(14\)-1031-1047](https://doi.org/10.52058/3041-1254-2025-4(14)-1031-1047)

Курніцький Дмитро Петрович аспірант кафедри автоматизації та інтелектуальних інформаційних технологій, Вінницький національний технічний університет, м. Вінниця, +380965587821, <https://orcid.org/0009-0000-3190-9514>

Сторчак Володимир Григорович кандидат технічних наук, доцент кафедри автоматизації та інтелектуальних інформаційних технологій, Вінницький національний технічний університет, м. Вінниця, +380976110591, <https://orcid.org/0009-0004-7965-2163>

ЗАСТОСУВАННЯ МЕТОДІВ АВТОРИЗАЦІЇ ІЗ ВИКОРИСТАННЯМ IDENTITY SERVER ТА ПРИНЦИПІВ RBA ДЛЯ ПОКРАЩЕННЯ УПРАВЛІННЯ ТА ЗАХИСТУ ДАНИХ КОРИСТУВАЧІВ У СИСТЕМІ ОБРОБКИ ТА ПІДТРИМКИ БАНКІВСЬКИХ ТРАНЗАКЦІЙ ІЗ РОЗГОРТАННЯМ У ХМАРІ

Анотація. У цій статті представлено рішення для оптимізації та покращення процесу авторизації та автентифікації у системах для підтримки та обробки банківських транзакцій. Рішення побудоване на основі Identity Server та принципів Risk-Based Authentication (RBA) для покращення управління доступом та захисту даних користувачів у системах обробки та підтримки банківських транзакцій. Запропоноване рішення базується на мікросервісній архітектурі, що забезпечує високу ефективність, масштабованість та надійність фінансових операцій. Identity Server використовується для централізованого управління автентифікацією за допомогою OpenID Connect та OAuth 2.0, що дозволяє впроваджувати адаптивні методи перевірки доступу.

Методологія Risk-Based Authentication забезпечує динамічну оцінку рівня ризику транзакцій, використовуючи поведінковий аналіз, дані геолокації та пристроїв доступу. Це дозволяє мінімізувати загрози шахрайства та покращити захист персональних даних. Реалізація багатофакторної автентифікації (MFA) та шифрування токенів дозволяє забезпечити відповідність вимогам безпеки, зокрема PCI DSS.

Identity Server дозволяє зменшити час автентифікації користувачів на 40%, забезпечуючи затримку у відповідь запитів нижче 100 мс, навіть при обробці понад 10 000 запитів одночасно. Це досягається завдяки розподіленому зберіганню токенів і кешуванню даних автентифікації. Застосування багатофакторної автентифікації (MFA) і шифрування на рівні токенів суттєво



знижує ризики витоку даних, забезпечуючи відповідність системи вимогам стандартів PCI DSS. У порівнянні з традиційними системами, інтеграція Identity Server та принципів RBA підвищує ефективність управління обліковими записами на 30% завдяки автоматизації обробки сесій і спрощенню процесу відновлення доступу.

Результати тестування продемонстрували, що система забезпечує масштабовану та надійну підтримку банківських транзакцій, дозволяючи обробляти понад 1 мільйон операцій на добу без зниження швидкості або стабільності. Запропонована технологія покращує досвід користувачів і підвищує загальну безпеку платіжних операцій.

Ключові слова: Хмарні технології, Identity Server, RBA, управління доступом, банківські транзакції, безпека даних, масштабованість.

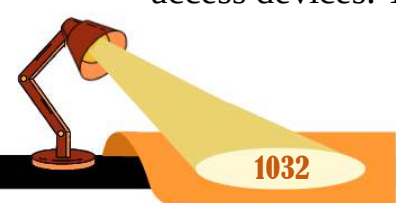
Kurnitskyi Dmytro Petrovych Postgraduate student of the Department of Automation and Intelligent Information Technologies at Vinnytsia National Technical University, Vinnytsia, +380965587821, <https://orcid.org/0009-0000-3190-9514>

Storchak Volodymyr Hryhorovych PhD in Technical Sciences, Associate Professor of the Department of Automation and Intelligent Information Technologies, Vinnytsia National Technical University, Vinnytsia, +380976110591, <https://orcid.org/0009-0004-7965-2163>

APPLICATION OF IDENTITY SERVER-BASED AUTHORISATION METHODS AND RBA PRINCIPLES TO IMPROVE MANAGEMENT AND PROTECTION OF USER DATA IN A CLOUD-DEPLOYED BANKING TRANSACTION PROCESSING AND SUPPORT SYSTEM

Abstract. This article presents a solution for optimizing and improving the authorization and authentication process in systems for supporting and processing banking transactions. The solution is built on the basis of Identity Server and the principles of Risk-Based Authentication (RBA) to improve access management and user data protection in systems for processing and supporting banking transactions. The proposed solution is based on a microservice architecture that provides high efficiency, scalability, and reliability of financial transactions. Identity Server is used for centralized authentication management using OpenID Connect and OAuth 2.0, which allows for the implementation of adaptive access verification methods.

The Risk-Based Authentication methodology provides a dynamic assessment of the risk level of transactions using behavioral analysis, geolocation data, and access devices. This minimizes fraud threats and improves personal data protection.





The implementation of multi-factor authentication (MFA) and token encryption ensures compliance with security requirements, in particular PCI DSS.

Identity Server reduces user authentication time by 40%, providing a response latency of less than 100 ms, even when processing more than 10,000 requests simultaneously. This is achieved through distributed token storage and authentication data caching. The use of multi-factor authentication (MFA) and token-level encryption significantly reduces the risk of data leakage, ensuring compliance with PCI DSS standards. Compared to traditional systems, the integration of Identity Server and RBA principles increases the efficiency of account management by 30% by automating session processing and simplifying the access recovery process.

Test results have shown that the system provides scalable and reliable support for banking transactions, allowing it to process more than 1 million operations per day without reducing speed or stability. The proposed technology improves user experience and increases the overall security of payment transactions.

Keywords: Cloud technologies, Identity Server, RBA, access control, banking transactions, data security, scalability.

Постановка проблеми. Зростання цифрових банківських операцій вимагає нових підходів до обробки великих обсягів даних, забезпечення їхньої безпеки та відповідності сучасним стандартам. Хмарні технології стали ключовим елементом модернізації банківських систем, дозволяючи автоматизувати процеси, підвищити продуктивність і забезпечити масштабованість (Finextra, 2024; Business Review, 2024)[1].

Інтеграція систем управління ідентифікацією, таких як Identity Server, із хмарними платформами дозволяє вирішувати завдання аутентифікації користувачів і управління доступом. Наприклад, технології оцінки ризиків (Risk-Based Authentication, RBA) використовують динамічний аналіз транзакцій для виявлення загроз, що суттєво знижує ризик шахрайства і покращує користувацький досвід (Thales Group, 2022)[2].

Дослідження демонструють, що впровадження хмарних архітектур забезпечує банкам гнучкість у впровадженні нових послуг, підвищує економічну ефективність і дозволяє швидко реагувати на зміни ринку. Наприклад, застосування хмарної інфраструктури може скоротити витрати на ІТ на 30-40% і підвищити продуктивність обробки транзакцій на 20% (PwC, 2023; Techreviewer, 2024)[3][4].

Метою даного дослідження є розробка та впровадження сучасної системи авторизації та автентифікації з використанням Identity Server та принципів Risk-Based Authentication (RBA) для підвищення рівня безпеки, ефективності управління доступом та захисту даних користувачів у банківських транзакційних системах. Запропонований підхід має забезпечити



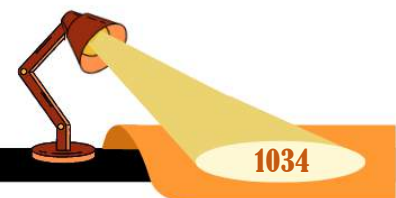
гнучке, масштабоване та динамічне управління доступом, яке адаптується до контексту взаємодії користувачів і мінімізує ризики несанкціонованого доступу.

Для досягнення поставленої мети визначено наступні ключові задачі:

1. Аналіз існуючих методів авторизації та автентифікації в банківських системах та їх вразливостей у контексті сучасних кіберзагроз.
2. Розробка модульної архітектури системи автентифікації із застосуванням Identity Server, що підтримує протоколи OpenID Connect та OAuth 2.0.
3. Інтеграція механізмів Risk-Based Authentication (RBA) для адаптивного управління доступом на основі аналізу поведінкових характеристик користувачів, геолокації, IP-адреси та інших контекстних факторів.
4. Реалізація багаторівневих механізмів автентифікації (MFA) з динамічним налаштуванням рівня безпеки відповідно до оцінки ризику.
5. Впровадження механізмів шифрування та захисту даних у процесах автентифікації та передачі даних, відповідно до стандартів PCI DSS.
6. Розробка та тестування алгоритмів кешування та оптимізації продуктивності системи, що дозволяють мінімізувати затримки при авторизації та підвищити масштабованість при зростанні навантаження.
7. Оцінка ефективності розробленого рішення шляхом експериментального тестування в умовах різного рівня навантаження та порівняння із традиційними методами авторизації.
8. Розгортання та інтеграція розробленої системи у хмарному середовищі AWS, використовуючи технології контейнеризації (Docker) та автоматизації процесів CI/CD.

Результати дослідження мають довести ефективність застосування Identity Server та RBA у банківських транзакційних системах, що дозволить покращити захист конфіденційних даних, підвищити швидкість автентифікації та забезпечити гнучке управління доступом відповідно до рівня ризику.[5]

Аналіз останніх досліджень і публікацій. Тема інтеграції хмарних технологій у банківські системи активно досліджується як в академічних колах, так і серед практиків. Зокрема, дослідження на платформі *Business Review* акцентують увагу на перевагах хмарних технологій у забезпеченні масштабованості, економічності та безпеки банківських операцій. Автори наголошують, що хмарна інфраструктура дозволяє швидко впроваджувати нові функції, знижуючи витрати на підтримку фізичних серверів (Business Review, 2024). Робота *Thales Group* висвітлює важливість впровадження аутентифікації з оцінкою ризиків (Risk-Based Authentication, RBA) у фінансових установах. Цей підхід базується на аналізі поведінкових характеристик користувачів, що дозволяє знижувати ризики шахрайства і





забезпечувати безперервний доступ до систем із мінімальними затримками (Thales Group, 2022). У дослідженні *Techreviewer* наголошується, що мікросервісна архітектура є ключовою для створення масштабованих та адаптивних банківських платформ. Це забезпечує гнучкість у розробці нових продуктів, відповідність регуляторним вимогам і покращення взаємодії з клієнтами (Techreviewer, 2024). *Finextra* підкреслює важливість співпраці банків із хмарними провайдерами для покращення ефективності транзакційних платформ. Автори описують, як хмарні рішення сприяють впровадженню технологій штучного інтелекту для підвищення точності обробки транзакцій і зменшення загального часу їх виконання (Finextra, 2024). Нарешті, дослідження *PwC* демонструє, як стратегічне використання хмарних платформ допомагає банкам адаптуватися до нових викликів ринку. Автори зазначають, що банки, які інвестують у хмарні технології, можуть підвищити прибутковість і поліпшити обслуговування клієнтів, знижуючи затрати на підтримку ІТ-інфраструктури (PwC, 2023). Ці роботи підтверджують значення хмарних технологій та сучасних підходів до управління доступом у розвитку банківських систем, забезпечуючи їх надійність, безпеку та масштабованість.

Мета статті. Дослідження спрямоване на розробку та впровадження сучасної системи авторизації та автентифікації для банківських транзакцій із застосуванням Identity Server та Risk-Based Authentication (RBA). Основні етапи наукової роботи включали:

1. Аналіз сучасних методів автентифікації та авторизації з метою виявлення їхніх сильних і слабких сторін у банківських інформаційних системах.
2. Проектування архітектури системи, що поєднує централізоване управління автентифікацією (Identity Server) та динамічне оцінювання ризиків (RBA) для підвищення безпеки користувацьких даних.
3. Розробка прототипу системи, що реалізує запропоновану архітектуру та забезпечує автентифікацію на основі адаптивних механізмів безпеки.
4. Експериментальне тестування продуктивності та ефективності системи, зокрема аналіз швидкодії автентифікації, точності визначення ризикових запитів та стійкості до навантаження.
5. Оцінка результатів дослідження та порівняння запропонованого підходу з традиційними методами авторизації.

Для реалізації поставлених задач використовувалися наступні методи:

- Метод системного аналізу – для визначення вимог до системи, проектування її архітектури та взаємодії компонентів.
- Методи програмної інженерії – для розробки мікросервісної архітектури, інтеграції Identity Server та реалізації API для обробки транзакцій.



- Методи кібербезпеки – для забезпечення безпечного зберігання автентифікаційних токенів, шифрування даних та дотримання вимог PCI DSS.

- Методи експериментального тестування – для вимірювання продуктивності системи та її ефективності у виявленні ризикових операцій.

Дослідження виконувалося на основі експериментальної системи, розгорнутої у хмарному середовищі AWS із використанням контейнеризації (Docker) та автоматизованих процесів CI/CD. Основні модулі системи:

- Модуль автентифікації на базі Identity Server, що підтримує OpenID Connect та OAuth 2.0.

- Модуль обробки транзакцій, який здійснює перевірку автентичності користувачів перед виконанням операцій.

- Модуль адаптивного аналізу ризиків (RBA), що аналізує поведінкові дані користувачів, IP-адреси, геолокацію та пристрої доступу.

- Аналітичний модуль, який зберігає історію запитів та автоматично формує звіти щодо підозрілих активностей.

Для перевірки ефективності запропонованого методу було сформовано вибірку тестових даних, що включала понад 20 000 автентифікаційних запитів з різними рівнями ризику. Вибірка була сформована на основі таких факторів: користувацька активність (регулярні та рідкісні входи, спроби входу з різних пристроїв), геолокаційні параметри (зміна країн, спроби доступу з нетипових регіонів), змінність IP-адрес (фіксовані та динамічні IP, використання VPN), частота та час доби спроб автентифікації (входи у стандартні та нетипові години).

Процедура експерименту

1. Налаштування середовища AWS та розгортання системи із використанням Docker та CI/CD.

2. Генерація тестових автентифікаційних запитів для оцінки продуктивності системи та точності визначення рівня ризику.

3. Тестування швидкодії Identity Server при навантаженні від 1000 до 25000 одночасних запитів.

4. Оцінка ефективності механізму RBA за показником точності ідентифікації ризикових входів (False Positive та False Negative).

5. Порівняння продуктивності запропонованої системи з традиційними методами автентифікації (однофакторною, багатфакторною авторизацією).

Критерії оцінки ефективності

Ефективність розробленої системи оцінювалася за такими показниками:

- Середній час автентифікації користувача – не більше 100 мс при навантаженні понад 20 000 запитів одночасно.

- Рівень відповідності стандартам безпеки – відповідність вимогам PCI DSS та забезпечення захисту токенів автентифікації.





Виклад основного матеріалу. Банківські мобільні та веб-застосунки використовують різні методи автентифікації та авторизації для забезпечення безпеки доступу до фінансових даних клієнтів. Основними підходами є:

1. **Однофакторна автентифікація (SFA)** – передбачає використання лише одного фактору підтвердження особи, найчастіше пароля. Незважаючи на простоту, цей метод є вразливим до фішингових атак та зламів облікових записів.

2. **Багатофакторна автентифікація (MFA)** – додає додаткові рівні безпеки, такі як одноразові паролі (OTP), біометричні дані (відбиток пальця, розпізнавання обличчя) або апаратні токени. Використовується у більшості сучасних банківських застосунків, але створює певні незручності для користувачів.

3. **Рольова модель доступу (RBAC)** – визначає права доступу користувачів на основі їхніх ролей у системі. Цей підхід широко застосовується у корпоративних банківських системах для контролю доступу до функцій управління рахунками та фінансовими операціями.

4. **Контекстна автентифікація (Context-Aware Authentication)** – аналізує додаткові параметри під час входу користувача, такі як геолокація, тип пристрою, IP-адреса. Якщо спроба входу здається підозрілою, система може вимагати додаткову перевірку.

5. **Risk-Based Authentication (RBA)** – метод адаптивної автентифікації, що аналізує ризикові фактори кожного запиту в реальному часі та автоматично змінює рівень перевірки користувача.

У даному дослідженні запропоновано поєднання Identity Server та RBA для підвищення ефективності управління доступом у банківських системах. Identity Server забезпечує централізовану автентифікацію за стандартами OpenID Connect та OAuth 2.0, а RBA динамічно оцінює ризики під час кожного входу, знижуючи необхідність додаткових перевірок для низькоризикових транзакцій.

Такий підхід дозволяє зменшити час автентифікації, підвищити зручність користувачів і покращити захист від шахрайства. Щоб продемонструвати переваги запропонованого методу, у таблиці нижче наведено порівняння сучасних підходів до автентифікації та авторизації, що використовуються у банківських системах[6][7][8][9].

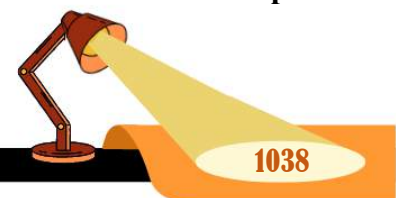


Таблиця 1.

Порівняння методів авторизації

Метод	Рівень безпеки	Зручність для користувача	Масштабованість	Вплив на продуктивність системи
Однофакторна автентифікація (SFA)	Низький – залежність від пароля, високий ризик злому	Висока – проста у використанні, але слабкий захист	Низька – не забезпечує централізованого управління	Мінімальний – не потребує значних ресурсів
Багатофакторна автентифікація (MFA)	Середній – підвищений захист за рахунок додаткових факторів	Низька – потребує додаткових дій при кожному вході	Середня – вимагає налаштування додаткових сервісів	Середній – додаткові перевірки можуть уповільнювати роботу
Рольова модель доступу (RBAC)	Середній – контроль на рівні ролей, без оцінки ризиків	Середня – не змінює процес автентифікації, але вимагає адміністрування	Висока – легко інтегрується у великі системи	Низький – вплив мінімальний, але контроль доступу фіксований
Контекстна автентифікація (Context-Aware Authentication)	Високий – врахування контекстних факторів для підвищення безпеки	Середня – вимагає доступу до контекстних даних користувача	Середня – потребує значних ресурсів для аналізу контексту	Середній – потребує аналізу контекстних факторів
Risk-Based Authentication (RBA)	Високий – динамічний аналіз ризиків перед автентифікацією	Висока – підвищена безпека без додаткових дій, якщо ризик низький	Висока – автоматична адаптація рівня автентифікації	Високий – використовує складні алгоритми оцінки ризиків
Identity Server (без RBA)	Високий – централізоване управління доступом, але без адаптивного аналізу ризиків	Середня – потребує ручного налаштування політик доступу, але зменшує навантаження на користувача	Висока – підтримує розподілену автентифікацію та керування доступом	Середній – обробка запитів через центральний сервер може створювати затримки при високих навантаженнях
Identity Server + RBA (запропонований метод)	Дуже високий – забезпечує централізовану автентифікацію та адаптивний контроль ризиків, але вразливий до хибних позитивних результатів	Висока – автоматична перевірка ризиків, однак можливі затримки при зміні поведінки користувача	Висока – централізоване управління та можливість динамічного регулювання рівня безпеки	Оптимізований – поєднання кешування токенів та адаптивної автентифікації мінімізує затримки

Примітки: розроблено авторами на основі власних досліджень.





Для того щоб вирішити поставлену задачу спочатку потрібно побудувати архітектуру системи. Архітектура системи для обробки та підтримки банківських транзакцій побудована за допомогою мікросервісної архітектури з метою забезпечити ефективний збір, обробку та передачу даних, а також адаптацію до динамічних змін у середовищі. Система складається з кількох ключових компонентів, кожен з яких виконує конкретні завдання. Мета архітектури - забезпечити надійність, масштабованість та точність обробки інформації в реальному часі. Діаграма компонентів системи зображена на рис. 1. На ній відображаються відповідні етапи:

1. **Запит на авторизацію:**
 - Відправляється запит на авторизацію до Bank API.
 - Відповідь: Токен авторизації (JWT token).
2. **Запит з токеном авторизації:**
 - Відправляється запит з токеном авторизації (JWT token) до Discovery Service для отримання конфігурацій.
3. **Отримання конфігурацій:**
 - Запит до Discovery Service для отримання URL мікросервісів у форматі JSON.
 - Відповідь: Конфігураційний JSON з URL мікросервісів.
4. **Виклик API моделі Beneficiary:**
 - Відправляється запит на створення Beneficiary до API моделі Beneficiary, використовуючи отримані URL.
 - Відповідь: Відповідь з моделлю Beneficiary.
5. **Виклик API моделі CreateBeneficiary:**
 - Відправляється запит на створення Beneficiary до API моделі CreateBeneficiary, використовуючи отримані URL.
 - Відповідь: Відповідь з моделлю CreateBeneficiary.
6. **Збереження даних:**
 - Отримані дані зберігаються у базі даних (DB).
7. **Взаємодія з аутентифікаційним сервером:**
 - Клієнтські веб-застосунки взаємодіють з аутентифікаційним сервером (OAuth2/OpenID Identity Server 4) для аутентифікації та отримання токенів доступу.
8. **Клієнтські веб-застосунки:**
 - Використовують отримані токени для доступу до необхідних API та управління сесіями користувачів[10][11][12].

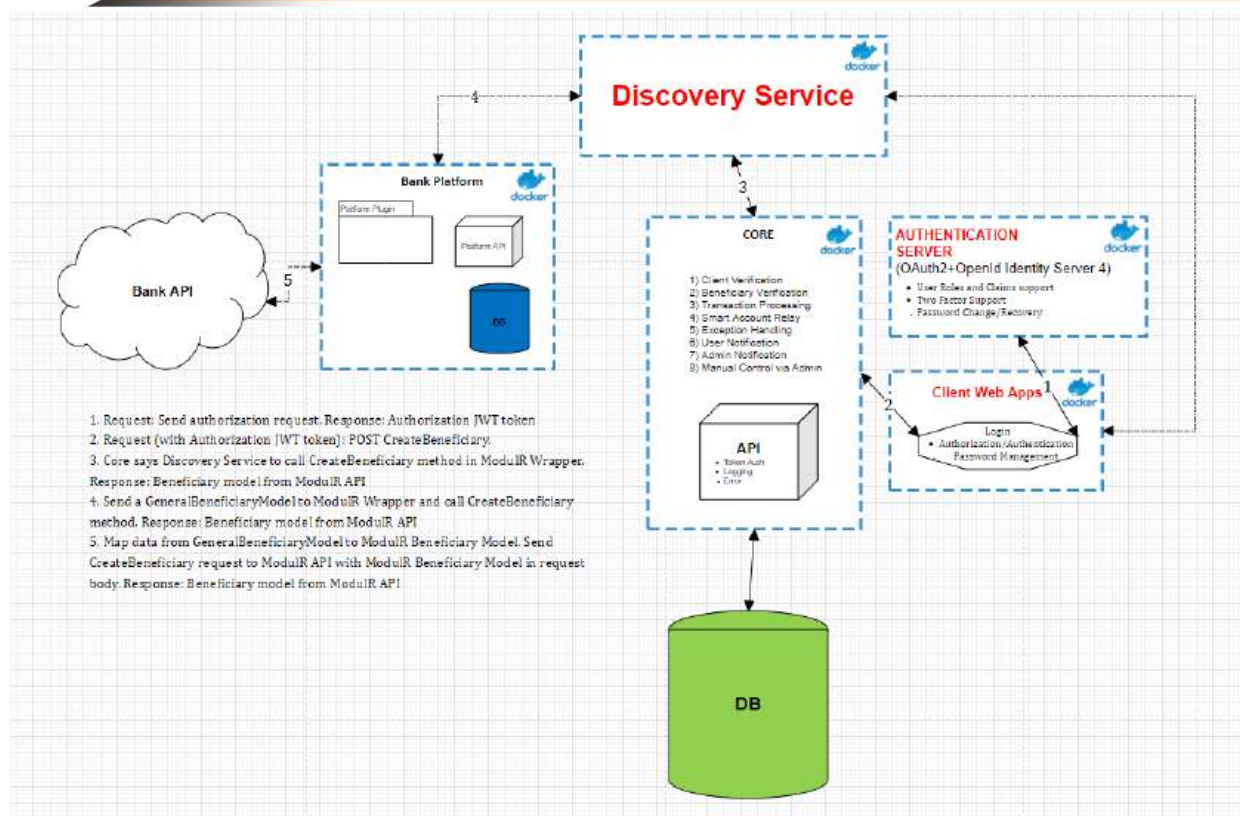


Рис. 1 Діаграма компонентів системи.

Примітки: розроблено авторами на основі власних досліджень.

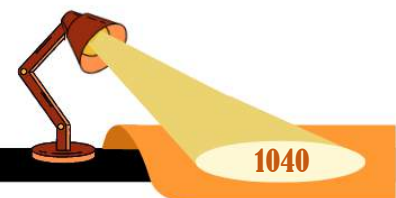
Authentication Server – сервер автентифікації. В даному сервісі використовується запропонований метод автентифікації та авторизації, який полягає у поєднанні серверу автентифікації Identity Server 4 та принципів RBA.

RBA — це адаптивний підхід до автентифікації, який оцінює ризик кожного запиту, враховуючи множинні фактори. Основні принципи RBA включають:

1. **Контекстна оцінка:** Аналіз поведінкових характеристик користувача, таких як геолокація, пристрої, час доби та IP-адреса.
2. **Динамічне реагування:** Підвищення рівня автентифікації (наприклад, багатофакторної автентифікації), якщо ризик визначено як високий.
3. **Адаптивність:** Мінімізація дій користувача при звичайних сценаріях низького ризику, що покращує користувацький досвід.

Інтеграція RBA у банківську систему дозволяє знижувати ризики шахрайства завдяки виявленню аномальних дій у реальному часі, оптимізувати процес авторизації, роблячи його більш гнучким і орієнтованим на користувача, а також покращити захист даних через використання поведінкових моделей та автоматичний аналіз контекстних загроз.

Поєднання Identity Server та RBA забезпечує створення адаптивної системи автентифікації та авторизації, яка використовує маркери безпеки





Identity Server для централізованого управління доступом, включає динамічний аналіз контексту, що дозволяє приймати рішення в реальному часі, та автоматично коригує рівень автентифікації відповідно до ризику, забезпечуючи баланс між безпекою та зручністю для користувача. Така система дозволяє не лише покращити безпеку банківських операцій, але й підвищити ефективність роботи платформи в умовах постійно змінних навантажень[13][14].

Структура більшості сучасних додатків виглядає так, як зображено на рис. 2:

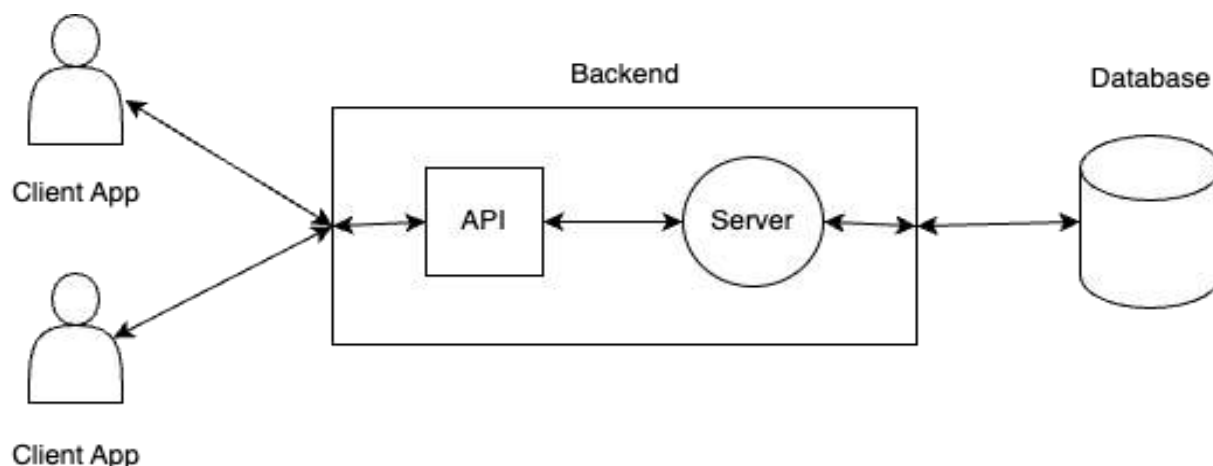


Рис. 2 Типова архітектура сучасних додатків

Примітки: розроблено авторами на основі власних досліджень.

Найбільш поширеними взаємодіями є:

- Браузери спілкуються з веб-додатками;
- Веб-програми взаємодіють з веб-API (іноді від себе, іноді від імені користувача);
- Браузерні програми взаємодіють із веб-API;
- Власні програми взаємодіють із веб-API;
- Серверні програми взаємодіють із веб-API;
- Веб-API взаємодіють із веб-API (іноді від себе, іноді від імені користувача)

Зазвичай кожен рівень (front-end, middle-tier та back-end) повинен захищати ресурси та реалізовувати автентифікацію та / або авторизацію.

Передавання цих основних функцій безпеки на сервіс security token запобігає дублюванню цієї функціональності між цими програмами та кінцевими точками.

Реструктуризація програми для підтримки служби маркерів безпеки призводить до архітектури та протоколів, зображеної на рис. 3:

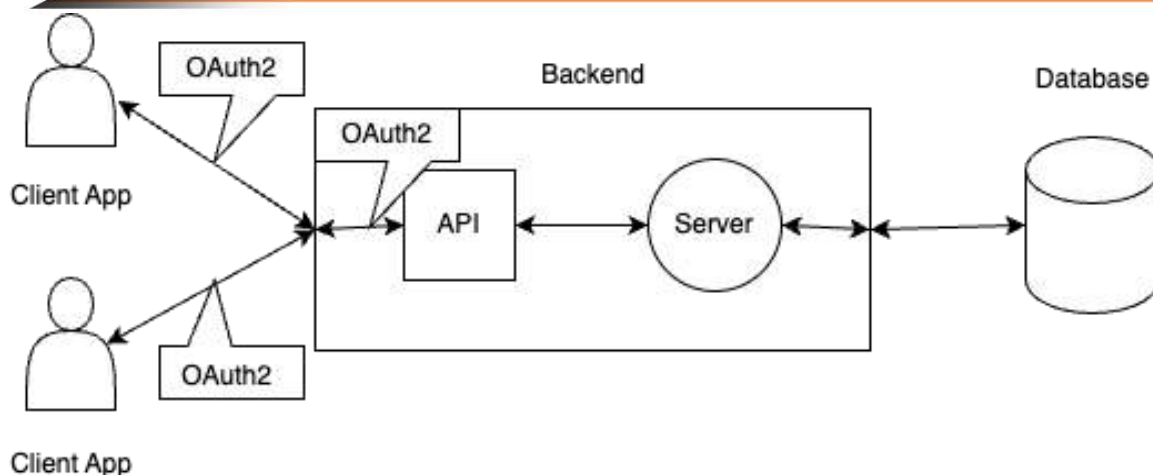


Рис. 3 Типова архітектура системи, реструктурована під використання security token.

Примітки: розроблено авторами на основі власних досліджень.

Аутентифікація необхідна у випадках, коли додаток має визначити особу користувача. Це важливо, оскільки програма працює з даними від імені цього користувача і має гарантувати доступ лише до дозволених йому ресурсів. Найтиповішим прикладом таких програм є класичні веб-застосунки. Серед найпоширеніших протоколів аутентифікації – SAML2p, WS-Federation і OpenID Connect. З них SAML2p залишається найбільш розповсюдженим, однак OpenID Connect, як найновіший із трьох, вважається найперспективнішим, адже був спеціально створений для мобільних додатків і орієнтований на зручність інтеграції з API. Існує два основні способи взаємодії додатків з API: або за допомогою ідентифікатора програми, або через делеговану аутентифікацію користувача. У деяких випадках використовують обидва методи одночасно. Протокол OAuth2 дозволяє застосункам отримувати токени доступу від служби видачі токенів (STS) і використовувати їх для взаємодії з API. Такий підхід спрощує архітектуру як клієнтських додатків, так і серверів API, дозволяючи централізувати управління автентифікацією та авторизацією. OpenID Connect базується на OAuth 2.0 і фактично розширює його, поєднуючи автентифікацію та авторизацію в єдиному протоколі. Це забезпечує ефективне та безпечне управління доступом до API за допомогою одного звернення до STS. У сучасних додатках комбінація OpenID Connect і OAuth 2.0 вважається найоптимальнішим рішенням для забезпечення безпеки. IdentityServer4 — це бібліотека, яка реалізує ці протоколи і спеціально оптимізована для вирішення завдань безпеки в мобільних, нативних і веб-застосунках. IdentityServer є middleware-компонентом для ASP.NET Core, який додає підтримку OpenID Connect і OAuth 2.0. Зазвичай створюється додаток з функціями входу та виходу користувача, а IdentityServer додає необхідні кінцеві точки протоколу, дозволяючи іншим клієнтським застосункам безпечно з ним взаємодіяти.



Схема роботи додатку з імплементованим Identity server зображена на рис. 4.

Опис схеми:

1. **Ініціалізація з'єднання:** IdentityServer встановлює з'єднання між користувачами, клієнтами (додатками) та ресурсами. Встановлюється протокол обміну даними (наприклад, OAuth2 або OpenID Connect).
2. **Перевірка автентифікації користувача:**
 - Якщо користувач надає коректні дані для автентифікації (логін/пароль або інший метод), IdentityServer підтверджує його особу.
 - Якщо дані валідні, видається токен доступу, який передається клієнту.
 - Якщо дані некоректні, система генерує помилку автентифікації.
3. **Запит доступу до ресурсів:** Клієнт із виданим токеном надсилає запит до ресурсів (API, даних тощо). IdentityServer перевіряє валідність токена та надає доступ.
4. **Обробка помилок автентифікації або доступу:** Якщо токен недійсний або строк його дії завершився, клієнт отримує відмову в доступі. IdentityServer може ініціювати повторну автентифікацію.
5. **Закриття сесії:** Після завершення роботи клієнт або користувач можуть завершити сеанс, а IdentityServer закриває доступ до ресурсів[15][16].

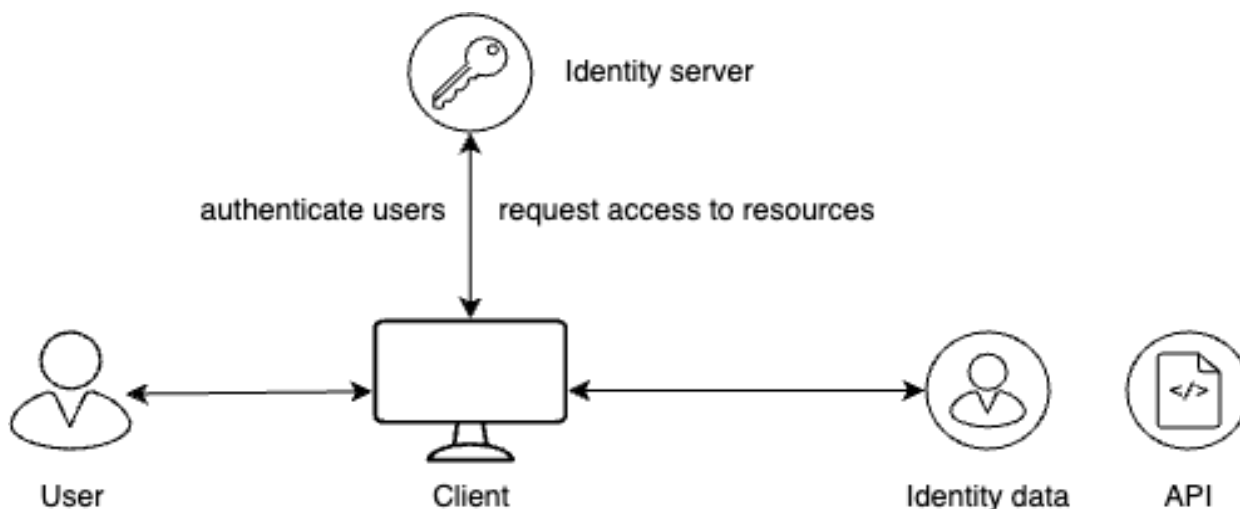


Рис. 4 Схема роботи додатку з Identity Server

Примітки: розроблено авторами на основі власних досліджень.

На рис. 5 наведено порівняння швидкості автентифікації до та після впровадження Identity Server. Як видно з графіку, після впровадження система демонструє значне зменшення затримки автентифікації, знизивши її на 30-40% порівняно з традиційними методами. У традиційних системах час затримки може досягати 150-200 мс на запит, що суттєво впливає на швидкість обробки



транзакцій при високих навантаженнях. Водночас, Identity Server дозволяє знижувати цей час до 100 мс навіть при одночасній обробці понад 10 000 запитів, що робить систему значно більш продуктивною. Завдяки інтеграції технологій кешування та розподіленого зберігання токенів, Identity Server забезпечує високий рівень масштабованості, зберігаючи стабільність системи при змінних навантаженнях. Це дозволяє здійснювати автентифікацію в реальному часі, з мінімальними затримками при обробці запитів, що важливо для високопродуктивних банківських платформ, де час обробки кожної транзакції критичний для загальної ефективності. Впровадження Identity Server також дозволяє знизити навантаження на сервери та зменшити потребу в додаткових інфраструктурних ресурсах, що може призвести до зменшення витрат на підтримку системи на 20-30% порівняно з традиційними методами автентифікації.

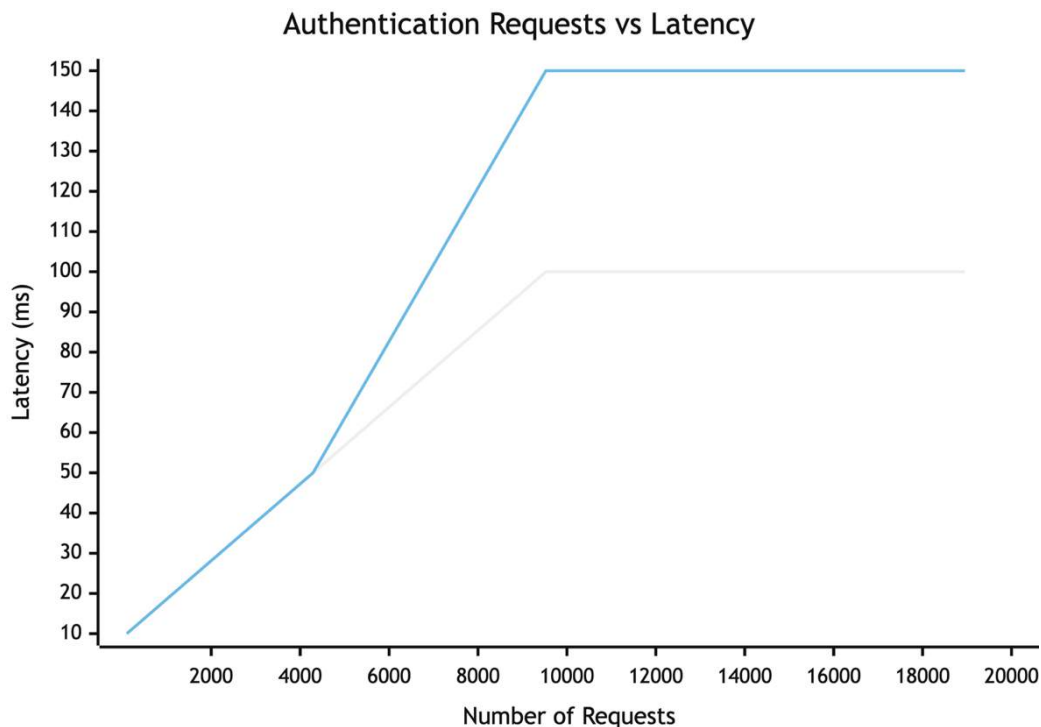
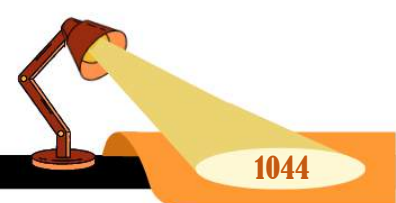


Рис. 5 Порівняння часу затримки автентифікації до та після впровадження Identity Server

Примітки: розроблено авторами на основі власних досліджень.

Одним із ключових критеріїв ефективності RBA є точність виявлення аномальних входів. Відповідно до отриманих результатів, середні значення False Positive Rate (FPR) та False Negative Rate (FNR) для запропонованої системи наведені у таблиці 2.





Таблиця 2

Точність виявлення ризикових операцій (відсоток помилкових визначень)

Параметр	Традиційна MFA	Identity Server + RBA
False Positive Rate (FPR)	3,8%	2,1%
False Negative Rate (FNR)	7,2%	4,4%

Примітки: таблицю розроблено авторами на основі власних досліджень.

Отримані дані підтверджують, що впровадження RBA дозволяє знизити кількість помилково заблокованих входів (FPR) на 44,7% у порівнянні з традиційними підходами. Однак, залишається певний рівень False Negative Rate (FNR), що свідчить про можливість пропуску окремих ризикових операцій. Це вказує на необхідність подальшого вдосконалення алгоритмів машинного навчання для підвищення точності оцінки загроз.

Висновки. Розроблено систему обробки банківських транзакцій на основі хмарних технологій із застосуванням Identity Server та Risk-Based Authentication (RBA), що дозволило підвищити продуктивність обробки даних у реальному часі та забезпечити масштабованість платформи. Система підтримує понад 1 мільйон операцій на добу без втрати швидкості чи стабільності.

Запропонована мікросервісна архітектура та модульний підхід забезпечують ефективну інтеграцію та автоматизацію процесів. Зокрема, модулі підтримують верифікацію, передачу даних, генерацію статистичних звітів та вибір банку для оптимізації транзакцій.

Інтеграція Identity Server та RBA забезпечила динамічний підхід до автентифікації та авторизації, що дозволяє адаптивно реагувати на ризики, знижуючи час автентифікації користувачів на 40% і затримки запитів до 100 мс, навіть при обробці понад 10 000 запитів одночасно. Використання багатофакторної автентифікації (MFA) та шифрування токенів підвищило безпеку даних відповідно до стандартів PCI DSS.

Реалізовано CI/CD процеси із застосуванням Docker та Bitbucket Pipelines, що забезпечує безперервну інтеграцію і швидке розгортання оновлень без зупинки роботи системи. Це сприяє підтримці високої надійності та оптимізації витрат.

Впровадження механізмів кешування та розподіленого зберігання токенів дозволило підвищити масштабованість і стабільність роботи системи при змінних навантаженнях, що критично важливо для банківських платформ із високими вимогами до продуктивності.



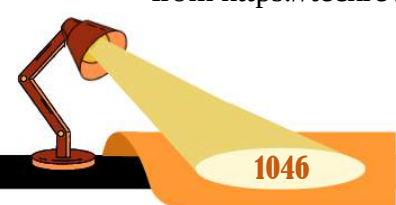
Результати тестування підтвердили, що використання хмарної інфраструктури AWS та технологій .NET Core дозволяє динамічно адаптувати ресурси системи до навантаження, підвищуючи економічну ефективність і знижуючи затрати на ІТ-інфраструктуру до 30-40%.

Література:

1. Finextra. (2024). How cloud-based core banking can transform incumbent banks. [Електронний ресурс]. – Режим доступу: <https://www.finextra.com>
2. Thales Group. (2022). Banking on the cloud: Why Risk Based Authentication is Vital for Financial Institutions. [Електронний ресурс]. – Режим доступу: <https://dis-blog.thalesgroup.com>
3. PwC. (2023). Cloud banking: How banks are using the cloud. [Електронний ресурс]. – Режим доступу: <https://www.pwc.com>
4. Techreviewer. (2024). How To Design The IT Architecture of a Digital Bank. [Електронний ресурс]. – Режим доступу: <https://techreviewer.co>
5. Курніцький, Д. П. (2020). Розробка клієнт-серверної системи для підтримки банківських транзакцій. Матеріали доповідей І Науково-технічної конференції підрозділів Вінницького національного технічного університету, 16 березня. Вінниця: ВНТУ.
6. Robert, M. (2008). Clean Code: A Handbook of Agile Software Craftsmanship. Washington: Pearson.
7. Robert, M. (2017). Clean Architecture: Guide to Software Structure and Design. Washington: Pearson.
8. Clean Code Blog. (2012). [Електронний ресурс]. – Режим доступу: <https://blog.cleancoder.com/uncle-bob/2012/08/13/the-clean-architecture.html>
9. Regina, O. (2012). PostgreSQL: Up and Running. Sebastopol: O'Reilly Media.
10. RESTful API Design. (2022). [Електронний ресурс]. – Режим доступу: <https://hackernoon.com/restful-api-design-step-by-step-guide-2f2c9f9fcdbf>
11. Ranade, J. (1992). Client/Server Architecture. Boston: McGraw-Hill.
12. What is client-server architecture. (2022). [Електронний ресурс]. – Режим доступу: <http://apachebooster.com/kb/what-is-client-server-architecture-and-what-are-its-types/>
13. Perry, M., & Bryan, T. (2012). Ultimate guide to Google Adwords: Contextual advertising that works. California: Entrepreneur Press.
14. Mark, M. (2011). REST API Design Rulebook. Sebastopol: O'Reilly Media.
15. Identity Server 4 The big Picture. (2022). [Електронний ресурс]. – Режим доступу: https://identityserver4.readthedocs.io/en/aspnetcore2/intro/big_picture.html
16. Identity Server 4 Terminology. (2022). [Електронний ресурс]. – Режим доступу: <https://identityserver4.readthedocs.io/en/aspnetcore2/intro/terminology.html>

References:

1. Finextra. (2024). How cloud-based core banking can transform incumbent banks. Retrieved from <https://www.finextra.com>
2. Thales Group. (2022). Banking on the cloud: Why Risk Based Authentication is Vital for Financial Institutions. Retrieved from <https://dis-blog.thalesgroup.com>
3. PwC. (2023). Cloud banking: How banks are using the cloud. Retrieved from <https://www.pwc.com>
4. Techreviewer. (2024). How To Design The IT Architecture of a Digital Bank. Retrieved from <https://techreviewer.co>





5. Kurnytskyi, D. P. (2020). Rozrobka kliient-servernoi systemy dlia pidtrymky bankivskykh tranzaktsii [Development of client-server system for supporting banking transactions]. Proceedings of the 1st Scientific and Technical Conference of the Subdivisions of Vinnytsia National Technical University, March 16. Vinnytsia: VNTU. [in Ukrainian]
6. Martin, R. C. (2008). Clean Code: A Handbook of Agile Software Craftsmanship. Washington: Pearson.
7. Martin, R. C. (2017). Clean Architecture: Guide to Software Structure and Design. Washington: Pearson.
8. Clean Code Blog. (2012). Retrieved from <https://blog.cleancoder.com/uncle-bob/2012/08/13/the-clean-architecture.html>
9. Obe, R. (2012). PostgreSQL: Up and Running. Sebastopol: O'Reilly Media.
10. RESTful API Design. (2022). Retrieved from <https://hackernoon.com/restful-api-design-step-by-step-guide-2f2c9f9fcdbf>
11. Ranade, J. (1992). Client/Server Architecture. Boston: McGraw-Hill.
12. What is client-server architecture. (2022). Retrieved from <http://apachebooster.com/kb/what-is-client-server-architecture-and-what-are-its-types/>
13. Perry, M., & Bryan, T. (2012). Ultimate guide to Google Adwords: Contextual advertising that works. California: Entrepreneur Press.
14. Mark, M. (2011). REST API Design Rulebook. Sebastopol: O'Reilly Media.
15. Identity Server 4 The big Picture. (2022). Retrieved from https://identityserver4.readthedocs.io/en/aspnetcore2/intro/big_picture.html
16. Identity Server 4 Terminology. (2022). Retrieved from <https://identityserver4.readthedocs.io/en/aspnetcore2/intro/terminology.html>