Форма № 24

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет інформаційної безпеки та електронних комунікацій
(повне найменування факультету)

Кафедра інформаційної безпеки та наноелектроніки
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)
магістр
(ступінь вищої освіти)

на тему Прикладні аспекти вдосконалення механізмів автентифікації в розподілених інформаційних системах
(назва теми)

Виконав: студент 2 курсу, групи БКз-714м

Спеціальності 125 Кібербезпека та захист інформації
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Системи технічного захисту інформації, автоматизація її обробки

ЯКУБЕНКО С.А.
(ПРІЗВИЩЕ та ініціали)

Керівник КОРОТУН А.В.
(ПРІЗВИЩЕ та ініціали)

Рецензент ЛИТВИЦЬКИЙ О.П.
(ПРІЗВИЩЕ та ініціали)

2025

Форма № 25

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет інформаційної безпеки та електронних комунікацій
Кафедра інформаційної безпеки та наноелектроніки
Ступінь вищої освіти магістр
Спеціальність 125 Кібербезпека та захист інформації
(код і найменування)
Освітня програма (спеціалізація) Системи технічного захисту інформації, автоматизація її обробки
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ
Завідувач кафедри ІБтаН
Андрій КОРОТУН
«____» _____2025року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА

ЯКУБЕНКА Станіслава Андрійовича
(ПРІЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Прикладні аспекти вдосконалення механізмів автентифікації в розподілених інформаційних системах
Applied aspects of improving authentication mechanisms in distributed information systems
керівник проєкту (роботи) к.ф.-м.н. доцент, КОРОТУН Андрій Віталійович,
(науковий ступінь, вчене звання, ПРІЗВИЩЕ, ім'я, по батькові)
затверджені наказом закладу вищої освіти від «26» листопада 2025 року № 530
2. Строк подання студентом проєкту (роботи) 19.12.2025
3. Вихідні дані до проєкту (роботи) Архітектура розподілених інформаційних систем, протоколи автентифікації в розподілених системах, загрози в системах автентифікації.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Концепція та архітектура розподілених інформаційних систем; Аналіз вразливостей механізмів автентифікації; Огляд існуючих протоколів автентифікації; Принципи Нульової довіри; Формалізація моделі динамічної оцінки ризику доступу; Архітектура системи «Адаптивний шлюз Нульової довіри»; Алгоритм адаптивної видачі токенів; Семантична структура модифікованого токена.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)
Презентація доповіді (в MS PowerPoint), 16 слайдів.

6. Консультанти розділів проєкту (роботи)

Розділ	ПРІЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1 – 3	КОРОТУН А. В., зав. кафедри ІБтаН	04.09.2025	19.12.2025
Нормоконтроль	КОРОЛЬКОВ Р. Ю., доцент кафедри ІБтаН		18.12.2025

7. Дата видачі завдання «04» вересня 2025року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1.	Аналіз літературних джерел за тематикою дослідження.	04.09.25 – 18.09.25	Виконано
2.	Огляд архітектур розподілених інформаційних систем.	19.09.25 – 04.10.25	Виконано
3.	Дослідження протоколів і загроз системи автентифікації.	05.10.25 – 18.10.25	Виконано
4.	Аналіз концепції Нульової довіри	19.10.25 – 02.11.25	Виконано
5.	Формалізація моделі динамічної оцінки ризику доступу	03.11.25 – 20.11.25	Виконано
6.	Проєктування модифікованого протоколу автентифікації	21.11.25 – 10.12.25	Виконано
7.	Оформлення матеріалів магістерської роботи.	11.12.25 – 17.12.25	Виконано

Студент _____ Станіслав ЯКУБЕНКО
(підпис) (Ім'я ПРІЗВИЩЕ)

Керівник проєкту (роботи) _____ Андрій КОРОТУН
(підпис) (Ім'я ПРІЗВИЩЕ)

АНОТАЦІЯ

Пояснювальна записка до магістерської роботи: 63 с., 2 табл., 9 рис., 1 дод., 60 джерел.

АВТЕНТИФІКАЦІЯ, ІНФОРМАЦІЙНА БЕЗПЕКА, КІБЕРАТАКА, РИЗИК ІНФОРМАЦІЙНОЇ БЕЗПЕКИ, РОЗПОДІЛЕНА ІНФОРМАЦІЙНА СИСТЕМА, НУЛЬОВА ДОВІРА, ПРОТОКОЛ АВТЕНТИФІКАЦІЇ.

Актуальність теми. Постійне зростанням кількості та складності розподілених інформаційних систем (хмарних обчислень, Інтернету речей, тощо) призвело до значного зростанням кібератак, особливо спрямованих на перехоплення облікових даних. Традиційні монолітні механізми автентифікації стають неефективними або вразливими до нових видів загроз та вимагають впровадження більш надійніших протоколів розмежування доступу до розподілений ресурсів.

Мета роботи: підвищення рівня захищеності розподілених інформаційних систем шляхом вдосконалення механізмів автентифікації користувачів і сервісів.

Об’єкт дослідження: процеси контролю доступу та забезпечення інформаційної безпеки в розподілених інформаційних системах.

Предмет дослідження: методи, протоколи та засоби вдосконалення процедур автентифікації користувачів і сервісів у розподілених інформаційних систем.

Методи дослідження: порівняльний аналіз, формалізація, системний підхід, моделювання.

Задачі дослідження:

– проаналізувати сучасний стан захищеності розподілених інформаційних систем;

– розробити концептуальну модель адаптивної автентифікації на принципах Zero Trust;

– побудувати математичну модель динамічної оцінки ризиків доступу;

– спроєктувати архітектуру та верифікувати модифікований протокол автентифікації.

Практичне значення одержаних результатів. Розроблені рекомендації можуть бути використані ІТ-спеціалістами та архітекторами безпеки для модернізації застарілих систем автентифікації. Результати роботи можуть стати основою для навчальних курсів та методичних посібників з кібербезпеки.

ABSTRACT

Explanatory note to the master's thesis: 63 pages, 2 tables, 9 figures, 1 app., 60 sources.

AUTHENTICATION, INFORMATION SECURITY, CYBER ATTACK, INFORMATION SECURITY RISK, DISTRIBUTED INFORMATION SYSTEM, AUTHENTICATION PROTOCOL, ZERO TRUST.

Actuality of theme. The constant growth in the number and complexity of distributed information systems (cloud computing, Internet of Things, etc.) has led to a significant increase in cyberattacks, especially aimed at intercepting credentials. Traditional monolithic authentication mechanisms are becoming ineffective or vulnerable to new types of threats and require the implementation of more reliable protocols for delimiting access to distributed resources.

The purpose of the work is increasing the level of security of distributed information systems by improving user and service authentication mechanisms.

The object of research is access control processes and information security in distributed information systems.

The subject of the study is methods, protocols and tools for improving user and service authentication procedures in distributed information systems.

Research methods: comparative analysis, formalization, systems approach, modeling.

Research objectives:

– to analyze the current state of security of distributed information systems;

– to develop a conceptual model of adaptive authentication based on the principles of Zero Trust;

– to build a mathematical model of dynamic access risk assessment; – to design the architecture and verify the modified authentication protocol.

Practical significance of the obtained results. The developed recommendations can be used by IT professionals and security architects to modernize legacy authentication systems. The results of the work can become the basis for cybersecurity training courses and methodological guides.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

IT – інформаційні технології;

PIC – розподілена інформаційна система;

2FA – two-factor authentication (двофакторна автентифікація);

ABAC – attribute- based access control (управління доступом на основі атрибутів);

ACR – authentication context recognition (розпізнавання контексту автентифікації);

AiTM – adversary-in-the-middle (супротивник посередині / Атака «людина посередині»);

API – application programming interface (інтерфейс прикладного програмування);

BIAS – biometric identity assurance services (сервіси забезпечення біометричної ідентифікації);

BYOD – bring your own device (концепція «Принеси свій власний пристрій»);

CARTA – continuous adaptive risk and trust assessment (безперервна адаптивна оцінка ризиків та довіри);

CBEFF – common biometric exchange formats framework (рамкова структура загальних форматів обміну біометричними даними);

DB – database (база даних);

DLT – distributed ledger technology (технологія розподіленого реєстру);

DPoP – demonstrating proof-of-possession (демонстрація доказу володіння);

EAP – extensible authentication protocol (розширюваний протокол автентифікації);

FAR – false acceptance rate (коефіцієнт помилкових допусків / помилка другого роду);

FIDO2 – fast identity online 2 (відкритий стандарт для безпечної автентифікації без паролів);

FRR – false rejection rate (коефіцієнт помилкових відмов / помилка першого роду);

gRPC – Google remote procedure call (система віддаленого виклику процедур Google);

HTTP – hypertext transfer protocol (протокол передачі гіпертекстових документів);

HTTPS – hypertext transfer protocol secure (захищений протокол передачі гіпертексту);

IaaS – infrastructure-as-a-service (інфраструктура як послуга);

IdP – identity provider (провайдер ідентичності);

IDS – intrusion detection system (система виявлення вторгнень);

IEC – International electrotechnical commission (Міжнародна електротехнічна комісія);

IoT – Internet of things (Інтернет речей);

IPS – intrusion prevention system (система запобігання вторгненням);

ISO – International organization for standardization (Міжнародна організація зі стандартизації);

ISP – Internet service provider (постачальник інтернет-послуг / Провайдер);

JSON – JavaScript object notation (запис об'єктів JavaScript);

JWT – JSON web token (веб-токен JSON);

KDC – key distribution centre (центр розподілу ключів);

MFA – multi- factor authentication (багатофакторна автентифікація);

mTLS – mutual transport layer security (взаємна безпека транспортного рівня / взаємна TLS-автентифікація);

NIST – National institute of standards and technology (Національний інститут стандартів і технологій);

OAuth 2.0 – open authorization 2.0 (відкритий протокол авторизації 2.0);

OIDC – OpenID Connect (відкритий стандарт децентралізованої автентифікації);

OTP – one-time password (одноразовий пароль);

PaaS – platform-as-a-service (платформа як послуга);

PAD – presentation attack detection (виявлення атак на пред'явлення / виявлення підробок біометрії);

PDP – policy decision point (точка прийняття рішень);

PEP – policy enforcement point (точка виконання рішень);

RBAC – role- based access control (управління доступом на основі ролей);

REST – representational state transfer (передача репрезентативного стану);

S2S – service-to- service (взаємодія типу «сервіс-сервіс»);

SaaS – software-as-a-service (програмне забезпечення як послуга);

SAML – security assertion markup language (мова розмітки тверджень безпеки);

SIEM – security information and event management (система управління інформацією про безпеку та подіями безпеки);

SOA – service-oriented architecture (сервіс-орієнтована архітектура);

SOAP – simple object access protocol (простий протокол доступу до об'єктів);

SOC – security operations centre (операційний центр безпеки);

SSO – single sign-on (технологія єдиного входу);

TPM – trusted platform module (довірений платформний модуль);

TTL – time to live (час життя);

UI – user interface (інтерфейс користувача);

UID – user identifier (унікальний ідентифікатор користувача);

UTC – Universal time coordinated (Всесвітній координований час);

VPN – virtual private network (віртуальна приватна мережа);

WebAuthn – web authentication (веб-автентифікації)

XML – eXtensible markup language (розширювана мова розмітки);

ZKP – Zero-knowledge proof (доказ з нульовим розголошенням);

ZTA – Zero trust architecture (архітектура нульової довіри).

ВСТУП

Стрімкий розвиток інформаційно-комунікаційних технологій призвів до фундаментальної зміни архітектури програмних комплексів. Перехід від монолітних систем до розподілених інформаційних систем (РІС), таких як хмарні платформи, мікросервісні середовища та мережі Інтернету речей (IoT), став домінуючим трендом цифровізації. Однак така децентралізація, забезпечуючи гнучкість та масштабованість, водночас руйнує традиційне поняття «периметра безпеки». У цих умовах критично важливим бар’єром захисту стає автентифікація – процедура підтвердження справжності суб’єкта доступу [1].

Сучасні реалії демонструють, що статичні парольні методи [2-8] вичерпали свій ресурс надійності. Зростання обчислювальних потужностей зловмисників та вдосконалення методів соціальної інженерії роблять класичні підходи вразливими. Водночас специфіка РІС висуває нові вимоги: механізми автентифікації мають бути не лише криптографічно стійкими, але й забезпечувати "безшовний" досвід користувача (SSO), підтримувати мобільність та працювати в умовах ненадійних каналів зв’язку. Існуючі стандарти (OAuth 2.0, OpenID Connect) [9] вирішують частину проблем, проте їх прикладна реалізація часто містить архітектурні вразливості або створює надмірне навантаження на інфраструктуру.

Таким чином, необхідність дослідження та вдосконалення механізмів автентифікації, зокрема впровадження адаптивних та біометричних технологій у розподілені середовища, є важливим завданням в контексті модернізації застарілих систем автентифікації..

1 ТЕОРЕТИЧНІ ОСНОВИ АВТЕНТИФІКАЦІЇ В РОЗПОДІЛЕНИХ ІНФОРМАЦІЙНИХ СИСТЕМАХ

1.1 Концепція та архітектура розподілених інформаційних систем

Розподілена інформаційна система (РІС) – це мережа взаємопов'язаних комп'ютерів або вузлів, які працюють разом як єдине ціле, в умовах фізичного розосередження (рис. 1.1). Вузли такої системи взаємодіють для досягнення спільної мети, часто пов'язаної з обробкою, зберіганням та обміном даними чи послугами [10-12]. На відміну від централізованої системи, де один комп'ютер керує всіма завданнями, РІС розподіляє завдання та дані між кількома вузлами, які можуть бути географічно розділені [13, 14].

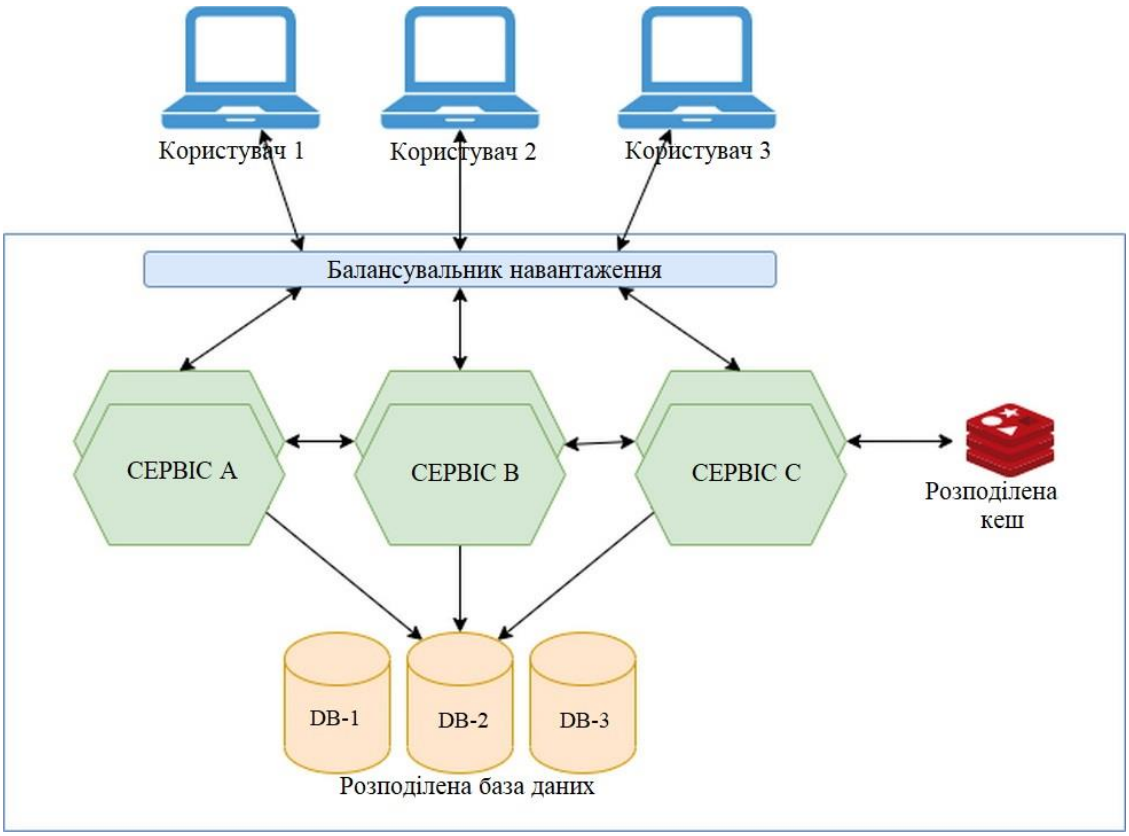


Рисунок 1.1 – Узагальнена архітектура РІС [12].

Така архітектура підвищує відмовостійкість, масштабованість та використання ресурсів, що робить PIC поширеними в різних сферах, включаючи хмарні обчислення, веб-додатки, наукові дослідження та телекомунікації [14].

Сучасні PIC класифікують за архітектурними підходами.

1. Мікросервісна архітектура: система розбивається на набір невеликих, слабко пов'язаних сервісів, кожен з яких виконує свою бізнес-функцію і комунікує через легковажні протоколи (наприклад, HTTP/REST або gRPC). Це найбільш поширений підхід у сучасній розробці [15]. На рис. 1.2 наведена узагальнена мікросервісна архітектура в порівнянні монолітною.

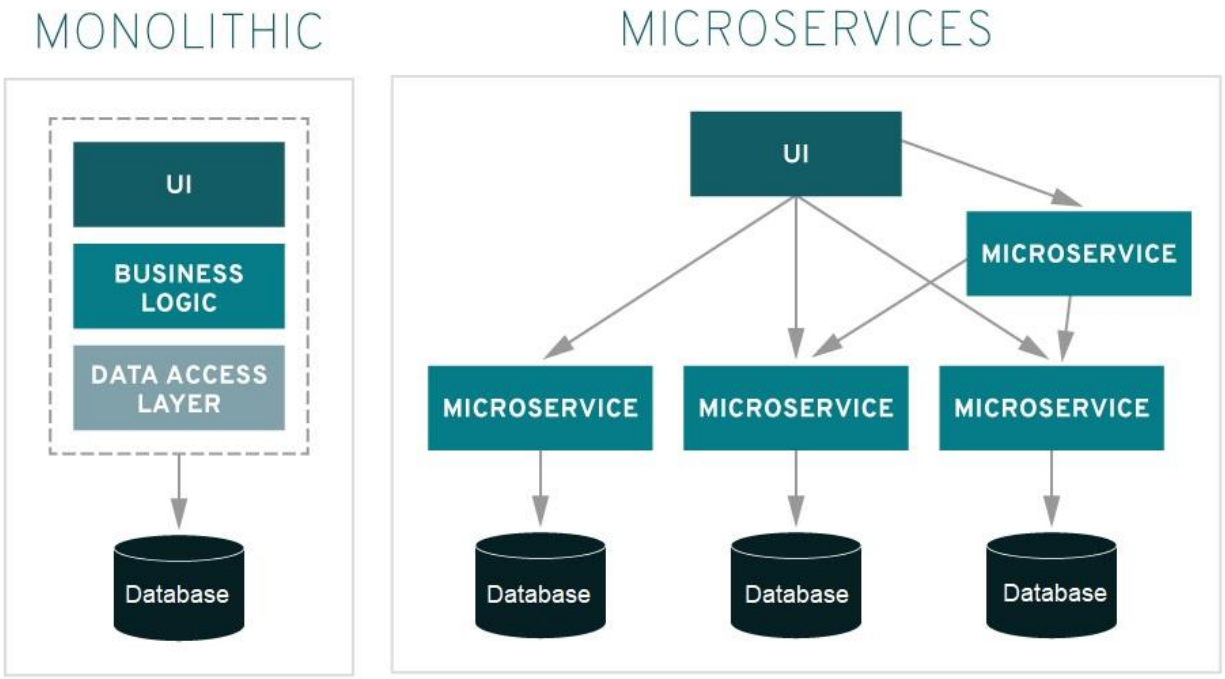


Рисунок 1.2 – Монолітна та мікросервісна архітектури [15].

2. Хмарні обчислення (cloud computing) [16]: інфраструктура, що надається як послуга [17]: «Інфраструктура як послуга» (Infrastructure-as-a-Service, IaaS), «Платформа як послуга» (Platform-as-a-Service, PaaS), «Програмне забезпечення як послуга» (Software-as-a-Service, SaaS) та інші.

Тут розподіленість реалізується на рівні віртуалізації ресурсів та географічного рознесення дата-центрів (рис. 1.3).

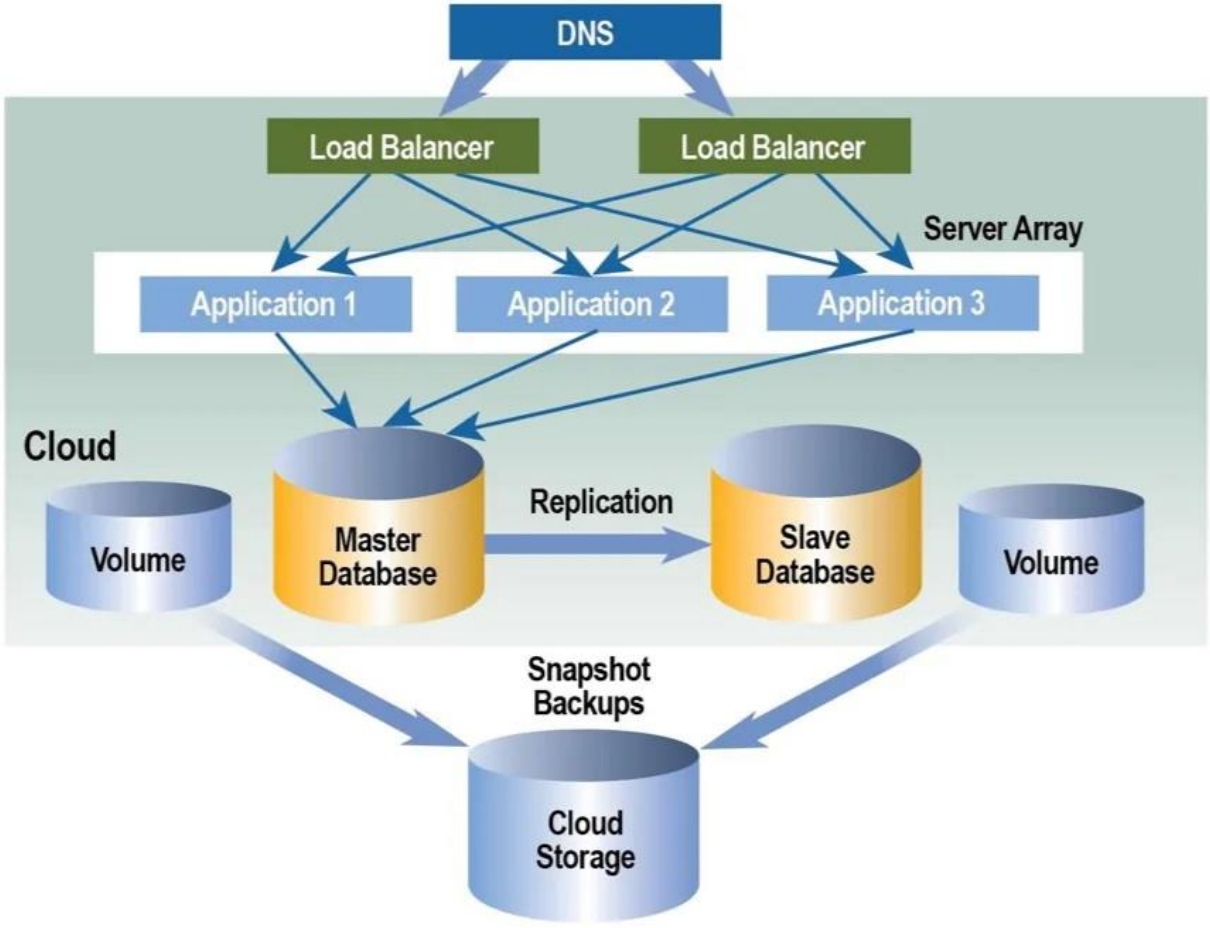


Рисунок 1.3 – Архітектура хмарних обчислень [16].

3. Інтернет речей (Internet of things, IoT): багаторівнева архітектура, яка забезпечує взаємодію фізичних пристроїв із хмарними сервісами для збору, обробки та аналізу даних. Основна структура включає сенсори/пристрої, мережевий рівень (шлюзи), рівень обробки даних (туманні/хмарні обчислення) та прикладний рівень [18]. Ця система об'єднує апаратне та програмне забезпечення для автоматичного керування

4. Блокчейн та децентралізовані системи (Distributed ledger technology, DLT) [19 – 29]: архітектура, де відсутній центральний керуючий орган. Дані та процеси валідації розподілені між усіма учасниками мережі, що вимагає специфічних криптографічних методів автентифікації. В блокчейн

архітектурі розподілена система розділена на сегменти – шарди (shard). Шардовані шаблони [14] дозволять горизонтально масштабувати рівень даних, щоб уникнути вузьких місць продуктивності та окремих точок відмови (рис. 1.4).

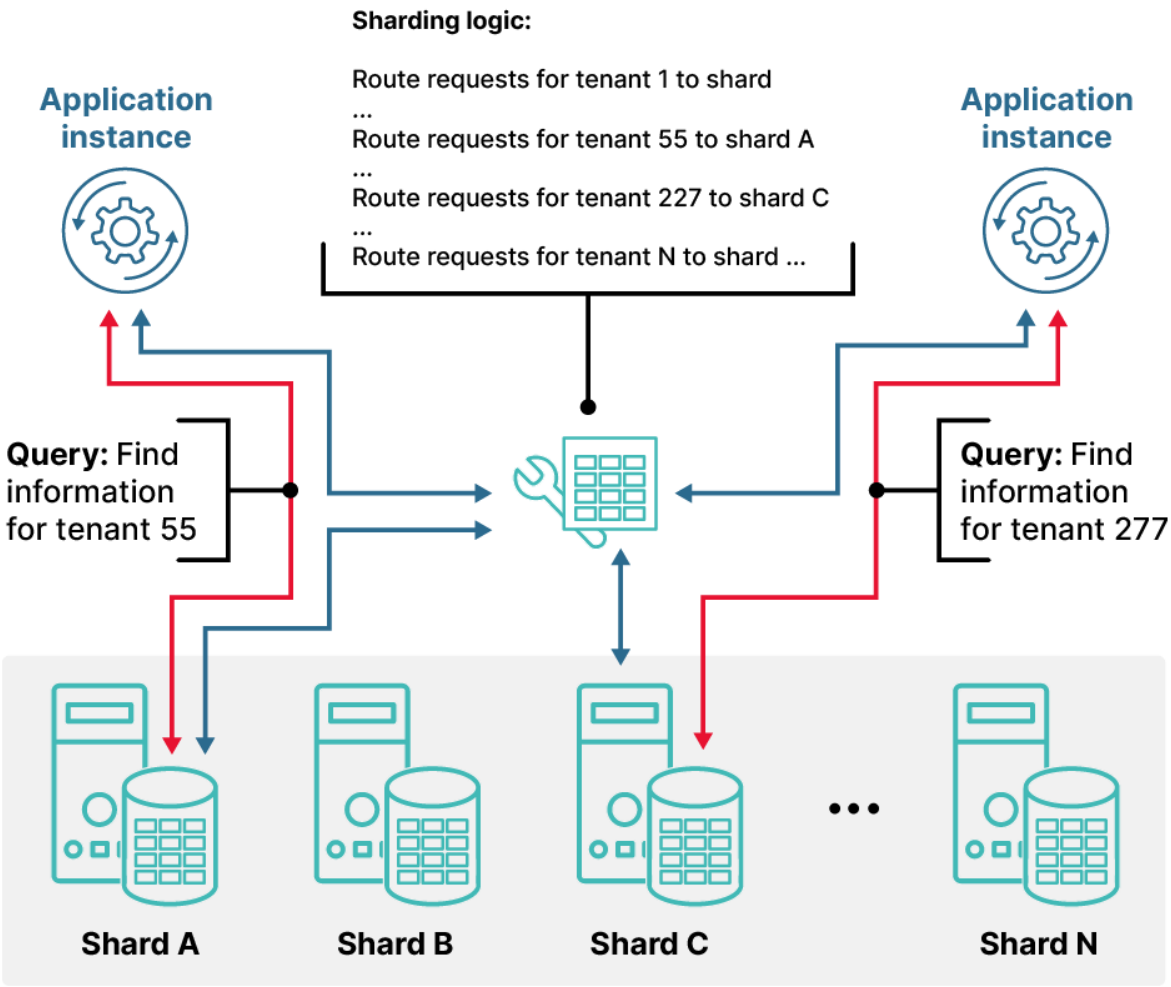


Рисунок 1.4 – Архітектура блокчейн [14].

1.2 Аналіз вразливостей механізмів автентифікації в PIC

В порівнянні з монолітними системами ландшафт загроз безпеці PIC кардинально змінився завдяки особливостям архітектури розподілених систем.

1. Розмивання периметру (Zero trust necessity). У традиційних системах існував захищений контур (периметр). У PIC, особливо хмарних та мікросервісних, поняття «внутрішньої довіреної мережі» зникає. Кожен запит, навіть між внутрішніми сервісами, має бути автентифікований.

2. Динамічність інфраструктури. Сервіси автоматично масштабуються, створюються та знищуються (контейнеризація, Kubernetes) [30]. IP-адреси є нереальними, тому прив'язка довіри до мережевої адреси є неможливою.

3. Множинність точок доступу. Замість одного входу (шлюзу) система може мати десятки публічних інтерфейсів програмування застосунків (Application programming interface, API), IoT-пристроїв та партнерських інтеграцій, кожна з яких є потенційним вектором атаки.

4. Проблема відкликання токенів (token revocation), Оскільки JWT (JSON (JavaScript object notation) web token) є «безстановим» – stateless (сервер не зберігає стан сесії), миттєво відкликати доступ (наприклад, при звільненні співробітника) складно [31 – 33]. Потрібні «чорні списки» або короткий час життя токенів, що збільшує навантаження на мережу.

5. Горизонтальне переміщення (lateral movement). Якщо зловмисник компрометує один мікросервіс, він може спробувати використати його права для доступу до інших сервісів. Це вимагає впровадження взаємної (двосторонньої) автентифікація, наприклад протокол mTLS (mutual Transport Layer Security) [34] та суворих політик доступу «сервіс-до-сервісу».

6. Складність управління ключами. У розподіленій системі сотні сервісів повинні мати доступ до публічних ключів для перевірки підписів токенів, що створює ризики при ротації ключів та управлінні сертифікатами.

В розподілених системах вектори атак зміщуються з перебору паролів на перехоплення сесій та маніпуляції токенами. До типових атак на систему автентифікації PIC можна віднести [35 – 37]:

– атаки повторного відтворення (replay attacks). Зловмисник перехоплює валідний токен (наприклад, JWT) або квиток Kerberos і

намагається використати його для доступу до ресурсу. Захисними механізмами до таких атак є використання протоколу HTTPS, токенів «з коротким життям» (time to Live, TTL), механізми ротації токенів (refresh tokens), прив'язка токена до відбитка клієнта (Demonstrating proof-of-possession, DPoP);

– фішинг та AiTM (Adversary-in-the-Middle). Класичний фішинг еволюціонував. Атаки AiTM дозволяють перехоплювати не тільки пароль, а й код двофакторної автентифікації (2FA) або сесійні куки в реальному часі;

– атаки з використанням витоків (credential stuffing). Автоматизоване використання пар логін/пароль, знайдених у базах витоків інших сервісів. Оскільки користувачі часто повторюють паролі, це залишається ефективним методом.

1.3 Огляд існуючих протоколів автентифікації

Традиційні протоколи епох корпоративних мереж із чітким периметром:

– Kerberos. Мережевий протокол автентифікації, який використовує концепцію «квитків» (tickets), щоб дозволити вузлам спілкуватися через незахищену мережу. Використовує довірену третю сторону (KDC – Key distribution center). Має обмеження: важко масштабується в глобальному інтернеті, вимагає суворої синхронізації часу.

– SAML (Security Assertion Markup Language). Стандарт на основі XML для обміну даними автентифікації та авторизації. Є основним стандартом для корпоративного SSO (Single Sign-On). Дозволяє користувачеві увійти в систему одного провайдера (identity provider – IdP) і отримати доступ до інших.

Протоколи, розроблені для роботи в мережі Інтернет (HTTP) та мобільних середовищах [38 – 43]:

- OAuth 2.0: фреймворк авторизації, який дозволяє додаткам отримувати обмежений доступ до акаунтів користувачів на HTTP-сервісах. Він делегує доступ, не передаючи пароль користувача.
- OpenID Connect (OIDC): шар ідентифікації поверх OAuth 2.0. Він дозволяє клієнтам верифікувати особу користувача на основі аутентифікації, виконаної сервером авторизації.
- JWT (JSON web token): компактний, безпечний для URL спосіб представлення заявок (claims) між двома сторонами. В розподілених системах JWT є стандартом де-факто для мікросервісів, оскільки токен є безстановим/самодостатнім (stateless). Сервіс може перевірити підпис токена і довіряти даним всередині без звернення до центральної бази даних при кожному запиті.

Розвиток біометрії та криптографії призвів до появи стандартів, що усувають найслабшу ланку – пароль. Біометрична автентифікація в PIC вимагає балансу між зручністю, безпекою передачі даних та приватністю (недопущення витоку «сирих» біометричних даних). Тому протоколи в цій сфері поділяються на стандарти обміну даними, протоколи перевірки та архітектурні фреймворки.

До ключових протоколів і стандартів, які використовуються для біометрії в PIC, відносяться [44 – 48]:

- FIDO2 / WebAuthn: відкритий стандарт для автентифікації в Web. Використовує асиметричну криптографію. Приватний ключ зберігається безпечно на пристрої користувача (в TPM (trusted platform module) модулі, на YubiKey або в смартфоні), а публічний – на сервері. Сервер надсилає «виклик» (challenge), який пристрій підписує приватним ключем після біометричної перевірки користувача. Використовує модель Match-on-Device. Біометричні дані (відбиток, обличчя) ніколи не покидають пристрій користувача. При

цьому, сервер не зберігає базу біометричних шаблонів, що усуває ризик їх масового витоку;

– протоколи на базі OASIS BIAS (Biometric identity assurance services). Це стандарт для сервіс-орієнтованих архітектур (SOA), який визначає, як біометричні сервіси (верифікація, ідентифікація) мають викликатися через мережу. Використовує XML/SOAP повідомлення для запиту біометричних операцій. Часто використовується в державних та банківських системах, де є централізований «біометричний двигун», до якого звертаються різні філії або агенції;

– IEEE 802.1X та EAP (extensible authentication protocol). Використовуються для контролю доступу до мережі (наприклад, корпоративний Wi-Fi). Протоколи EAP-TLS / EAP-TTLS можуть служити «тунелем» для безпечної передачі біометричних даних від клієнта до сервера автентифікації (RADIUS). Дозволяють інтегрувати біометрію на рівні входу в мережу, а не лише в окремі додатки.

У розподілених системах різні вузли (сканери, сервери) можуть бути від різних виробників. Щоб вони «розуміли» один одного, використовуються стандарти ISO – стандарти обміну даними (data interchange standards):

– CBEFF (common biometric exchange formats framework – ISO/IEC 19785): рамка для біометричних даних. Додає метадані (який алгоритм, яка якість скану), щоб будь-який вузол PIC міг коректно обробити отриманий шаблон;

– ISO/IEC 19794: серія стандартів, що описує формати самих зображень (fingerprint minutiae, face image data) для забезпечення взаємодії (interoperability).

Для сучасних PIC, де вузли можуть бути недовіреними, розробляються спеціальні криптографічні протоколи з посиленою приватністю (privacy-preserving protocols) [49 – 53]:

– скасовна біометрія (cancelable biometrics): «сирий» біометричний образ трансформується за допомогою незворотної функції на стороні клієнта. На сервер передається лише трансформований хеш. Якщо база даних викрадена, хеш можна «скасувати» і згенерувати новий (змінивши параметри функції), як звичайний пароль;

– біометрія з доказом нульового розголошення (zero-knowledge proof (ZKP) biometrics): протоколи, що дозволяють довести серверу, що користувач є тим, за кого себе видає (наприклад, «відстань між векторами обличчя менша за поріг X»), не розкриваючи самих векторів. За допомогою даного протоколу можна значно спростити цифрову аутентифікацію, не використовуючи паролі та іншу конфіденційну інформацію;

– ISO/IEC 30107 (presentation attack detection – PAD) – стандарт перевірки «живучості» (liveness detection). У PIC це критично, оскільки зловмисник може перехопити відеопотік або фото. Стандарт описує механізми виявлення атак (фото, маски, діпфейки) на стороні клієнта або сервера.

В табл. 1.1 наведені характеристики деяких біометричних протоколів.

Таблиця 1.1 – Характеристики біометричних протоколів

Тип протоколу	**Представники**	**Де використовується в PIC**
Веб/хмарний стандарт	FIDO2 / WebAuthn	Вхід у веб-портали, мобільні додатки, SaaS.
Корпоративні сервіси	OASIS BIAS	Банківські сервіси, державні реєстри.
Мережевий доступ	IEEE 802.1X (EAP)	Захист Wi-Fi, VPN, фізичних портів доступу.
Формат даних	CBEFF (ISO 19785)	Обмін даними між сканерами та серверами різних вендорів.
Покращена конфіденційність	Cancelable / ZKP	Блокчейн-системи, високозахищені сховища.

1.4 Висновки до розділу 1

У цьому розділі проведено аналіз теоретичних основ автентифікації в розподілених інформаційних системах, розглянуто їхню архітектуру, вразливості та сучасні протоколи захисту. Головною відмінністю РІС від монолітних систем є зникнення чіткого захищеного периметра, що вимагає переходу до концепції Zero Trust («нульової довіри»), де кожен запит повинен бути автентифікований незалежно від його походження.

Специфічнми вразливостями розподілених систем є динамічність інфраструктури та множинність точок доступу (API, IoT), що роблять неможливою прив'язку довіри до IP-адрес. Забезпечення надійної автентифікації в сучасних РІС вимагає комплексного підходу, що поєднує використання безстанових токенів, взаємної автентифікації (mTLS), сучасних біометричних стандартів та криптографічних методів захисту від атак на рівні сесій.

2 КОНЦЕПЦІЯ ВДОСКОНАЛЕННЯ МЕХАНІЗМІВ АВТЕНТИФІКАЦІЇ

2.1 Принципи Нульової довіри

Глобальні тенденції до децентралізації, зростання кількості віддалених підключень, масове використання хмарних сервісів та фрагментація додатків на мікросервіси призвели до того, що поняття «довірена внутрішня мережа» втратило свою актуальність. У цьому розділі обґрунтовується необхідність переходу від традиційної моделі захисту периметра (perimeter-based security) до парадигми Нульова довіри (Zero trust) як базису для побудови вдосконаленої системи автентифікації, а також визначається роль адаптивних механізмів у цій архітектурі [54 – 56].

Варто окремо зупинитися на аналізі еволюції загроз інформаційній безпеці, оскільки розуміння історичного контексту є ключем до усвідомлення необхідності впровадження нових підходів. Протягом останніх десятиліть парадигма захисту корпоративних даних зазнала кардинальних змін, які були продиктовані не стільки бажанням покращити технології, скільки вимушеною реакцією на зростаючу складність атак.

У епоху домінування монолітних архітектур, яка тривала до початку 2010-х років, основна увага приділялася захисту фізичного периметра мережі. Інформаційні системи будувалися за принципом середньовічної фортеці, де товсті стіни (міжемережеві екрани) захищали внутрішній двір від зовнішніх ворогів. Вважалося аксіомою, що будь-який суб'єкт, який пройшов процедуру первинної автентифікації на прохідній (шлюзі), автоматично є доброчесним. Така модель, відома як "perimeter-based security", була ефективною в умовах, коли всі співробітники працювали в офісі зі стаціонарних комп'ютерів, а корпоративні дані не покидали локальну мережу.

Однак цифрова трансформація бізнесу, масовий перехід на віддалену роботу та популяризація концепції BYOD (bring your own device – «принеси свій власний пристрій») зруйнували ці «стіни». Сучасний користувач отримує доступ до корпоративних ресурсів з кав'ярні, аеропорту або домашнього офісу, використовуючи при цьому особисті смартфони та планшети, рівень захищеності яких не піддається повному контролю з боку адміністраторів системи. У таких умовах мережевий периметр перестав бути фізичною межею і трансформувався у логічну сутність, яка проходить через кожен пристрій та кожну сесію доступу. Саме ця фундаментальна зміна ландшафту загроз зробила традиційні статичні методи автентифікації не просто застарілими, а й небезпечними, оскільки вони створюють ілюзію захищеності там, де її насправді немає. Зловмисники адаптувалися до нових реалій швидше за захисників, перемістивши вектор атак зі злому інфраструктури на викрадення цифрових особистостей, що й зумовлює актуальність переходу до адаптивних моделей нульової довіри.

Традиційна модель інформаційної безпеки, яку в літературі часто називають «фортеця та рів» (castle-and-moat), будується на бінарному припущенні: усі суб'єкти та трафік, що знаходяться ззовні корпоративної мережі, апріорі вважаються ворожими, а ті, що пройшли процедуру входу і знаходяться всередині периметра – довіреними. Такий підхід передбачає існування чіткої межі розмежування, що захищається міжмережевими екранами (Firewalls) та VPN-шлюзами.

Проте статистика кіберінцидентів останніх років свідчить про неефективність цього підходу. Згідно зі звітами провідних аналітичних агентств, значна частка успішних атак (до 70%) відбувається або через компрометацію облікових даних легітимних користувачів, або через інсайдерські загрози [57]. У такій ситуації зловмисник, подолавши зовнішній контур захисту, отримує можливість вільного горизонтального переміщення (lateral movement) всередині мережі, оскільки внутрішні системи не вимагають повторної суворої перевірки.

Архітектура Zero trust (ZTA), формалізована, зокрема, у стандарті NIST SP 800-207, пропонує фундаментальну зміну парадигми: «ніколи не довіряй, завжди перевіряй». У контексті автентифікації це означає повну відмову від концепції неявної довіри (implicit trust), що базується виключно на мережевому розташуванні суб'єкта або пристрою.

Основні фактори, що зумовлюють необхідність впровадження ZTA в сучасних PIC:

– розмивання периметру (De-perimeterization). У розподілених системах користувачі, пристрої та сервіси можуть знаходитися у будь-якій географічній точці. IP-адреса більше не може слугувати гарантом ідентичності або безпеки;

– статичність прав доступу. Традиційна автентифікація зазвичай перевіряє користувача лише один раз − на початку сесії. Якщо зловмисник перехоплює сесійний токен після входу (Session Hijacking), система не має механізмів виявлення аномалії до моменту закінчення терміну дії токена;

– гетерогенність клієнтів. Доступ до системи здійснюється не лише зі стандартизованих корпоративних ПК, а й з мобільних пристроїв (BYOD), IoT-датчиків та через API сторонніх партнерів, що вимагає уніфікованого, але гнучкого підходу до валідації.

Для систематизації відмінностей між традиційним підходом та пропонованою концепцією Zero Trust складено порівняльну таблицю 2.1.

Впровадження ZTA вимагає, щоб процес автентифікації перемістився з мережевого шлюзу безпосередньо до логічних компонентів системи: PDP (policy decision point) – точки прийняття рішень, та PEP (policy enforcement point) – точки виконання рішень. PDP аналізує кожен запит на доступ до ресурсу, незалежно від походження цього запиту, та видає дозвіл або відмову в реальному часі.

Таблиця 2.1 – Порівняльний аналіз моделей автентифікації

Характеристика	**Периметральна модель**	**Модель Zero Trust**
Принцип довіри	"Довіряй, але перевіряй". Довіра базується на локації (всередині мережі/VPN).	"Ніколи не довіряй, завжди перевіряй". Відсутність неявної довіри.
Момент автентифікації	Одноразова перевірка при вході (login time).	Безперервна перевірка (continuous verification) кожного запиту.
Обсяг доступу	Широкий мережевий доступ після проходження шлюзу.	Мікросегментація та принцип найменших привілеїв (least privilege).
Контекст рішення	Статичний (пароль + логін).	Динамічний (користувач + пристрій + поведінка + середовище).
Реакція на загрози	Реактивна (після виявлення атаки IDS/IPS).	Проактивна (автоматичне блокування при зміні контексту ризику).

2.2 Принципи архітектури Zero trust

Для коректної побудови моделі автентифікації у даній магістерській роботі необхідно детально проаналізувати сім фундаментальних принципів, викладених у публікації NIST SP 800-207 [58], та адаптувати їх до контексту розподілених систем.

1. Всі джерела даних та обчислювальні сервіси вважаються ресурсами. Це означає, що механізм автентифікації повинен захищати не лише доступ користувача до інтерфейсу, а й взаємодію типу "сервіс-сервіс" (S2S). Наприклад, запит мікросервісу Billing до бази даних повинен проходити таку ж сувору перевірку ідентичності (через mTLS або JWT), як і вхід зовнішнього користувача.

2. Усі комунікації захищені незалежно від розташування мережі. Трафік у внутрішній мережі кластера (наприклад, Kubernetes) не повинен

передаватися у відкритому вигляді. Обов'язковим є шифрування на транспортному рівні.

3. Доступ до окремих ресурсів надається на сеансовій основі. Поняття "сеанс" трансформується. Якщо раніше сесія могла тривати 24 години, то в Zero Trust доступ надається до конкретного ресурсу на мінімально необхідний час. Це вимагає використання короткоживучих токенів (Short-lived access tokens) з часом життя $T_{ttl} \approx 5 - 10$ хвилин.

4. Доступ до ресурсів визначається динамічною політикою. Політика доступу P є функцією від стану системи: $P = f$(користувач, актив, середовище). Це означає перехід від статичного управління доступом RBAC (Role-based access control) до динамічного ABAC (Attribute-based access control).

5. Підприємство відстежує та вимірює цілісність і стан безпеки всіх активів. Система автентифікації повинна інтегруватися з системами моніторингу. Якщо на пристрої виявлено вразливість, рівень довіри до нього автоматично знижується.

6. Усі автентифікації та авторизації є динамічними та суворо виконуються. Це принцип CARTA (Continuous adaptive risk and trust assessment). Рейтинг довіри повинен перераховуватися не лише при логіні, а й протягом сесії.

7. Підприємство збирає якомога більше інформації про поточний стан активів та інфраструктури. Цей принцип обґрунтовує необхідність збору розширеної телеметрії для математичної моделі оцінки ризиків.

Центральним елементом реалізації концепції Zero trust є перехід від статичної до адаптивної (контекстно-залежної) автентифікації (adaptive authentication). Статичні методи не враховують змінний ландшафт загроз під час сесії. Адаптивна автентифікація − це динамічний процес вибору методу підтвердження особистості залежно від поточного рівня ризику.

Для цього необхідно розширити поняття факторів автентифікації. Традиційна теорія виділяє три класи:

– фактор знання (пароль) − вразливий до фішингу та перебору (H(P) ≈30 біт ентропії);

– фактор володіння (токен, телефон) − вразливий до фізичної крадіжки або соціальної інженерії;

– фактор властивості (біометрія) − має ймовірнісний характер помилок першого (FRR – false rejection rate) та другого (FAR – false acceptance rate) роду.

У розроблюваній системі ми вводимо четвертий фактор − Контекст. Контекст − це сукупність метаданих, що супроводжують запит. Його можна формалізувати як вектор атрибутів C:

$$C = \{U_{ID}, D_{dev}, L_{loc}, T_{time}, N_{net}, B_{beh}\} \quad (2.1)$$

де:

U_{ID} − ідентифікатор користувача;

D_{dev} − цифровий відбиток пристрою;

L_{loc} − геолокаційні дані;

T_{time} − часові характеристики;

N_{net} − репутація мережі (IP);

B_{beh} − поведінкова біометрія (швидкість набору тексту, рух миші).

Адаптивна система працює за логікою умовного переходу на основі функції оцінки ризику $R(C)$, яка повертає значення ймовірності загрози в діапазоні [0,1]:

$$\text{Дія} = \begin{cases} \text{Дозволено}, & \text{якщо} \quad R(C) < \theta_{low} \\ \text{Запит на БФА}, & \text{якщо} \quad \theta_{low} \le R(C) < \theta_{high} \\ \text{Заборонено}, & \text{якщо} \quad R(C) \ge \theta_{high} \end{cases} \quad (2.2)$$

де θ_{low} та θ_{high} − порогові значення ризику, що налаштовуються адміністратором.

Для ілюстрації роботи адаптивного механізму розглянемо сценарій атаки типу «Impossible travel» (Неможлива подорож). Припустимо, користувач автентифікується з Києва (t_1), а через 15 хвилин надходить запит з тим самим токеном з Лондона (t_2).

Традиційна система перевірить лише валідність криптографічного підпису токена і пропустить запит. Система Zero trust виконає розрахунок кінематичної правдоподібності. Відстань D між координатами розраховується за формулою гаверсинуса [59]:

$$D = 2r \cdot \arcsin\sqrt{\left(\sin\frac{\varphi_2 - \varphi_1}{2}\right)^2 + \cos(l_1)\cos(l_2)\left(\sin\frac{\lambda_2 - \lambda_1}{2}\right)^2}\,, \qquad (2.3)$$

де: φ_1, φ_2– широта точок 1 та 2, відповідно; λ_1, λ_2 – довгота точок 1 та 2, відповідно.

Швидкість переміщення визначається як:

$$V = \frac{D}{|t_2 - t_1|}\,, \qquad (2.4)$$

де: t_1 і t_2 – моменти часу запиту на доступ.

Якщо V перевищує фізично можливу швидкість пересування (наприклад, 900 км/год), система миттєво блокує сесію, незважаючи на правильний пароль чи токен. Це демонструє перевагу контекстної логіки над статичною.

2.3 Формалізація моделі динамічної оцінки ризику доступу

Перехід до адаптивної автентифікації вимагає створення суворої математичної моделі, здатної кількісно оцінити рівень загрози в будь-який момент часу. Якщо в традиційних системах результат автентифікації є

булевою величиною Auth $\in\{0,1\}$ (доступ дозволено або заборонено), то в запропонованій системі ми оперуємо неперервною величиною − рівнем довіри (trust score).

2.3.1 Формування векторного простору ознак

Нехай S − множина станів сесії користувача. Кожен запит до системи в момент часу t характеризується вектором ознак $X(t)$, який формує контекст події:

$$X(t) = \{x_1(t), x_2(t), \ldots, x_n(t)\}, \tag{2.5}$$

де n − кількість метрик, що відстежуються.

Для побудови моделі виділяємо чотири групи метрик, які мають найбільший вплив на оцінку ймовірності компрометації.

1. Геопросторова метрика (X_{geo}). Визначає аномальність фізичного розташування користувача. Вхідними даними є координати поточної сесії $L_{cur}(\varphi, \lambda)$ та координати попередньої успішної сесії $L_{prev}(\varphi, \lambda)$. Основним показником є швидкість переміщення між точками доступу. Відстань d обчислюється за ортодромією (найкоротшою відстанню на сфері) [59]:

$$d\,(L_{prev}, L_{cur}) = R \cdot \arccos\,(\sin\varphi_1 \sin\varphi_2 + \cos\varphi_1 \cos\varphi_2 \cos(\lambda_2 - \lambda_1)), \tag{2.6}$$

де R – радіус Землі.

2. Часова метрика (X_{time}). Характеризує відхилення часу активності від типового профілю користувача. Нехай H − множина даних про час входів користувача за останні 30 днів. Розподіл часу входів можна апроксимувати нормальним законом розподілу ймовірності $N(\mu, \sigma^2)$, де:

μ − математичне сподівання (середній час входу, наприклад, 09:30);

σ − стандартне відхилення.

Тоді відхилення поточного часу t_{cur} від профілю визначається як z-показник:

$$z = \frac{|t_{cur} - \mu|}{\sigma}. \quad (2.7)$$

3. Метрика пристрою та середовища (X_{dev}). Базується на порівнянні цифрового відбитка (fingerprint) поточного пристрою з переліком відомих пристроїв користувача D_{known}. Відбиток F формується як хеш-функція від набору параметрів браузера та ОС: Відбиток F визначається як результат застосування криптографічної хеш-функції H до конкатенації (об'єднання) рядкових значень п'яти ключових параметрів:

$$F = H(UserAgent||ScreenRes||TimeZone||Fonts||Canvas), \quad (2.8)$$

де символ || позначає операцію конкатенації (склеювання рядків).

UserAgent – агент користувача: комп'ютерна програма, що представляє користувача і виконує дії від його особи. Подається у вигляді рядка, який браузер передає серверу. Він містить детальну інформацію про тип та версію браузера, операційну систему (наприклад, Windows 10, macOS), її розрядність та версію ядра;

ScreenRes – роздільна здатність екрана: фізичні розміри дисплея в пікселях (ширина x висота) та глибина кольору. Це дозволяє розрізняти мобільні пристрої, ноутбуки та стаціонарні монітори;

TimeZone – часовий пояс: зсув часу відносно UTC. Це допомагає локалізувати користувача та перевірити відповідність його геолокації налаштуванням системи;

Fonts – перелік встановлених у системі шрифтів. Оскільки користувачі часто встановлюють специфічні шрифти для роботи (дизайн, програмування) або офісні пакети, цей набір є досить унікальним для кожного ПК;

Canvas – результат візуалізації графічного примітиву. Браузеру дається команда приховано відмалювати певний текст або 3D-фігуру. Через відмінності у драйверах відеокарт, браузерних двигунах та згладжуванні шрифтів, бінарний код отриманого зображення буде відрізнятися на різних пристроях.

Метрика X_{dev} приймає бінарне або дискретне значення залежно від збігу хешів або ступеня подібності (наприклад, відстань Левенштейна для рядків *UserAgent*).

4. Поведінкова біометрія (X_{beh}). Найскладніша метрика, що базується на аналізі динаміки роботи з клавіатурою (keystroke dynamics). Ключовими параметрами є:

– T_{hold} − час утримання клавіші;

– T_{lat} − час затримки між натисканням клавіш (digraph latency).

Відхилення поточного профілю (зразка) введення P_{cur} від еталонного P_{ref} розраховується через відстань Махаланобіса, яка враховує кореляцію між параметрами [60]:

$$D_M(P_{cur}) = \sqrt{\left(P_{cur} - P_{ref}\right)^T Y^{-1}\left(P_{cur} - P_{ref}\right)}, \tag{2.9}$$

де P_{cur} – це вектор поточних значень T_{hold} і T_{lat};

P_{ref} – це вектор еталонних (з минулого) значень T_{hold} і T_{lat};

Y − коваріаційна матриця еталонного профілю.

Коваріаційна матриця Y - це «карта розсіювання» звичок користувача. Вона описує, наскільки стабільно користувач друкує. Якщо користувач завжди друкує як робот (однаково) – розсіювання мале, і будь-яке відхилення буде підозрілим. Якщо користувач хаотичний (то швидко, то повільно) – розсіювання велике, і система буде менш суворою.

Як приклад, розглянемо побудову вектора P_{cur} для введеного паролю «ОК»:

$$P_{cur} = \begin{bmatrix} T_{hold}(\text{для клавіші "}O\text{"}) \\ T_{lat}(\text{між клавішами "}O\text{" та "}K\text{"}) \\ T_{hold}(\text{для клавіші "}K\text{"}) \end{bmatrix}. \quad (2.10)$$

2.3.2 Нормалізація вхідних даних

Оскільки вхідні метрики мають різну розмірність (кілометри, секунди, біти), для їх агрегації необхідно виконати процедуру нормалізації, відобразивши їх у діапазон [0,1], де 0 − мінімальний ризик (максимальна довіра), а 1 − максимальний ризик.

Використаємо для цього логістичні функції (сигмоїди), які дозволяють згладити порогові переходи.

1. Для геопозиції (R_{geo}): ризик зростає зі збільшенням швидкості переміщення v. Критичним порогом є фізично можлива швидкість $v_{\max}$ (наприклад, 900 км/год).

$$R_{geo}(v) = \frac{1}{1 + e^{-k(v - v_0)}} \quad , \quad (2.11)$$

де v_0 − точка перегину (середня допустима швидкість), k − коефіцієнт крутизни функції.

2. Для часу (R_{time}): ризик зростає при відхиленні від звичного графіка. Використовуємо функцію Гауса, інвертовану для оцінки ризику (чим менша ймовірність події, тим вищий ризик):

$$R_{time}(t) = 1 - e^{-\frac{(t-\mu)^2}{2\sigma^2}} \quad . \quad (2.12)$$

3. Для IP-репутації (R_{ip}): дана метрика зазвичай отримується від зовнішніх сервісів збору інформації про загрози (Threat intelligence services), наприклад, AbuseIPDB, у вигляді показника (S_{exter}) від 0 до 100.

$$R_{ip} = \frac{S_{exter}}{100} \quad . \tag{2.13}$$

2.3.3 Математична модель розрахунку інтегрального показника довіри

Після отримання нормалізованих оцінок ризику по кожному фактору $R_i \in [0,1]$, необхідно обчислити загальний рівень довіри до сесії T_S. Пропонується використати гібридну модель, що поєднує лінійну згортку з системою штрафних коефіцієнтів (penalty system) для критичних відхилень.

Базовий рівень ризику $Risk_{total}$ обчислюється як:

$$Risk_{total} = \sum_{i=1}^{n} w_i \cdot R_i \quad , \tag{2.14}$$

де w_i − ваговий коефіцієнт i-го фактора, що відображає його значущість для системи безпеки.

Однак, лінійна модель має недолік: низький ризик по одному фактору може компенсувати критичний ризик по іншому. Наприклад, зловмисник з правильного пристрою (вкрадений ноутбук), але з аномальної локації. Щоб уникнути цього, вводимо мультиплікативний коефіцієнт блокування K_{block}.

$$K_{block} = \prod_{i=1}^{n} \left(1 - \mathrm{I}\left(R_i > \theta_{kp}\right)\right) \quad , \tag{2.15}$$

де I − індикаторна функція, яка дорівнює 1, якщо частковий ризик R_i перевищує критичний поріг θ_{kp} (наприклад, 0.9), і 0 в іншому випадку. Якщо хоча б один фактор є критичним, K_{block} стає рівним 0.

Тоді фінальний показник довіри T_S (Trust score) визначається як обернена величина до ризику з урахуванням блокування:

$$T_s = K_{,block} \cdot (1 - Risk_{total}) \cdot 1000 \quad . \tag{2.16}$$

Для зручності програмної реалізації (аналогічно кредитному рейтингу) результат масштабовано до діапазону [0,1000].

2.3.4 Байєсівське уточнення для адаптивної системи

У процесі експлуатації системи накопичується статистика, що дозволяє уточнювати оцінку ризику. Для цього доцільно застосувати теорему Байєса, розглядаючи подію A – «користувач є легітимним», і подію B – «спостерігається вектор ознак X».

Апріорна ймовірність $P(A)$ − це довіра до користувача до початку аналізу (зазвичай базується на історії успішних входів).

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \quad , \tag{2.17}$$

де:

$P(A|B)$ − апостеріорна ймовірність легітимності (уточнений показник довіри T_S);

$P(B|A)$ − функція правдоподібності: наскільки ймовірним є поява таких ознак (наприклад, вхід о 3-й ночі) для цього конкретного легітимного користувача.

Це дозволяє реалізувати механізм самонавчання: якщо користувач систематично працює вночі, система з часом перестане вважати це аномалією, збільшуючи $P(B|A)$ для нічних годин, і, як наслідок, підвищуючи загальний рівень довіри.

2.3.5 Динаміка довіри в часі

Важливим аспектом, який часто ігнорується, є деградація довіри (trust decay). Довіра не може бути константою протягом всієї сесії. Що довше триває сесія без активних дій користувача або повторних перевірок, то вища ймовірність, що пристроєм заволодів зловмисник (наприклад, користувач відійшов від комп'ютера).

Вводимо функцію деградації довіри $T_S(t)$:

$$T_s(t) = T_s(t_0) \cdot e^{-\alpha(t-t_{last})} \quad , \tag{2.18}$$

де:

$T_S(t_0)$ − початковий рівень довіри при вході;

t_{last} − час останньої активної дії;

α − коефіцієнт деградації.

Якщо в момент часу t значення $T_S(t)$ падає нижче порогу θ_{MFA}, система ініціює активну перевірку – повторну автентифікацію (re-authentication).

Розроблена математична модель дозволяє формалізувати процес прийняття рішень про надання доступу, переходячи від суб'єктивних експертних оцінок до обчислюваних величин. Використання векторного

простору ознак, що включає геолокацію, часові мітки, характеристики пристрою та поведінкову біометрію, забезпечує багатовимірний аналіз контексту. Запропонований алгоритм розрахунку інтегрального показника довіри з використанням механізму «trust decay» створює основу для програмної реалізації модуля адаптивної автентифікації.

2.4 Висновки до розділу 2

У другому розділі роботи проведено теоретичне обґрунтування та математичну формалізацію вдосконаленої системи автентифікації, побудованої на принципах архітектури нульової довіри (Zero trust).

Встановлено, що статичні методи автентифікації створюють вразливість через наявність «неявної довіри після проходження шлюзу, що дозволяє зловмисникам вільно переміщуватися всередині мережі.

Визначено ключові принципи побудови системи автентифікації, адаптовані для розподілених систем. Основним постулатом прийнято принцип «ніколи не довіряй, завжди перевіряй» та перехід від статичного RBAC до динамічного атрибутивного управління доступом.

Запропоновано розширення класичної тріади факторів автентифікації четвертим фактором – «Контекстом». Формалізовано вектор атрибутів контексту, який включає ідентифікатор користувача, цифровий відбиток пристрою, геолокацію, часові характеристики, репутацію мережі та поведінкову біометрію.

Розроблено математичну модель динамічної оцінки ризику доступу, яка базується на розрахунку інтегрального показника довіри.

3 ПРОЄКТУВАННЯ МОДИФІКОВАНОГО ПРОТОКОЛУ АВТЕНТИФІКАЦІЇ

Спираючись на теоретичний базис Zero trust та математичну модель оцінки ризиків, розроблену у попередньому розділі, буде проведено проєктування архітектури модифікованого протоколу автентифікації. Запропоноване рішення є інтелектуальною надбудовою над стандартним протоколом OpenID Connect (OIDC). Основна мета модифікації полягає у переході від статичної перевірки паролів до динамічного управління доступом, де рішення про видачу електронного ключа (токена) приймається на основі поточного рівня довіри до користувача.

3.1 Архітектура системи «Адаптивний шлюз Нульової довіри»

Традиційні системи автентифікації працюють за лінійним сценарієм: клієнт надає облікові дані, а сервер перевіряє їх наявність у базі. У разі успіху видається маркер доступу з фіксованим терміном дії. Головний недолік такого підходу – неможливість врахувати контекст безпеки (звідки прийшов запит, з якого пристрою, чи є поведінка типовою).

Для усунення цього недоліку пропонується архітектура Adaptive Zero trust gateway (Адаптивний шлюз Нульової довіри),, що розділяє процес на дві площини: площину даних (де проходить трафік) та площину управління (де приймаються рішення). Система складається з чотирьох взаємопов'язаних компонентів (рис. 3.1).

1. Точка виконання політик (Policy enforcement point – PEP). Цю роль виконує шлюз API (API Gateway). Він виступає бар'єром між зовнішнім середовищем та внутрішніми мікросервісами. Функція PEP суто виконавча:

він не аналізує, чи є користувач «хорошим», він лише перевіряє наявність валідної перепустки. Якщо перепустка (токен) відсутня або прострочена, шлюз блокує запит.

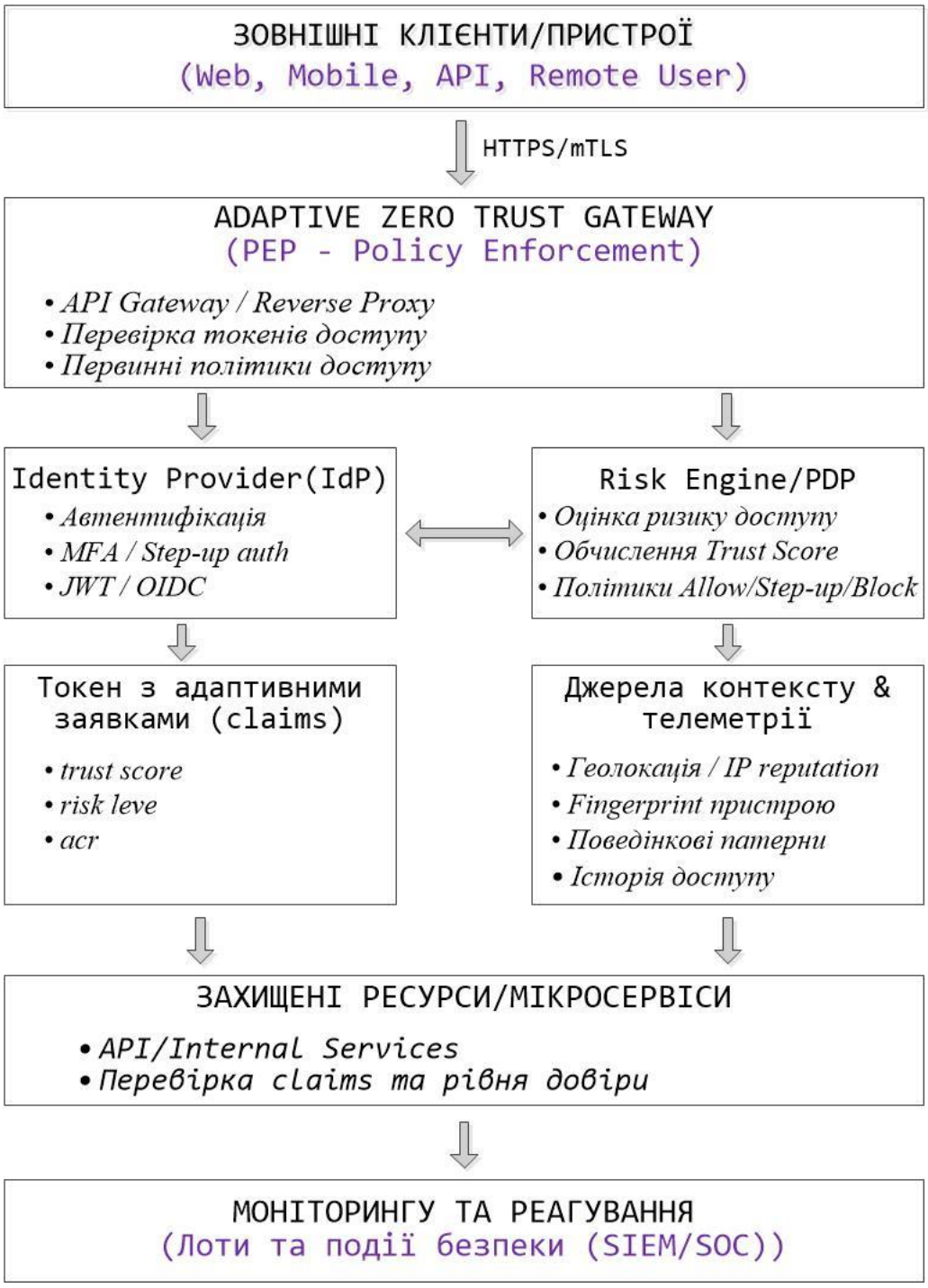


Рисунок 3.1 – Архітектуру Adaptive Zero trust gateway.

2. Сервер ідентифікації (identity provider – IdP). Компонент, відповідальний за зберігання облікових записів та перевірку паролів. У нашій архітектурі його роль змінюється: він більше не є єдиним центром прийняття рішень. Після перевірки пароля IdP не видає токен автоматично, а звертається за «консультацією» до аналітичного ядра.

3. Центр оцінки ризиків (Risk Engine / Policy decision point). Це центральний інтелектуальний вузол системи. Він отримує набір даних про поточну сесію, застосовує математичну модель оцінки ризиків і формує вердикт щодо рівня довіри (рис. 3.2). Саме цей компонент визначає, чи безпечно пускати користувача в систему.

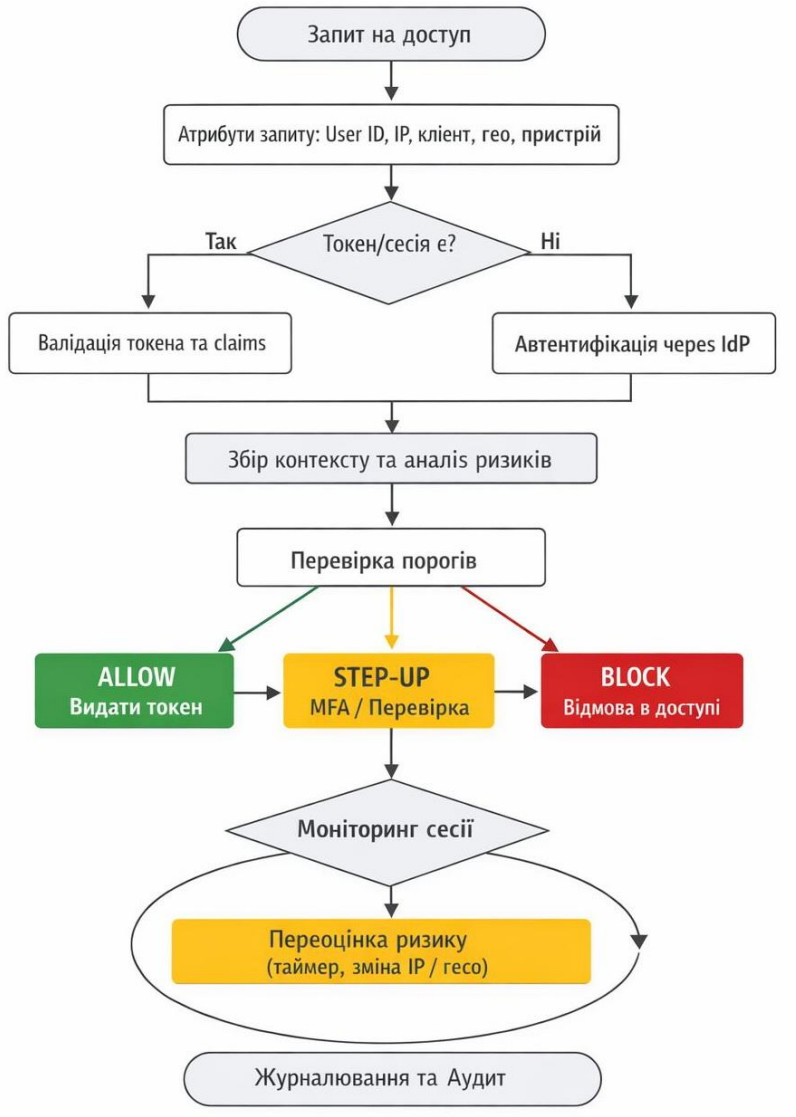


Рисунок 3.2 – Схема роботи Центру оцінки ризиків (Risk Engine).

4. Колектори контексту. Це програмні модулі, що збирають допоміжну інформацію: геолокацію IP-адреси, цифрові відбитки пристрою, історію попередніх входів та дані із зовнішніх баз загроз (Threat intelligence).

3.2 Алгоритм адаптивної видачі токенів

Ключовою відмінністю розробленого протоколу є логіка генерації токена доступу. Замість бінарного підходу («дозволити» або «заборонити»), впроваджується гранулярна система доступу, яка базується на розрахованому інтегральному показнику ризику.

Процес прийняття рішення можна описати як алгоритм розподілу користувачів по трьох «коридорах безпеки» (рис. 3.3).

1. Зелений коридор (низький ризик). Як3що розрахований рівень ризику не перевищує мінімального порогу (користувач входить зі звичного пристрою, з типової локації, у звичний час), система генерує стандартний токен доступу. Такий токен має повний термін дії (наприклад, 1 година) і не містить обмежень щодо функціоналу. Для користувача процес виглядає абсолютно прозорим і миттєвим.

2. Жовтий коридор (середній/невизначений ризик). Цей сценарій активується при виявленні аномалій, які не є критичними, але викликають підозру (наприклад, вхід з нового браузера або зміна міста). У цьому випадку система призупиняє видачу основного токена. Замість доступу ініціюється процедура Step-up Authentication (підвищення рівня автентифікації). Системі потрібні додаткові докази легітимності користувача. Це може бути запит на введення одноразового коду (one time password , OTP) з мобільного додатку або підтвердження через електронну пошту. Тільки після успішного проходження додаткової перевірки рівень ризику знижується, і користувач отримує доступ.

3. Червоний коридор (високий ризик). Якщо сукупність факторів вказує на явну загрозу (наприклад, неможлива швидкість переміщення між містами або IP-адреса з «чорного списку»), система діє превентивно. Процес входу блокується, токен не видається за жодних обставин, а інцидент безпеки реєструється для подальшого розслідування адміністратором.

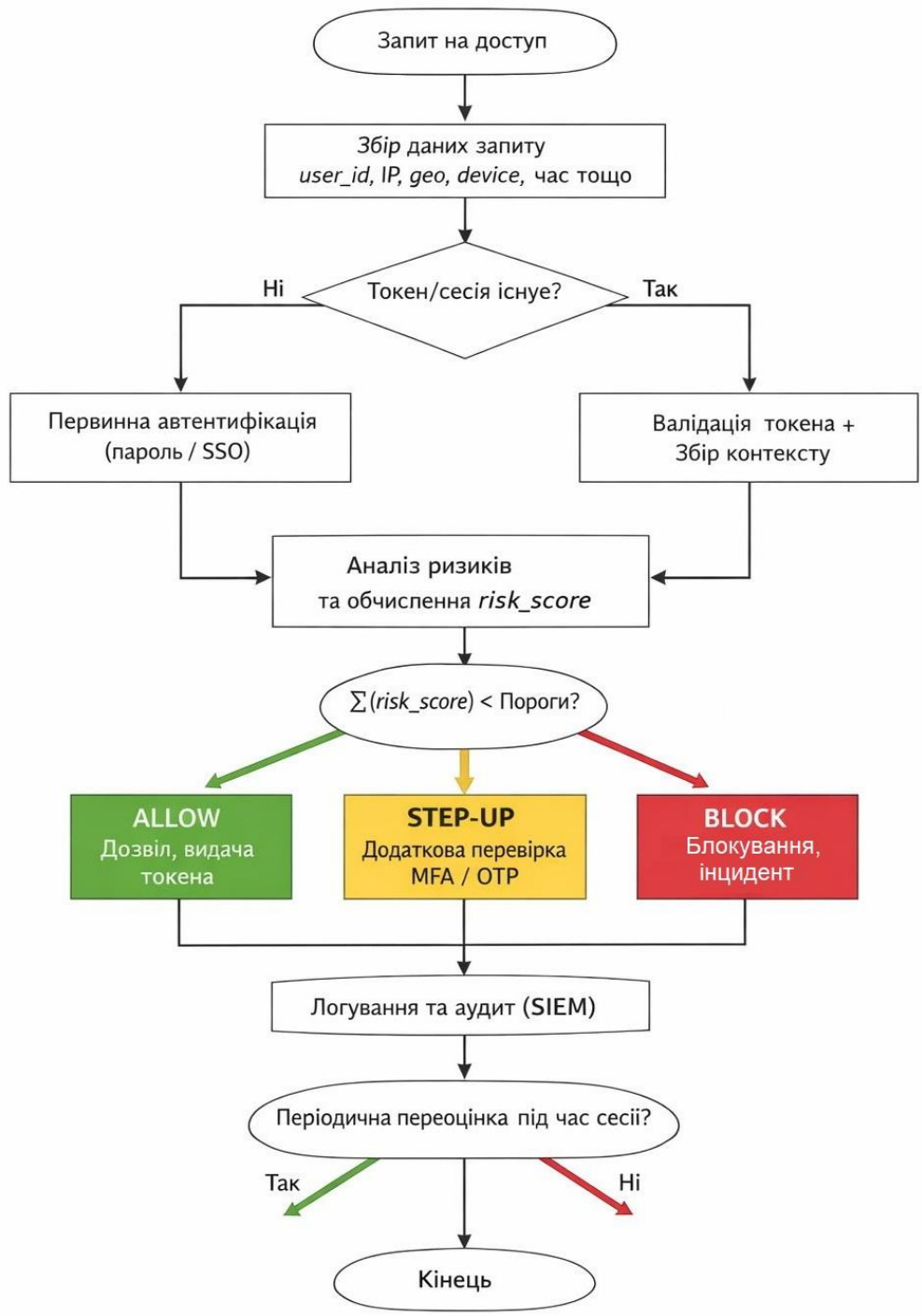


Рисунок 3.3 – Алгоритм адаптивної видачі токенів.

3.3 Семантична структура модифікованого токена

У розподілених системах токен (зазвичай формату JWT) слугує контейнером інформації, яку мікросервіси використовують для авторизації. У запропонованій моделі ми розширюємо інформаційне наповнення токена, перетворюючи його з простого посвідчення особи на «паспорт безпеки».

Замість стандартного набору полів, модифікований токен містить розширений набір заявок (claims), які описують контекст поточної сесії.

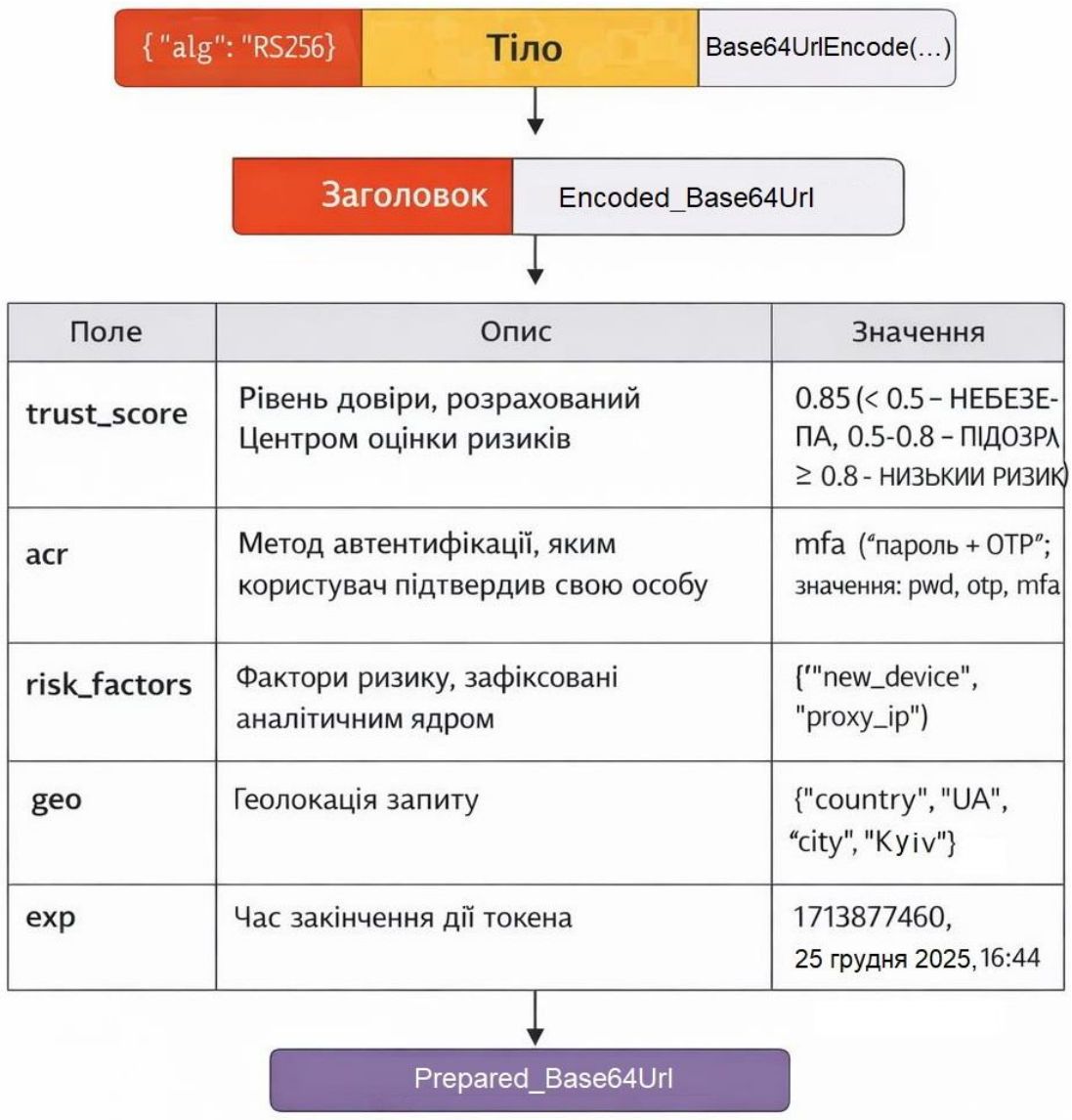
1. Рівень довіри (Trust score): числове значення, що відображає ймовірність того, що користувач є легітимним. Це дозволяє сервісам приймати автономні рішення. Наприклад, сервіс перегляду профілю може пускати користувача з низьким рівнем довіри, тоді як сервіс фінансових переказів вимагатиме високого рівня.

2. Контекст автентифікації (authentication context recognition, ACR): поле, що вказує, яким саме методом було перевірено користувача (тільки пароль, пароль + SMS, біометрія).

3. Фактори ризику: перелік виявлених відхилень (наприклад, "new_device", "proxy_ip"). Це дає можливість сервісам розуміти природу підозри.

4. Географічна мітка: прив'язка сесії до конкретного регіону, що дозволяє реалізувати геофенсинг (geofencing – обмеження доступу за географічною ознакою) на рівні окремих ресурсів.

Така структура (рис. 3.4) дозволяє реалізувати принцип «Розумної авторизації» (Smart authorization), де права доступу залежать не лише від ролі користувача (Адміністратор/Менеджер), а й від поточного стану його безпеки.



Поле	Опис	Значення
trust_score	Рівень довіри, розрахований Центром оцінки ризиків	0.85 (< 0.5 – НЕБЕЗЕ-ПА, 0.5-0.8 – ПІДОЗРЛ ≥ 0.8 - НИЗЬКИИ РИЗИК)
acr	Метод автентифікації, яким користувач підтвердив свою особу	mfa ("пароль + OTP"; значення: pwd, otp, mfa
risk_factors	Фактори ризику, зафіксовані аналітичним ядром	{'"new_device", "proxy_ip")
geo	Геолокація запиту	{"country", "UA", "city", "Kyiv"}
exp	Час закінчення дії токена	1713877460, 25 грудня 2025, 16:44

Рисунок 3.4 – Структура модифікованого токена JWT.

В додатку А наведений приклад (демо-версія) програмного коду JWT-токену на мові Python. Коректність роботи написаного програмного коду та згенерований токен було досліджено в хмарному середовищі Google Colab. Результати моделювання наведені на рис. 3.5.

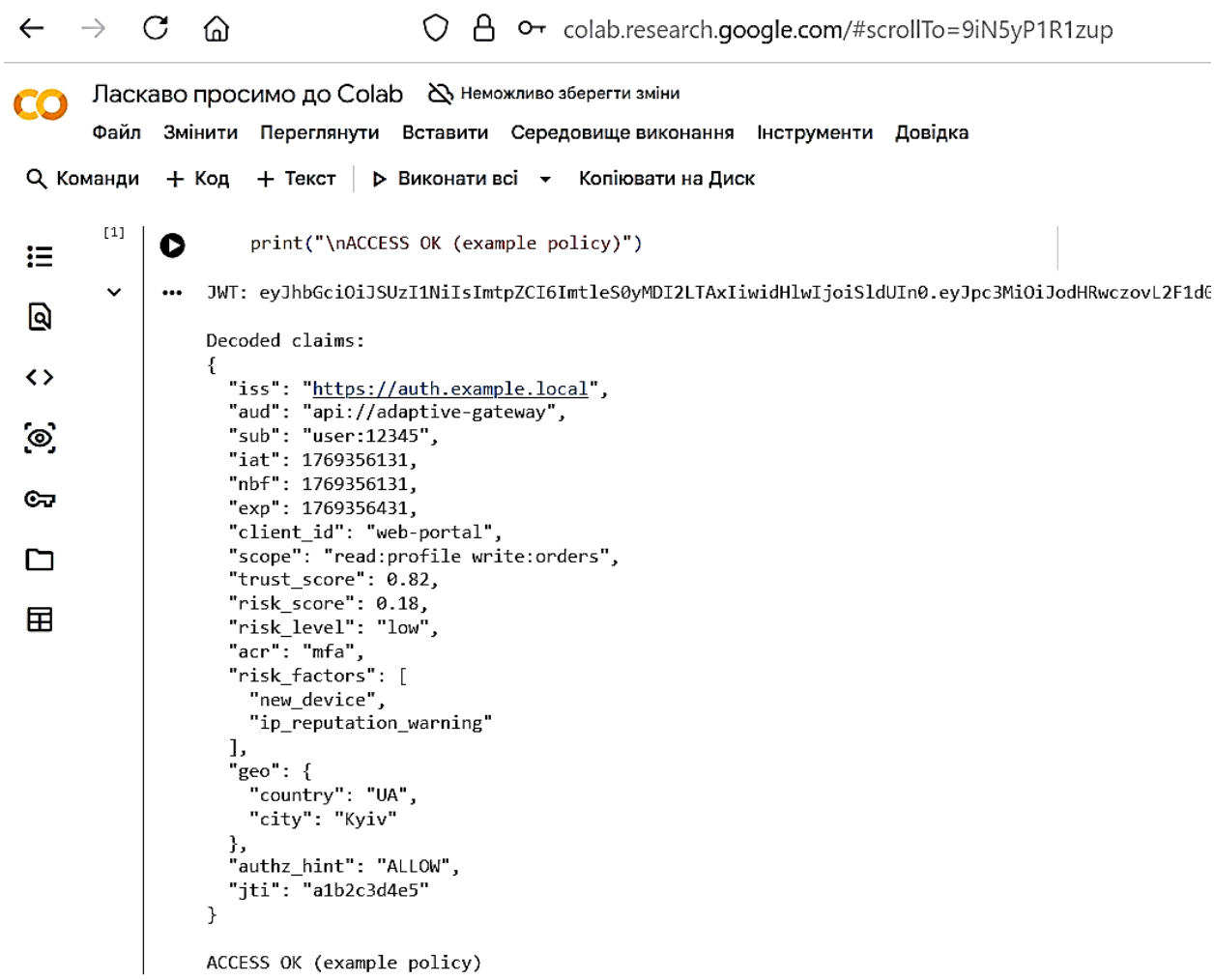


Рисунок 3.5 – Фрагмент JWT-токену в середовищі Google Colab.

3.4 Вибір технологічного стеку для реалізації

Для практичної реалізації спроєктованої архітектури доцільно використовувати набір інструментів з відкритим вихідним кодом, що забезпечують гнучкість налаштування та відповідність промисловим стандартам.

1. Сервер ідентифікації (Identity provider): рекомендовано використання платформи Keycloak. Її архітектура дозволяє впроваджувати власні програмні модулі (плагіни) безпосередньо в процес обробки входу

користувача. Це дозволяє інтегрувати логіку звернення до Risk engine без необхідності переписувати ядро системи автентифікації.

2. Шлюз API (Gateway): використання рішень на базі Kong або NGINX дозволяє винести перевірку цифрового підпису токенів на інфраструктурний рівень. Це розвантажує мікросервіси від криптографічних операцій та гарантує, що жоден запит не омине перевірку безпеки.

3. Управління секретами: для захисту криптографічних ключів, якими підписуються токени, використовується система HashiCorp Vault. Вона забезпечує динамічну ротацію ключів, що робить викрадення ключів зловмисниками практично неможливим, оскільки ключі постійно змінюються.

4. Аналітичний модуль: реалізація математичної моделі виконується на мові Python, яка має потужний математичний апарат та бібліотеки для роботи з даними, що є критичним для швидкого розрахунку ризиків.

3.5 Забезпечення відмовостійкості

Введення додаткового компонента (Risk Engine) створює потенційну точку відмови. Якщо сервіс оцінки ризиків стане недоступним (наприклад, через перевантаження), це не повинно зупинити роботу всієї системи.

Для цього передбачено механізм "м'якої відмови" (Fail-Safe). У разі недоступності аналітичного модуля система переходить у режим обмеженої функціональності. Замість блокування входу, система видає токени з мінімальним терміном дії та примусово вимагає від усіх користувачів проходження багатофакторної автентифікації (MFA). Це дозволяє зберегти доступ до сервісів, компенсуючи відсутність аналітики підвищеними вимогами до перевірки користувача.

3.6 Типові сценарії функціонування системи адаптивної автентифікації

В рамках проведеного дослідження було змодельовано та виконано чотири ключові сценарії, які демонструють адаптивну поведінку системи у відповідь на зміни контексту.

1. Сценарій №1: «Золотий шлях» (низький ризик).

Передумови: У базі даних накопичено історію входів користувача user_01. Типова локація – Київ (ISP: Ukrtelecom), типовий пристрій – Chrome 120 on Windows 10.

Дія: Емулюється вхід з тими ж параметрами контексту.

Результат Risk Engine: Всі перевірки пройдено. Розрахований показник довіри Trust Score = 0.95.

Реакція системи: Keycloak миттєво генерує Access Token з терміном дії 60 хвилин. У токені присутній claim acr: "level-1". Користувач отримує доступ до ресурсів без додаткових затримок.

Висновок: Система не створює перешкод для легітимних користувачів.

2. Сценарій №2: Детектування кінематичної аномалії (impossible travel).

Передумови: Зловмисник отримав доступ до валідних облікових даних user_02.

Хронологія подій:

T=10:00:00: Легітимний вхід користувача з IP-адреси 185.x.x.x (Київ). Сесія успішна.

T=10:05:00: Спроба входу з тим самим логіном з IP-адреси 212.x.x.x (Лондон).

Внутрішня логіка: Модуль перевірки геолокаційних даних обчислює дистанцію (~2400 км) та дельту часу (5 хвилин). Розрахована швидкість переміщення становить 28 800 км/год.

Реакція системи: Risk Engine повертає критичний вердикт BLOCK. Keycloak розриває з'єднання з кодом 403 Forbidden та повідомленням «Security

Alert: Impossible travel detected». Обліковий запис автоматично блокується на 30 хвилин.

Висновок: Атаку успішно відбито завдяки контекстному аналізу, незважаючи на правильний пароль.

3. Сценарій №3: Зміна контексту пристрою (Step-up Authentication)

Передумови: Легітимний користувач user_03 придбав новий ноутбук (зміна Device fingerprint), але знаходиться у звичній локації.

Дія: Спроба входу з нового пристрою.

Результат Risk Engine: Фактор геолокації – позитивний, фактор пристрою – негативний (unknown device). Інтегральний Trust score = 0.45 (потрапляє у діапазон для θ_{MFA}).

Реакція системи: Keycloak призупиняє видачу токена. Клієнт отримує редірект на сторінку введення OTP.

Завершення: Після успішного введення коду з Google Authenticator, Risk Engine оновлює профіль користувача, додаючи новий хеш пристрою до списку довірених. Наступні входи з цього ноутбука будуть проходити без MFA.

Висновок: Реалізовано механізм адаптивної безпеки, який вимагає додаткових доказів лише при підвищенні ризику.

4. Сценарій №4: Захист від повторного використання токена (Replay Attack)

Передумови: Зловмисник перехопив токен користувача, виданий для сесії з низьким рівнем довіри (наприклад, вхід з публічного Wi-Fi). Токен має заявлений показник довіри Trust_score: 0.3.

Дія: Зловмисник намагається використати цей токен для доступу до критичного мікросервісу Payment-Service.

Реакція системи: API Gateway пропускає запит (підпис валідний). Однак, мікросервіс Payment-Service має локальну політику: Require trust_score >= 0.8. Сервіс відхиляє транзакцію.

Висновок: Продемонстровано роботу принципу Zero trust на рівні ресурсів (Smart Authorization).

3.7 Висновки до розділу 3

У третьому розділі проведено проєктування та моделювання модифікованого протоколу автентифікації, який базується на концепції Zero trust та динамічній оцінці ризиків, а саме:

– розроблено архітектуру системи «Адаптивний шлюз Нульової довіри» (Adaptive Zero trust gateway), яка розділяє процес автентифікації на дві площини: площину виконання (Policy Enforcement Point на базі API Gateway) та площину прийняття рішень (Risk Engine). Це дозволило перейти від статичної перевірки паролів до безперервного аналізу контексту безпеки;

– впроваджено логіку розподілу користувачів по трьох «коридорах безпеки» (зелений, жовтий, червоний) залежно від розрахованого інтегрального показника ризику;

– модифіковано семантичну структуру токена доступу (JWT), який містить розширений набір заявок (claims), зокрема trust_score (рівень довіри), acr (контекст автентифікації) та risk_factors;

– розроблено механізм «м'якої відмови» (fail-safe), який гарантує, що у випадку недоступності аналітичного модуля (Risk Engine) система не припиняє роботу, а переходить у захищений режим з примусовим використанням багатофакторної автентифікації та скороченим терміном дії токенів;

– шляхом моделювання чотирьох ключових сценаріїв доведено здатність системи протидіяти типовим загрозам.

ВИСНОВКИ

У магістерській роботі вирішено актуальне науково-прикладне завдання підвищення рівня захищеності розподілених інформаційних систем (РІС) шляхом розробки та впровадження механізмів адаптивної автентифікації на основі концепції Zero Trust («Нульова довіра»). За результатами виконання роботи отримано наступні висновки та практичні результати:

1. Проведено аналіз архітектури та загроз сучасних РІС. Встановлено, що глобальна трансформація ІТ-інфраструктури (перехід до мікросервісів, хмарних технологій та IoT) призвела до деактуалізації традиційної моделі захисту периметра.

2. Обґрунтовано та розроблено концепцію адаптивної автентифікації. Запропоновано розширити класичну тріаду факторів автентифікації (знання, володіння, властивість) четвертим динамічним фактором – «Контекстом». Формалізовано векторний простір ознак контексту, який включає геопросторові дані, часові характеристики, цифрові відбитки пристроїв та поведінкову біометрію.

3. Створено математичну модель динамічної оцінки ризику. Розроблено алгоритм розрахунку інтегрального показника довіри (Trust Score), що базується на нормалізації різнорідних метрик та використанні логістичних функцій ризику.

4. Спроєктовано архітектуру системи «Адаптивний шлюз Нульової довіри». Запропоновано архітектурне рішення, що розділяє площину даних (API Gateway) та площину прийняття рішень (Risk Engine). Розроблено структуру модифікованого токена JWT, що містить дані про рівень довіри та фактори ризику.

5. Розроблено та перевірено сценарії функціонування системи. Реалізовано алгоритм адаптивної видачі токенів за принципом «коридорів безпеки»: зелений (повний доступ), жовтий (вимога додаткової MFA), червоний (блокування).

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. 2025 Data Breach Investigations Report. Verizon Business. URL: https://www.verizon.com/business/resources/reports/dbir (last accessed: 06.09.2025).

2. Золотар О. О. Інформаційна безпека людини: теорія і практика: монографія. Київ: ТОВ «Видавничий дім «АртЕк», 2018. 446 с.

3. Лісовська Ю. П. Кібербезпека: ризики та заходи: навч. посіб. Київ: Кондор, 2019. 272 с.

4. Остапов С. Е., Євсеєв С. П., Король О. Г. Кібербезпека: сучасні технології захисту: навч. посіб. Львів: Новий Світ-2000, 2020. 678 с.

5. Смірнов О. А. та ін. Основи безпеки в комп'ютерних мережах: навч. посіб. Кропивницький: ПП «Лисенко В. Ф.», 2018. 177 с.

6. Тарнавський Ю. А. Технології захисту інформації. Київ: КПІ ім. Ігоря Сікорського, 2018. 162 с.

7. Бурячок В. Л. та ін. Інформаційний та кіберпростори: проблеми безпеки, методи та засоби боротьби: посібник. Київ: ДУТ, КНУ, 2016. 178 с.

8. Бурячок В. Л. Модель формування дерева атак для одержання інформації в інформаційно-телекомунікаційних системах і мережах при вилученому доступі. *Інформатика та математичні методи в моделюванні*. 2013. № 2. С. 123–131.

9. ELHejazi M. F., Muragaa W. H. A. Improving the security and reliability of SDN controller REST APIs using JSON Web Token (JWT) with OpenID and Auth2.0. *2024 IEEE 4th International Maghreb Meeting of the Conference on Sciences and Techniques of Automatic Control and Computer Engineering (MI-STA)*, Tripoli, Libya, 19–21 May 2024. IEEE, 2024. URL: https://doi.org/10.1109/mi-sta61267.2024.10599643SSO.

10. Що таке розподілена система? URL: https://www.solarwinds.com/blog/what-is-a-distributed-system (дата звернення: 06.09.2025).

11. Architecture of Distributed System. URL: https://medium.com/@skchamith14/architecture-of-distributed-system-b07ee95755f (last accessed: 06.09.2025).

12. An Overview of Distributed Systems Architectures. URL: https://www.learncsdesign.com/overview-of-distributed-systems-architectures/ (last accessed: 06.09.2025).

13. Understanding Distributed Systems: A Comprehensive Deep Dive for Modern Developers. URL: https://jinlow.medium.com/understanding-distributed-systems-a-comprehensive-deep-dive-for-modern-developers-9c11b08d28f0 (last accessed: 06.09.2025).

14. What Is a Distributed System? URL: https://www.solarwinds.com/blog/what-is-a-distributed-system (last accessed: 07.09.2025).

15. Microservices Architectures. URL: https://blog.nashtechglobal.com/microservices-architecture/ (last accessed: 07.09.2025).

16. Guide to Cloud Computing Architectures. URL: https://www.networkcomputing.com/cloud-networking/guide-to-cloud-computing-architectures./ (last accessed: 07.09.2025).

17. IaaS, PaaS, SaaS: яку хмарну модель обрати бізнесу. URL: https://hub.kyivstar.ua/articles/iaas-paas-saas-yaku-hmarnu-model-obraty-biznesu (last accessed: 07.09.2025).

18. Як працює IoT зсередини: сенсори, протоколи, обробка даних. URL: https://robotdreams.cc/uk/blog/772-yak-pracyuye-iot-zseredyny-sensory-protokoly-obrobka-danyh (last accessed: 07.09.2025).

19. Seo J. The future of digital authentication: blockchain-driven decentralized authentication in Web 3.0. *J. Web Eng.* 2024. P. 611–636. URL: https://doi.org/10.13052/jwe1540-9589.2351.

20. Benjamin N. How Effective Is Blockchain in Cybersecurity? *ISACA Journal*. 2021. No. 4.

21. Moosavi N., Taherdoost H. Blockchain Technology Application in Security: A Systematic Review. *Blockchains*. 2023. Vol. 1, no. 2. P. 58–72. URL: https://doi.org/10.3390/blockchains1020005.

22. Mahmood S., Chadhar M., Firmin S. Cybersecurity Challenges in Blockchain Technology: A Scoping Review. *Human Behavior and Emerging Technologies*. 2022. P. 1–11. URL: https://doi.org/10.1155/2022/7384000.

23. Zwitter A. J., Gstrein O. J., Yap E. Digital Identity and the Blockchain: Universal Identity Management and the Concept of the "Self-Sovereign" Individual. *Front. Blockchain*. 2020. Vol. 3. URL: https://doi.org/10.3389/fbloc.2020.00026.

24. Feng Y., Zhong Z., Sun X., Wang L., Lu Y., Zhu Y. Blockchain enabled zero trust-based authentication scheme for railway communication networks. *Journal of Cloud Computing*. 2023. Vol. 12. URL: https://doi.org/10.1186/s13677-023-00411-z.

25. Ao W., Fu S., Zhang C., Huang Y., Xia F. A Secure Identity Authentication Scheme Based on Blockchain and Identity-based Cryptography. *IEEE 2nd International Conference on Computer and Communication Engineering Technology (CCET)*. 2019. P. 90–95.

26. MultiChain: Enterprise blockchain platform. URL: https://www.multichain.com/ (last accessed: 10.09.2025).

27. Sultan K., Ruhi U., Lakhani R. Conceptualizing Blockchains: Characteristics and Applications. *11th IADIS International Conference on Information Systems*. 2018. P. 49–57.

28. Poberezhnyk V., Opirskyy I. Developing of Blockchain Method in Message Interchange Systems. *Workshop on Cybersecurity Providing in Information and Telecommunication Systems*. 2023. Vol. 3421. P. 148–157.

29. Poberezhnyk V., Balatska V., Opirskyy I. Development of the Learning Management System Concept based on Blockchain Technology. *Workshop on Cybersecurity Providing in Information and Telecommunication Systems II*. 2023. Vol. 3550. P. 143–156.

30. Розбираємо Kubernetes зрозумілою мовою. URL: https://dou.ua/forums/topic/55156/ (дата звернення 08.09.2025).

31. Bucko A. et al. Enhancing JWT authentication and authorization in Web applications based on user behavior history. *Computers*. 2023. Vol. 12, no. 4. 78. URL: https://doi.org/10.3390/computers12040078.

32. Dimitrijevic N. et al. Advanced security mechanisms in the Spring framework: JWT, oauth, LDAP and keycloak. *Fourteenth Int. Conf. Bus. Inf. Secur. (BISEC'2023)*. 2024. Vol. 3676. P. 64–70. URL: https://ceur-ws.org/Vol-3676/short_09.pdf.

33. Xu B. et al. JWTKey: automatic cryptographic vulnerability detection in JWT applications. *Computer security – ESORICS 2023*. Cham: Springer Nature Switzerland, 2024. P. 263–282. URL: https://doi.org/10.1007/978-3-031-51479-1_14.

34. What is mutual TLS (mTLS)? URL: https://www.cloudflare.com/learning/ access-management/what-is-mutual-tls/ (last accessed: 10.09.2025).

35. Authentication vulnerabilities. URL: https://portswigger.net/web-security/authentication (last accessed: 10.09.2025).

36. Остапов С. Е., Євсеєв С. П., Король О. Г. Технології захисту інформації: навч. посіб. Харків: ХНЕУ, 2013. 476 с.

37. Гулак Г. М. та ін. Основи криптографічного захисту інформації: підручник. Вінниця: ВНТУ, 2011. 199 с.

38. IBM Security X-Force Threat Intelligence Index 2024. IBM. https://www.ibm.com/reports/threat-intelligence (last accessed: 10.09.2025).

39. Innocenti, T., et al.: OAuth 2.0 redirect URI validation falls short, literally. In: ACSAC '23: annual computer security applications conference, Austin TX USA. ACM, New York, NY, USA (2023). https://doi.org/10.1145/3627106.3627140.

40. Lakhno, V., et al.: A model developed for teaching an adaptive system of recognising cyberattacks among non-uniform queries in information systems.

Eastern-European J. Enterp. Technol. 4(9(82)), 27 (2016). https://doi.org/10.15587/1729-4061.2016.73315

41. Navas, J., Beltrán, M.: Understanding and mitigating OpenID Connect threats. Comput. & Secur. 2019. vol. 84, PP. 1–16 (). https://doi.org/10.1016/j.cose.2019.03.003.

42. Primbs, J., Menth, M.: OIDC2: open identity certification with OpenID connect. IEEE Open J. Commun. Soc. 1 (2024). https://doi.org/10.1109/ojcoms.2024.3376193

43. Rushdy, E., Khedr, W., Salah, N.: Framework to secure the OAuth 2.0 and JSON Web Token for REST API. J. Theor. Appl. Inf. Technol. 2021. № 99 (9). PP. 2144–2161.

44. Banerjee S., Woodard D. L. Biometric authentication and identification using keystroke dynamics: A survey. *Journal of Pattern Recognition Research*. 2012. Vol. 7, no. 1. P. 116–139.

45. Chen F. et al. Behavioral biometrics for continuous authentication in the Internet of Things era: An overview. *IEEE Internet of Things Journal*. 2020. Vol. 7, no. 8. P. 7118–7131. URL: https://doi.org/10.1109/JIOT.2020.2975332.

46. Mondal S., Bours P. A study on continuous authentication using a combination of keystroke and mouse biometrics. *Neurocomputing*. 2015. Vol. 230. P. 1–22. URL: https://doi.org/10.1016/j.neucom.2016.12.007.

47. Traore I., Ahmed A., Woungang I. Behavioral biometrics for continuous and transparent authentication. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*. 2013. Vol. 43, no. 3. P. 531–546. URL: https://doi.org/10.1109/TSMC.2012.2218613.

48. Yang Z., Hu Y., Yu Z., Wang Y., Li J. Multi-modal behavioral biometrics authentication based on deep learning. *IEEE Access*. 2020. Vol. 8. P. 24690–24700. URL: https://doi.org/10.1109/ACCESS.2020.2969195.

49. Fridman L., Weber S., Greenstadt R., Kam M. Active authentication on mobile devices via stylometry, application usage, web browsing, and GPS location.

IEEE Systems Journal. 2015. Vol. 11, no. 2. P. 513–521. URL: https://doi.org/10.1109/JSYST.2015.2453215.

50. Morales A., Fierrez J., Ortega-Garcia J. Keystroke dynamics recognition based on personal data: A comparative experimental evaluation. *Pattern Recognition Letters*. 2016. Vol. 79. P. 20–27. URL: https://doi.org/10.1016/j.patrec.2016.03.007.

51. Shen C., Cai Z., Guan X., Du X. A privacy-preserving protocol for secure and efficient user authentication. *Computer Standards & Interfaces*. 2013. Vol. 35, no. 2. P. 240–246.

52. Shen C., Guan X., Cai Z. Continuous authentication for mouse dynamics: A pattern-growth approach. *International Journal of Information Security*. 2011. Vol. 10. P. 221–229. URL: https://doi.org/10.1007/s10207-011-0120-2.

53. Tiwari A., Gupta A. A hybrid model for user authentication using keystroke and mouse dynamics with machine learning. *Journal of Intelligent & Fuzzy Systems*. 2022. Vol. 42, no. 2. P. 1283–1296. URL: https://doi.org/10.3233/JIFS-210883.

54. Тітова В., Кльоц Ю., Пирч О., Шемчук У., & Божок Д. (2024). Аналіз сучасних методів автентифікації користувачів. *Herald of Khmelnytskyi National University. Technical Sciences*, 2024. 345(6(2). P. 234-237. https://doi.org/10.31891/2307-5732-2024-345-6-35.

55. Zero Trust Design: Secure API Architecture. URL: https://www.krakend.io/docs/design/zero-trust/?gad_source=1&gad_campaignid=23444938753&gclid=EAIaIQobChMIrs_R_NyWkgMVI9JEBx3ZnCB1EAMYASAAEgKvqfD_BwE (last accessed: 10.09.2025).

56. Zero Trust: Модель кібербезпеки, яка не вірить нікому — і саме тому рятує бізнес. URL: https://my-itspecialist.com/zero-trust-model-kyberbezpeky (дата звернення 11.09.2025).

57. Risk in focus 2026. URL: https://www.theiia.org/globalassets/site/foundation/latest-research-and-products/risk-in-focus/2026/2026-global-report-en-riskinfocus.pdf (last accessed: 10.09.2025).

58. NIST Risk Management Framework. URL: https://csrc.nist.gov/projects/risk-management/about-rmf (last accessed: 21.09.2025).

59. Haversine formula. URL: https://en.wikipedia.org/wiki/Haversine_formula (last accessed: 10.09.2025).

60. Mahalanobis distance. URL: https://en.wikipedia.org/wiki/Mahalanobis_distance (last accessed: 10.09.2025).

ДОДАТОК А

Програмний код токена JWT на мові Python

```
"""
Adaptive JWT (modified claims) — issue + verify
Requirements: pip install pyjwt cryptography
"""

from __future__ import annotations
import time
import json
from typing import Any, Dict, List, Optional
import jwt  # PyJWT
from cryptography.hazmat.primitives.asymmetric import rsa
from cryptography.hazmat.primitives import serialization

# -----------------
# Key management (demo only)
# -----------------

def generate_rsa_keypair() -> tuple[bytes, bytes]:

    """
    Generates an RSA keypair for RS256 signing.
    Returns (private_pem, public_pem) as bytes.
    In production: store private key in Vault/HSM and rotate keys (kid).
    """

    private_key = rsa.generate_private_key(public_exponent=65537,
key_size=2048)
    private_pem = private_key.private_bytes(
        encoding=serialization.Encoding.PEM,
        format=serialization.PrivateFormat.PKCS8,
        encryption_algorithm=serialization.NoEncryption(),

    )

    public_pem = private_key.public_key().public_bytes(

        encoding=serialization.Encoding.PEM,

        format=serialization.PublicFormat.SubjectPublicKeyInfo,
```

```
    )

    return private_pem, public_pem

# ----------------------------
# Risk Engine output model
# ----------------------------

def compute_trust_score(risk_score: float) -> float:

    """
    Example mapping: higher risk => lower trust.
    Clamp to [0,1]. Replace with your PDP model/weights.
    """

    trust = 1.0 - risk_score
    return max(0.0, min(1.0, trust))

def decide_action(trust_score: float, t1: float = 0.8, t2: float = 0.5) -> str:

    """
    Example policy:
      trust >= 0.8  => ALLOW
      0.5..0.8      => STEP_UP
      < 0.5         => BLOCK
    """
    if trust_score >= t1:

        return "ALLOW"

    if trust_score >= t2:

        return "STEP_UP"

    return "BLOCK"

# ----------------------------
# JWT issuing (IdP / Token Service)
# ----------------------------

def issue_adaptive_jwt(
    *,
    private_pem: bytes,
    kid: str,
    issuer: str,
```

```
    audience: str,
    subject: str,
    client_id: str,
    scope: str,
    risk_score: float,
    acr: str,
    risk_factors: List[str],
    geo: Dict[str, str],
    ttl_seconds: int = 300,
    jti: Optional[str] = None,
) -> str:

    """
    Issues a signed JWT (RS256) with adaptive claims:
      - trust_score
      - risk_level
      - acr
      - risk_factors
      - geo
    """

    now = int(time.time())
    trust_score = compute_trust_score(risk_score)
    decision = decide_action(trust_score)

    # Risk level is often a small categorical label used by services/policies

    if decision == "ALLOW":
        risk_level = "low"
    elif decision == "STEP_UP":
        risk_level = "medium"
    else:
        risk_level = "high"

    payload: Dict[str, Any] = {

        # Standard claims

        "iss": issuer,
        "aud": audience,
        "sub": subject,
        "iat": now,
        "nbf": now,
        "exp": now + ttl_seconds,
```

```
        # Optional identifiers

        "client_id": client_id,
        "scope": scope,

        # Adaptive / modified claims

        "trust_score": round(trust_score, 3),      # e.g., 0.853
        "risk_score": round(risk_score, 3),        # e.g., 0.147
        "risk_level": risk_level,                  # low|medium|high
        "acr": acr,                                # pwd|otp|mfa (OIDC ACR concept)
        "risk_factors": risk_factors,              # ["new_device", "proxy_ip"]
        "geo": geo,                                # {"country":"UA","city":"Kyiv"}

        # Decision hint (optional). Services can also recompute from trust_score.

        "authz_hint": decision,                    # ALLOW|STEP_UP|BLOCK

    }

    if jti:
        payload["jti"] = jti

    headers = {"kid": kid, "typ": "JWT", "alg": "RS256"}
    token = jwt.encode(
        payload=payload,
        key=private_pem,
        algorithm="RS256",
        headers=headers,
    )

    return token

# ----------------------------
# JWT verification (PEP / API Gateway / resource server)
# ----------------------------

def verify_adaptive_jwt(
    *,
    token: str,
    public_pem: bytes,
    issuer: str,
    audience: str,
    leeway_seconds: int = 10,
) -> Dict[str, Any]:
```

```
    """
    Verifies signature + standard constraints. Returns decoded claims.
    """

    decoded = jwt.decode(
        jwt=token,
        key=public_pem,
        algorithms=["RS256"],
        issuer=issuer,
        audience=audience,
        leeway=leeway_seconds,
        options={

            "require": ["exp", "iat", "nbf", "iss", "aud", "sub"],
        },
    )

    return decoded

# ----------------------------
# Demo run
# ----------------------------

if __name__ == "__main__":
    private_pem, public_pem = generate_rsa_keypair()

    # Example inputs (imitate what Risk Engine would provide)

    risk_score = 0.18  # 0..1 (example)
    risk_factors = ["new_device", "ip_reputation_warning"]
    geo = {"country": "UA", "city": "Kyiv"}

    token = issue_adaptive_jwt(

        private_pem=private_pem,
        kid="key-2026-01",
        issuer="https://auth.example.local",
        audience="api://adaptive-gateway",
        subject="user:12345",
        client_id="web-portal",
        scope="read:profile write:orders",
        risk_score=risk_score,
        acr="mfa",              # pwd|otp|mfa
        risk_factors=risk_factors,
```

```
    geo=geo,
    ttl_seconds=300,          # short-lived token
    jti="a1b2c3d4e5",
)

print("JWT:", token, "\n")

claims = verify_adaptive_jwt(

    token=token,
    public_pem=public_pem,
    issuer="https://auth.example.local",
    audience="api://adaptive-gateway",
)
print("Decoded claims:")
print(json.dumps(claims, ensure_ascii=False, indent=2))

# Example resource-side check

trust = float(claims.get("trust_score", 0.0))
if trust < 0.5:
    raise SystemExit("ACCESS DENIED: trust_score below threshold")
print("\nACCESS OK (example policy)")
```