DOI: https://doi.org/10.36910/6775-2524-0560-2026-64-34
УДК 004.056.5
Чухрій Максим Сергійович, аспірант
https://orcid.org/0009-0009-1407-0302
Луцький національний технічний університет, м. Луцьк, Україна

МЕХАНІЗМИ ДИНАМІЧНОГО КОНТРОЛЮ ДОСТУПУ В БЕЗСЕРВЕРНИХ СЕРЕДОВИЩАХ ДЛЯ ПІДВИЩЕННЯ БЕЗПЕКИ

Чухрій М. С. Механізми динамічного контролю доступу в безсерверних середовищах для підвищення безпеки. У сучасному цифровому середовищі безсерверні обчислення стали ключовим інструментом для масштабованої обробки даних, проте вони створюють специфічні виклики для безпеки через свою ефемерну та динамічну природу. Традиційні моделі контролю доступу, що базуються на статичних ролях, часто виявляються недостатніми для захисту короткоживучих функцій, що взаємодіють із конфіденційними ресурсами. У статті проведено огляд сучасних механізмів динамічного контролю доступу, включно з атрибутивним керуванням та концепцією Policy as Code. Проаналізовано їх призначення, переваги та обмеження з позиції практичного застосування в хмарних інфраструктурах, таких як AWS, Google Cloud та Azure. Окрему увагу приділено адаптивним політикам, що базуються на аналізі контексту реального часу та оцінці ризиків. Додатково у роботі описано архітектурні особливості безсерверних середовищ, на яких виникає потреба у впровадженні дрібнозернистого захисту, а також розглянуто роль автоматизації комплаєнсу. У межах дослідження розглянуто наукові публікації щодо безпеки мікросервісів, ефективності адаптивних моделей безпеки та концепції Zero Trust у хмарних обчисленнях. Проведений аналіз може слугувати орієнтиром для хмарних архітекторів та спеціалістів із кібербезпеки під час проектування захищених систем, що потребують високого рівня ізоляції та гнучкості. Запропоновані підходи дозволяють мінімізувати поверхню атаки й можуть бути корисними у розвитку стратегій захисту сучасних хмарних застосунків.
Ключові слова: безсерверні обчислення, динамічний контроль доступу, Policy as Code, хмарна безпека, Zero Trust, адаптивні політики, захист даних у хмарі.

Chukhrii M. Dynamic Access Control Mechanisms in Serverless Environments for Enhanced Security. In the modern digital landscape, serverless computing has become a key tool for scalable data processing; however, it poses specific security challenges due to its ephemeral and dynamic nature. Traditional access control models based on static roles often prove insufficient for protecting short lived functions interacting with confidential resources. This article reviews modern dynamic access control mechanisms, including attribute based control and the Policy as Code concept. Their purpose, advantages, and limitations are analyzed regarding practical application in cloud infrastructures such as AWS, Google Cloud, and Azure. Particular attention is paid to adaptive policies based on real time context analysis and risk assessment. Additionally, the paper describes the architectural features of serverless environments that necessitate the implementation of fine grained security, and considers the role of automated compliance. The study examines scientific publications regarding microservices security, the efficiency of adaptive security models, and the Zero Trust concept in cloud computing. The conducted analysis can serve as a guide for cloud architects and cybersecurity specialists when designing secure systems requiring a high level of isolation and flexibility. The proposed approaches allow for minimizing the attack surface and can be useful in developing protection strategies for modern cloud applications.
Keywords: serverless computing, dynamic access control, Policy as Code, cloud security, Zero Trust, adaptive policies, cloud data protection.

Постановка проблеми. Стрімке впровадження безсерверних обчислень суттєво трансформувало підходи до розробки програмного забезпечення, дозволяючи розробникам зосередитися на бізнес логіці без необхідності управління інфраструктурою. Водночас динамічна та ефемерна природа таких середовищ формує нові вектори атак, які не враховуються традиційними моделями захисту периметра. Основна проблема безпеки полягає у невідповідності статичних механізмів контролю доступу швидкоплинному характеру виклику функцій. Класичні підходи, зокрема рольовий контроль доступу, зазвичай передбачають довготривалі дозволи, що суперечить принципу найменших привілеїв. У результаті в безсерверних архітектурах виникає ризик надання надлишкових прав, якими можуть скористатися зловмисники у разі компрометації окремого сервісу.

Додаткові труднощі створює висока складність конфігурації політик доступу в середовищах із великою кількістю незалежних функцій, що підвищує ймовірність помилок у налаштуваннях. Статичні правила авторизації не здатні враховувати контекст виконання запиту в реальному часі, зокрема географічне розташування користувача, тип пристрою або поточний стан системи безпеки. Такі обмеження роблять безсерверні застосунки вразливими до експлуатації помилок авторизації, коли доступ до чутливих ресурсів може бути отриманий через функцію з некоректно визначеними правами. Окрім цього, впровадження додаткових перевірок безпеки ускладнюється проблемою затримок, оскільки будь які затримки можуть суттєво впливати на продуктивність сервісів. Таким

чином, баланс між рівнем захисту та швидкодією стає одним із ключових викликів для сучасних безсерверних платформ.

У сформованих умовах зростає потреба в адаптивних механізмах контролю доступу, здатних реагувати на зміни контексту виконання запитів у реальному часі. Такі підходи мають ґрунтуватися на атрибутивному управлінні доступом та оцінюванні ризиків під час кожного виклику функції. Динамічне коригування дозволів залежно від поточних параметрів взаємодії дає змогу зменшити поверхню атаки та запобігти зловживанню надмірними привілеями. Крім того, інтеграція контекстно орієнтованих політик відкриває можливості для більш гнучкого та точного захисту розподілених систем. Це робить дослідження подібних механізмів важливим напрямом розвитку безпеки безсерверних архітектур.

Таким чином, виникає гостра потреба у розробці та дослідженні адаптивних механізмів контролю доступу. На сьогоднішній день ринок пропонує широкий спектр архітектурних рішень,але проблема полягає у відсутності чітких критеріїв вибору цих інструментів залежно від вимог до затримки та складності системи.

Аналіз актуальних досліджень та публікацій. Для повного розуміння проблеми та підтримки актуальності дослідження необхідно розглянути існуючі наукові роботи у даній темі. Дослідники наголошують, що перехід до мікросервісної архітектури вимагає переосмислення класичних моделей доступу. Основна увага в сучасних працях приділяється обмеженості моделі RBAC, яка через свою статичність не здатна забезпечити дрібнозернистий контроль у хмарі. Науковці пропонують інтеграцію атрибутивного керування, що дозволяє приймати рішення про доступ на основі розширеного набору метаданих, проте зазначають, що складність опису таких політик залишається високим бар'єром для розробників [1].

Важливим етапом розвитку захисту безсерверних рішень стало впровадження парадигми політика як код та використання інструментів автоматизованої перевірки того,чи відповідає інфраструктура встановленим правилам та стандартам безпеки. Зокрема, у дослідженнях Чарльза Куммарапуругу детально аналізується інтеграція динамічного контролю доступу в мультихмарних середовищах [2]. Автор обґрунтовує, що використання Policy as Code дозволяє не лише автоматизувати перевірку прав, а й забезпечити безперервний моніторинг відповідності стандартам безпеки у режимі реального часу. Це мінімізує людський фактор при конфігуруванні складних систем, де кількість функцій може вимірюватися тисячами.

Окремим перспективним напрямом є використання алгоритмів машинного навчання для адаптивної безпеки. Сучасні публікації розглядають перехід від жорстко заданих правил до моделей, що здатні аналізувати поведінку користувача та виявляти аномалії в реальному часі [3]. Такий підхід дозволяє динамічно змінювати рівень доступу у разі виявлення підозрілої активності, наприклад, нетипової локації запиту або аномального обсягу вивантаження даних. Однак дослідники застерігають, що впровадження ML моделей безпосередньо в цикл виконання функцій може призводити до небажаних затримок що вимагає пошуку балансу між рівнем захисту та продуктивністю системи.

Паралельно розвивається напрям токен-орієнтованого контролю доступу на рівні мікросервісів, коли зовнішня автентифікація та внутрішня авторизація розмежовуються між компонентами системи, а кожен мікросервіс самостійно валідує токен доступу без постійного звернення до централізованого сховища облікових даних, що знижує ризик поширення наслідків компрометації одного облікового запису на всю систему [10].

Метою роботи є проведення порівняльного аналізу архітектурних підходів до реалізації контролю доступу в безсерверних середовищах, класифікації наявних інструментів за рівнем інтеграції та визначенні їх впливу на продуктивність системи. Дослідження спрямоване на обґрунтування ефективності моделі Policy as Code та демонстрацію практичної реалізації динамічної авторизації на базі зовнішні модуля прийняття рішень для мінімізації ризиків несанкціонованого доступу.

Виклад основного матеріалу. Перш ніж перейти до прикладу реалізації динамічної авторизації, доцільно проаналізувати існуючі інструментальні засоби, класифікувавши їх за рівнем інтеграції в архітектуру системи. Оскільки безсерверні обчислення накладають жорсткі обмеження на час виконання запиту, критичними факторами при виборі засобу авторизації є не лише функціональна гнучкість, а й вплив на продуктивність системи.

У межах дослідження було виділено та проаналізовано чотири основні підходи до реалізації динамічного контролю доступу:

- Нативні хмарні механізми – базовий рівень захисту інфраструктури.
- Зовнішні модулі прийняття рішень – дозволяє відокремити логіку авторизації.
- Вбудовані бібліотеки – інтегруються безпосередньо в код функції.
- Хмарні сервіси авторизації – системи керування доступом на основі графів зв'язків.

Першим та найбільш розповсюдженим є використання нативних хмарних механізмів. Ці інструменти, що діють на рівні інфраструктури, є обов'язковим фундаментом безпеки, проте вони здебільшого оперують статичними декларативними політиками, що обмежує їхню здатність адаптивно реагувати на зміни контексту запиту в реальному часі. Альтернативою статичним методам є застосування зовнішніх модулів прийняття рішеньою. Такий підхід передбачає повне відокремлення логіки авторизації від бізнес логіки функції. Це дозволяє фахівцям із безпеки централізовано оновлювати правила доступу без необхідності втручання в код мікросервісів та їх повторного розгортання.

Для сценаріїв, де критичним показником є мінімальна затримка обробки запиту, доцільно розглядати вбудовані бібліотеки, що інтегруються безпосередньо в середовище виконання функції та не потребують додаткових мережевих викликів. Натомість, для розподілених систем із надскладними ієрархічними зв'язками між користувачами дедалі частіше використовуються спеціалізовані хмарні сервіси авторизації, які надають можливості моделювання прав доступу на основі графів, хоча й вносять певну затримку через звернення до зовнішніх API. У таблиці 1 наведена характеристика цих інструментів, які можна використовувати для реалізації. В свою чергу, у таблиці 2 вказані переваги та недоліки даних інструментів.

Таблиця 1. Огляд інструментів реалізації контролю доступу в безсерверних архітектурах

Інструмент	Опис	Коли застосовувати	Рівень інтеграції
Нативні хмарні механізми AWS IAM, Azure Entra ID	Вбудована система керування доступом хмарного провайдера. Використовує декларативні JSON політики для визначення дозволів.Є фундаментом безпеки у хмарі [4].	Для базового захисту інфраструктури,обмеженнь доступу функції до баз даних, черг повідомлень або сховищ файлів.	Налаштовується через консоль хмари.
Зовнішній модуль прийняття рішень	Універсальний модуль прийняття рішень з відкритим кодом. Дозволяє винести логіку авторизації з коду програми в окремі файли на мові Rego. Рішення приймається на основі будь яких даних [5; 6].	Для реалізації складної бізнес логіки та відповідності стандартам. Підходить, коли правила часто змінюються і їх потрібно оновлювати без зміну функцій.	Працює як окремий сервіс, що перехоплює запит.
Вбудовані бібліотеки Casbin, Oso	Бібліотека авторизації, що підтримує моделі ACL, RBAC, ABAC. Працює на основі файлів конфігурації. Не є окремим сервісом, а вбудовується безпосередньо в код [7].	Для мікросервісів, де критична швидкість і немає можливості звертатися до зовнішніх серверів авторизації.	Підключається як залежність у Node.js, Python.
Сервіси авторизації Amazon Verified Permissions, Keycloak, Auth0	Керовані сервіси для дрібнозернистої авторизації, що використовують спеціалізовані мови. Дозволяє будувати політики на основі зв'язків та ієрархій [8].	Для фінансових, державних чи критично захищених застосунків, коли потрібен максимальний захист ключів.	API.Функція робить мережевий запит до сервісу для отримання рішення.

Таблиця 2. Переваги та недоліки інструментів

Інструмент	Переваги	Недоліки
Нативні хмарні механізми AWS IAM, Azure Entra ID	Нульова затримка , глибока інтеграція з сервісами, відсутність додаткових витрат.	Статичність правил, складність управління при зростанні кількості функцій, відсутність контексту таких, як час, геолокація тощо.
Зовнішній модуль	Універсальний модуль логіки авторизацій. Логіка вноситься в окремі	Необхідність вивчення мови Rego, незначна мережева затримка при обробці запиту [9].

прийняття рішень	файли, рішення примається на основі будь-яких даних.	
Вбудовані бібліотеки Casbin, Oso	Висока швидкість, підтримка багатьох мов програмування, простота інтеграції.	Децентралізованість, зміна правил вимагає повторного розгортання сервісу.
Сервіси авторизації AVP, Keycloak, Auth0	Можливість моделювання складних зв'язків , масштабованість та вбудований механізм аудиту доступу.	Висока затримка через зовнішні HTTP виклики, прив'язка до постачальника технологій, платна модель використання.

Наведений аналіз інструментів реалізації контролю доступу в безсерверних архітектурах, представлений у таблицях 1 і 2, може слугувати практичним орієнтиром для архітекторів хмарних систем та фахівців із кібербезпеки. Він охоплює як нативні механізми хмарних провайдерів, так і зовнішні рушії політик, вбудовані бібліотеки та керовані сервіси авторизації, демонструючи їхні можливості, переваги та обмеження з позиції практичного використання. Такий огляд дозволяє обґрунтовано обирати підхід до реалізації доступу залежно від вимог до затримки, складності бізнес логіки, частоти змін політик і рівня критичності захищених ресурсів. Узагальнення характеристик також підкреслює компроміс між гнучкістю політик і операційними витратами, що є ключовим чинником при проєктуванні захищених безсерверних застосунків.

На основі проведеного аналізу для практичної апробації обрано архітектурний підхід Policy as Code із використанням зовнішнього модуля прийняття рішень. Цей вибір зумовлений необхідністю забезпечення гнучкості правил безпеки без безпосереднього втручання у вихідний код функцій. Для демонстрації переваг цього методу розроблено модель адаптивного захисту, яка, на відміну від бінарної логіки статичних ролей, оперує кількісною оцінкою загроз у реальному часі. Замість суб'єктивного прийняття рішень адміністратором, система автоматично розраховує сукупний бал ризику для кожного вхідного запиту. Цей розрахунок базується на аналізі метаданих,а саме часу доби, мережевого оточення та стану клієнтського пристрою та інших. На рисунку 1 наведено програмну реалізацію логіки, де кожному потенційно небезпечному фактору присвоюється певна вага, наприклад, доступ у неробочий час збільшує ризик на 30 балів, а використання незнайомої ip адреси на 40. Така структура дозволяє легко додавати нові фактори загрози без зміни основної логіки авторизації.

```
risk_score := score {
    base_risk := 0
    time_risk := 30 if not is_working_hours else 0
    net_risk := 40 if not input.net.is_vpn else 0
    device_risk := 20 if input.device.is_new else 0
    score := base_risk + time_risk + net_risk + device_risk
}
is_working_hours if {
    clock := time.clock(time.now_ns())
    clock[0] >= 9
    clock[0] < 18
}
```

Рисунок 1. Функція алгоритму розрахунку динамічного показника ризику.

Отримане числове значення ризику передається на другий етап,де прийняття рішення про надання доступу. На цьому рівні застосовується диференційований підхід: система не просто блокує підозрілі запити, а вимагає додаткових підтверджень залежно від рівня загрози. На рисунку 2 представлено головний модуль політики безпеки, який інтерпретує розрахований раніше бал. Логіка передбачає три рівні довіри.Низький ризик дозволяє вільний доступ, середній вимагає проходження багатофакторної автентифікації, а критичний рівень призводить до автоматичного блокування, за винятком спеціальних сценаріїв екстреного доступу.

```
allow if {
    input.user.role == "doctor"
    risk_score < 20
}
allow if {
    input.user.role == "doctor"
    risk_score >= 20
    risk_score < 60
    input.auth_context.mfa_verified == true
}
allow if {
    input.request.emergency_flag == true
    input.user.role == "doctor"
}
```

Рисунок 2. Функція політики авторизації на основі рівнів ризику

Запропонована двокомпонентна реалізація демонструє значну перевагу над традиційними статичними моделями. Вона базується на фундаментальному принципі Zero Trust. Навіть якщо користувач пройшов первинну автентифікацію і має валідний токен, це не гарантує йому права на виконання критичної операції. Політика виступає другим, незалежним контуром захисту, який безперервно валідує не лише суб'єкта доступуа й контекст середовища, обставин при яких був зроблений вхід. Це дозволяє системі адаптуватися до нестандартних ситуацій, наприклад, лікар може отримати віддалений доступ вночі, але тільки за умови використання другого фактора автентифікації. Це забезпечує баланс між зручністю використання та високим рівнем безпеки, мінімізуючи ймовірність успішних атак через скомпрометовані облікові записи. Також важливим етапом проектування системи захисту є аналіз впливу механізмів безпеки на загальну продуктивність безсерверних функцій. Впровадження зовнішнього авторизатора неминуче створює додаткове навантаження на мережу. Так, у дослідженні Міллера та співавторів [5], де аналогічна модель зовнішньої авторизації на основі OPA була розгорнута для перевірки політик мікросервісів, засоби захисту в середньому додавали близько двох секунд до часу запуску пода, а варіація тривалості обробки запиту становила від 7 до 65 відсотків залежно від типу з'єднання. Автори підкреслюють, що такий показник є прийнятним з огляду на переваги, які надає відокремлена перевірка політик. Водночас виграш у безпеці, який забезпечується переходом від статичних ролей до динамічного розрахунку ризиків, значно переважає ці часові витрати.

Висновки та перспектива подальших досліджень. У роботі проведено комплексне дослідження проблем захисту безсерверних архітектур та обґрунтовано необхідність переходу від статичних моделей доступу до динамічних адаптивних механізмів. Встановлено, що традиційні засоби, такі як RBAC, не забезпечують достатньої гнучкості в умовах ефемерності хмарних функцій. За результатами порівняльного аналізу інструментальних засобів визначено, що найбільш ефективним інструментом є використання зовнішнього модуля прийняття рішень та продемонстровано, як відокремлення логіки авторизації від бізнес логіки дозволяє централізовано керувати політиками безпеки без необхідності повторного розгортання мікросервісів.

Практична реалізація механізму підтвердила можливість створення адаптивних систем захисту, які автоматично реагують на зміни контексту. Розроблений алгоритм розрахунку динамічного балу ризику дозволяє гнучко балансувати між зручністю користування та рівнем безпеки, надаючи доступ у критичних ситуаціях при одночасному блокуванні підозрілих активностей.

Перспективи подальших досліджень полягають в інтеграції алгоритмів машинного навчання для переходу від статичних вагових коефіцієнтів ризику до динамічних. Замість ручного калібрування балів загроз, використання систем поведінкового аналізу користувачів, які здатні автоматично виявляти аномалії в патернах доступу та коригувати політики безпеки в режимі реального часу. Крім того, важливим напрямком є дослідження застосування генеративного штучного інтелекту для автоматизованого написання та тестування політик, що дозволить мінімізувати людський фактор при конфігурації складних систем захисту.

Список бібліографічного опису

1. Access Control Design Practice and Solutions in Cloud-Native Architecture: A Systematic Mapping Study / M. S. Rahaman та ін. *Sensors*. 2023. Т. 23, № 7. С. 3413. DOI: https://doi.org/10.3390/s23073413.
2. Kummarapurugu C. S. Enhancing Serverless Computing Security in Multi-Cloud Environments: Integrating Policy-as-Code, Automated Compliance, and Dynamic Access Controls. *International Journal of Innovative Research in Engineering & Multidisciplinary Physical Sciences*. 2022. Т. 10, № 2. URL: https://www.researchgate.net/publication/389747418 (дата звернення: 19.08.2026).
3. Abibulaiev A., Pukach P., Vovk M. Context-Aware ML/NLP Pipeline for Real-Time Anomaly Detection and Risk Assessment in Cloud API Traffic. *Machine Learning and Knowledge Extraction*. 2026. Т. 8, № 1. С. 25. DOI: https://doi.org/10.3390/make8010025.
4. Arif T., Jo B., Park J. H. A Comprehensive Survey of Privacy-Enhancing and Trust-Centric Cloud-Native Security Techniques Against Cyber Threats. *Sensors*. 2025. Т. 25, № 8. С. 2350. DOI: https://doi.org/10.3390/s25082350.
5. Securing Workflows Using Microservices and Metagraphs / L. Miller та ін. *Electronics*. 2021. Т. 10, № 24. С. 3087. DOI: https://doi.org/10.3390/electronics10243087.
6. Security in Cloud-Native Services: A Survey / T. Theodoropoulos та ін. *Journal of Cybersecurity and Privacy*. 2023. Т. 3, № 4. С. 758–793. DOI: https://doi.org/10.3390/jcp3040034.
7. Static-Analysis-Based Solutions to Security Challenges in Cloud-Native Systems: Systematic Mapping Study / M. S. Rahaman та ін. *Sensors*. 2023. Т. 23, № 4. С. 1755. DOI: https://doi.org/10.3390/s23041755.
8. A Survey of Context-Aware Access Control Mechanisms for Cloud and Fog Networks: Taxonomy and Open Research Issues / A. S. M. Kayes та ін. *Sensors*. 2020. Т. 20, № 9. С. 2464. DOI: https://doi.org/10.3390/s20092464.
9. Atlam H. F., Yang Y. Enhancing Healthcare Security: A Unified RBAC and ABAC Risk-Aware Access Control Approach. *Future Internet*. 2025. Т. 17, № 6. С. 262. DOI: https://doi.org/10.3390/fi17060262.
10. Enhancing Microservices Security with Token-Based Access Control Method / A. Venčkauskas та ін. *Sensors*. 2023. Т. 23, № 6. С. 3363. DOI: https://doi.org/10.3390/s23063363.

References

1. Access Control Design Practice and Solutions in Cloud-Native Architecture: A Systematic Mapping Study / M. S. Rahaman et al. *Sensors*. 2023. Vol. 23, no. 7. P. 3413. DOI: https://doi.org/10.3390/s23073413.
2. Kummarapurugu C. S. Enhancing Serverless Computing Security in Multi-Cloud Environments: Integrating Policy-as-Code, Automated Compliance, and Dynamic Access Controls. *International Journal of Innovative Research in Engineering & Multidisciplinary Physical Sciences*. 2022. Vol. 10, no. 2. URL: https://www.researchgate.net/publication/389747418 (accessed: 19.08.2026).
3. Abibulaiev A., Pukach P., Vovk M. Context-Aware ML/NLP Pipeline for Real-Time Anomaly Detection and Risk Assessment in Cloud API Traffic. *Machine Learning and Knowledge Extraction*. 2026. Vol. 8, no. 1. P. 25. DOI: https://doi.org/10.3390/make8010025.
4. Arif T., Jo B., Park J. H. A Comprehensive Survey of Privacy-Enhancing and Trust-Centric Cloud-Native Security Techniques Against Cyber Threats. *Sensors*. 2025. Vol. 25, no. 8. P. 2350. DOI: https://doi.org/10.3390/s25082350.
5. Securing Workflows Using Microservices and Metagraphs / L. Miller et al. *Electronics*. 2021. Vol. 10, no. 24. P. 3087. DOI: https://doi.org/10.3390/electronics10243087.
6. Security in Cloud-Native Services: A Survey / T. Theodoropoulos et al. *Journal of Cybersecurity and Privacy*. 2023. Vol. 3, no. 4. P. 758–793. DOI: https://doi.org/10.3390/jcp3040034.
7. Static-Analysis-Based Solutions to Security Challenges in Cloud-Native Systems: Systematic Mapping Study / M. S. Rahaman et al. *Sensors*. 2023. Vol. 23, no. 4. P. 1755. DOI: https://doi.org/10.3390/s23041755.
8. A Survey of Context-Aware Access Control Mechanisms for Cloud and Fog Networks: Taxonomy and Open Research Issues / A. S. M. Kayes et al. *Sensors*. 2020. Vol. 20, no. 9. P. 2464. DOI: https://doi.org/10.3390/s20092464.
9. Atlam H. F., Yang Y. Enhancing Healthcare Security: A Unified RBAC and ABAC Risk-Aware Access Control Approach. *Future Internet*. 2025. Vol. 17, no. 6. P. 262. DOI: https://doi.org/10.3390/fi17060262.
10. Enhancing Microservices Security with Token-Based Access Control Method / A. Venčkauskas et al. *Sensors*. 2023. Vol. 23, no. 6. P. 3363. DOI: https://doi.org/10.3390/s23063363.

Історія статті:
Отримано: 30.05.2026 Доопрацьовано: 11.09.2026 Прийнято до друку: 23.09.2026 Опубліковано: 29.09.2026