УДК 004.056:004.738.5:004.4:614

В.М. СЕМЧИШИН, Д.М. МИХАЛИК
Тернопільський національний технічний університет імені Івана Пулюя

МОДЕЛЬ КОМПЛЕКСНОЇ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ ХМАРНОЇ СИСТЕМИ УПРАВЛІННЯ МЕДИЧНИМИ ПОСЛУГАМИ

Досліджено комплексну проблему забезпечення інформаційної безпеки у сучасних хмарних медичних інформаційних системах, які реалізують алгоритмічне управління медичними послугами на основі аналізу даних та автоматизованих процедур ухвалення рішень. Показано, що зростання обсягів медичних даних, активне впровадження телемедицини, мобільних сервісів та інтегрованих електронних медичних записів потребують створення захищених, масштабованих і стійких до кібератак архітектур. Обґрунтовано, що традиційні підходи до кіберзахисту не забезпечують належного рівня стійкості в умовах динамічного навантаження та наявності інсайдерських ризиків, а також не враховують особливостей алгоритмічного управління, де критичне значення має цілісність параметрів прийняття рішень.

Запропоновано модель інтеграції механізмів безпеки безпосередньо в логіку прийняття рішень, побудовану на методі аналізу ієрархій (AHP). Обґрунтовано класифікацію медичних даних за рівнем критичності з урахуванням їхнього впливу на результат клінічних та адміністративних рішень. Ідентифіковано ключові загрози для хмарних медичних систем, зокрема модифікацію ваг AHP, replay-атаки, компрометацію API, несанкціоноване використання кешу, підміну запитів до бази даних, витік облікових даних і спотворення журналів подій. На основі цього сформовано систему принципів побудови безпечної архітектури, яка включає рольову модель контролю доступу (RBAC), багаторівневе журналювання, криптографічний захист даних за алгоритмом AES-256, контроль цілісності, обмеження часу життя кешованих даних, а також впровадження систем моніторингу та автоматичного реагування на інциденти.

Подано приклади реалізації захисних механізмів у середовищах Redis та PostgreSQL, що демонструють практичну можливість застосування розробленої моделі у хмарній інфраструктурі з високими вимогами до масштабованості. Показано, що інтеграція безпеки на рівні алгоритмів ухвалення рішень забезпечує підвищення довіри до автоматизованих систем, гарантує цілісність та конфіденційність медичної інформації та суттєво знижує ризики інсайдерських і зовнішніх атак. Запропонована концепція може слугувати основою для створення нових поколінь безпечних хмарних рішень у сфері охорони здоров'я та подальшого розвитку нормативних вимог до захисту медичних даних.

Ключові слова: *інформаційна безпека, хмарні технології, управління медичними послугами, метод аналізу ієрархій (AHP), криптографічний захист, контроль доступу (RBAC), цілісність даних, масштабованість, система моніторингу, захист персональних даних.*

V.M. SEMCHYSHYN, D.M. MYHALYK
Ternopil Ivan Puluj National Technical University

COMPREHENSIVE INFORMATION SECURITY MODEL OF A CLOUD MEDICAL MANAGEMENT SYSTEM

A comprehensive problem of ensuring information security in modern cloud-based medical information systems implementing algorithmic management of medical services based on data analysis and automated decision-making procedures has been investigated. It is shown that the growing volume of medical data, the active adoption of telemedicine, mobile services, and integrated electronic health records require the development of secure, scalable, and cyberattack-resistant architectures. It is substantiated that traditional cybersecurity approaches do not provide an adequate level of resilience under dynamic workloads and insider threats, nor do they account for the specific features of algorithmic management, where the integrity of decision-making parameters is critical.

A model for integrating security mechanisms directly into the decision-making logic, built on the Analytic Hierarchy Process (AHP), is proposed. A classification of medical data based on criticality, considering their influence on clinical and administrative decision outcomes, is justified. Key threats to cloud-based medical systems are identified, including modification of AHP weights, replay attacks, API compromise, unauthorized cache access, database query substitution, credential leakage, and event log tampering. Based on these findings, a set of principles for designing a secure architecture is formulated, incorporating a role-based access control (RBAC) model, multi-level logging, AES-256 cryptographic data protection, integrity monitoring, cache lifetime restrictions, as well as the deployment of monitoring and automated incident response systems.

Examples of implementing security mechanisms in Redis and PostgreSQL environments are provided, demonstrating the practical feasibility of applying the proposed model in cloud infrastructures with high scalability requirements. It

https://doi.org/10.32782/mathematical-modelling/2025-8-2-26 ISSN 2618-0332 (print), 2618-0340 (online)

is shown that integrating security at the level of decision-making algorithms increases trust in automated systems, ensures the integrity and confidentiality of medical information, and significantly reduces the risks of insider and external attacks. The proposed concept may serve as a foundation for developing new generations of secure cloud solutions in healthcare and for further advancing regulatory requirements for the protection of medical data..

Key words: *information security, cloud technologies, medical service management, Analytic Hierarchy Process (AHP), cryptographic protection, access control (RBAC), data integrity, scalability, monitoring system, personal data protection.*

Постановка проблеми

Сучасні системи управління медичними послугами активно переходять до використання хмарних технологій, що забезпечують масштабованість, мобільність доступу та централізоване зберігання медичних даних. Проте така трансформація супроводжується зростанням ризиків, пов'язаних із витоком персональних даних, несанкціонованим доступом до медичних записів, модифікацією аналітичних алгоритмів і порушенням цілісності інформаційних потоків.

Особливу небезпеку становлять складні кібератаки, орієнтовані на компрометацію систем ідентифікації користувачів, API-з'єднань та механізмів кешування даних.

Традиційні підходи до захисту інформації, що базуються лише на криптографічних методах або автентифікації користувачів, виявляються недостатніми для хмарних середовищ, де безпека має бути інтегрована безпосередньо в логіку управління та процес прийняття рішень. Відсутність комплексних моделей, які одночасно враховують аспекти інформаційної безпеки, масштабованості та стійкості до внутрішніх і зовнішніх загроз, ускладнює створення довірених медичних інформаційних систем.

Отже, актуальною науковою проблемою є розроблення моделі комплексної інформаційної безпеки хмарної системи управління медичними послугами, що поєднує механізми багаторівневого захисту даних, контроль доступу, моніторинг подій безпеки та аналітичну оцінку ризиків на основі методів ієрархічного аналізу (AHP).

Аналіз останніх досліджень і публікацій

У сучасних умовах цифровізації охорони здоров'я використання хмарних технологій забезпечує новий рівень інтеграції, масштабованості та оперативності управління медичними послугами. Узагальнення результатів досліджень, представлених у праці [1], показує, що впровадження хмарних обчислень у медичних системах сприяє підвищенню ефективності обміну даними між лікувальними закладами, але водночас вимагає вирішення проблем конфіденційності та цілісності інформації.

І. Гао, А. Суняєв [2] доводять, що контекстні чинники, такі як нормативно-правова база, довіра до постачальників послуг і внутрішня організаційна культура, є визначальними під час ухвалення рішень про перехід до хмарних технологій у сфері охорони здоров'я.

У дослідженні М. Путцієр та ін. [3] розглянуто практичний досвід Німеччини щодо впровадження хмарних рішень у національній системі охорони здоров'я, зокрема використання безпечних API для обміну медичними даними.

Проблемам інформаційної безпеки у хмарних середовищах присвячено роботи М.А. Аль-Ісса, Ю. Оттом [4], які запропонували багаторівневу архітектуру захисту даних, що включає контроль доступу, автентифікацію та шифрування. Аналогічно А. Саджид, Н. Аббас [5] виділяють основні виклики захисту персональних медичних даних у хмарних системах і визначають напрями їх подальшого вдосконалення.

У роботі Н. Аль-Дмур, М. Аль-Дмур, Р. Шаннак [6] обґрунтовано, що використання хмарних технологій у медичних організаціях безпосередньо впливає на підвищення якості медичних послуг, зокрема за рахунок швидшої обробки даних та покращеної комунікації між лікарями й пацієнтами.

Згідно з М. Джаваїд та ін. [7; 8], подальший розвиток хмарних систем у медицині визначатиметься здатністю таких систем інтегрувати аналітичні інструменти, big data та штучний інтелект, при цьому безпека і масштабованість залишаються критичними аспектами.

Дослідження Y. Чентхара та ін. [9] деталізує проблематику забезпечення конфіденційності та цілісності даних у хмарних медичних рішеннях, підкреслюючи важливість моделей безпеки, орієнтованих на доступ на основі ролей (RBAC) та криптографічний захист (AES-256).

Класичні підходи до прийняття рішень на основі аналітичних моделей, зокрема метод аналізу ієрархій (AHP), висвітлено в роботах J. Долан [10] та Т. Сааті [11]. Ці праці стали теоретичною основою для побудови моделей алгоритмічного управління медичними процесами з урахуванням ризиків і пріоритетів безпеки даних.

Таким чином, проведений аналіз показує, що, попри значні успіхи у впровадженні хмарних технологій у медичну сферу, залишаються актуальними питання комплексного забезпечення інформаційної безпеки, масштабованості архітектур та інтеграції захисних механізмів у процеси прийняття управлінських рішень.

Мета дослідження

Метою дослідження є розроблення моделі комплексної інформаційної безпеки для хмарної системи управління медичними послугами, яка забезпечує захист персональних даних, безперервність функціонування сервісів та відповідність вимогам міжнародних стандартів безпеки.

Виклад основного матеріалу дослідження

Основні поняття, визначення і алгоритми

З метою побудови ефективної моделі безпеки запропоновано класифікувати інформацію, яка циркулює в межах системи, на кілька категорій. Така класифікація дає змогу чітко розмежувати об’єкти захисту, оцінити їх критичність і визначити відповідні заходи (Табл. 1). Ми виділяємо такі практичні вектори атак у межах нашої системи:

1. Модифікація ваг AHP призводить до зміни логіки пріоритету, може бути використана для просування запитів поза чергою.
2. Несанкціоноване читання матриць порівнянь дозволяє зовнішньому зловмиснику відтворити логіку прийняття рішень (наприклад, як система «цінує» пацієнтів).
3. Replay-атаки через кеш – повторне відправлення вже обробленого запиту для створення штучного навантаження або маніпуляції результатом.
4. Підміна запитів у межах хмарної мережі – використання відкритого API для втручання у сценарій маршрутизації запиту.

Таблиця 1

Класифікація даних у системі

Категорія даних	**Приклади даних**	**Рівень критичності**	**Коментар щодо захисту**
Персональні	Ім’я, дата народження, контактні дані	Високий	Підлягає захисту згідно з GDPR / ДСТУ
Медичні	Діагнози, анамнез, результати аналізів	Високий	Визнається чутливою інформацією за HIPAA
Аналітичні	AHP-матриці, ваги критеріїв, інтегральні оцінки	Високий	Впливають на рішення → ціль атаки
Адміністративні	Дані про виконавців, маршрутизацію, логіку обробки	Середній	Потребують контролю змін
Лог-файли	Журнали звернень, аудиту, IP-адреси користувачів	Середній	Потребують захисту від фальсифікації
Службові	Кешовані результати, тимчасові маркери, технічні метадані	Низький-середній	Можуть стати вектором для Replay-атак

Така класифікація та оцінка загроз стали основою для формування цільової архітектури інформаційної безпеки. Розміщення інформаційної системи у хмарній інфраструктурі,

https://doi.org/10.32782/mathematical-modelling/2025-8-2-26 ISSN 2618-0332 (print), 2618-0340 (online)

з одного боку, забезпечує масштабованість, гнучкість і доступність сервісів, однак, з іншого боку, створює додатковий рівень уразливості, пов'язаний із багатокористувацькою природою середовища, непрямим контролем над фізичними ресурсами, а також відкритістю API-інтерфейсів для взаємодії з іншими медичними чи аналітичними сервісами.

У межах розробки системи пріоритетного управління медичними послугами ці загрози набувають особливого значення, оскільки будь-який вплив на процес ранжування пацієнтів може мати прямі наслідки для якості й швидкості надання медичної допомоги.

1. Зовнішні загрози

Основними загрозами зовнішнього походження, які ми ідентифікували як критичні для роботи AHP-модуля в хмарі, є:

– Несанкціонований доступ через відкритий API – за наявності слабкої автентифікації можлива імітація запитів від системи обробки звернень. Це відкриває шлях до масових вставок запитів, які не повинні були бути допущені.

– Атаки типу replay (повторного запиту) – надсилання повторних легітимних запитів з метою викликати дублювання рішень або перевантаження кешу пріоритетних відповідей. У системі з алгоритмом AHP це може призвести до штучної ескалації вибраних запитів.

– Атаки типу XSS і SQL-injection через вхідні форми – у випадку, якщо адміністративний інтерфейс (наприклад, редагування ваг критеріїв) не досить захищений, можливе втручання у саму логіку обрахунків [3].

2. Внутрішні загрози (інсайдерські)

В умовах хмарного середовища загрозу становлять також внутрішні дії уповноважених осіб, які можуть свідомо або несвідомо змінити ключові налаштування системи. Ми виділяємо такі загрози:

– Редагування ваг критеріїв без належного контролю – навіть незначна зміна одного коефіцієнта може суттєво змінити порядок обслуговування пацієнтів. Це створює можливості для маніпуляцій або саботажу.

– Видалення або редагування історії рішень AHP – якщо немає незалежного журналу, зміни в даних можна приховати, і система втратить контроль над обґрунтованістю прийнятих рішень.

– Перевизначення структури критеріїв (ієрархій) – особливо небезпечно у системах, де адміністратор має право змінювати не лише ваги, а і саму логіку моделі AHP.

3. Загрози, пов'язані з архітектурою кешу та масштабування

Особливої уваги вимагають загрози повторного використання результатів, пов'язані з кешуванням даних у Redis або Memcached. Кеш у хмарному середовищі використовується для прискорення обробки запитів, але саме він може стати вектором атаки, якщо:

– відсутня перевірка унікальності запиту (nonce);

– не встановлено TTL (часу життя);

– використовуються загальні ключі (без індивідуального зв'язку з користувачем / запитом).

Це дозволяє зловмиснику використати закешовану відповідь з високим пріоритетом для просування інших звернень.

Наприклад, так виглядає replay-атака у системі ранжування:

```
# Вразливий сценарій – запит без nonce і TTL
cached_response = redis. get (f'priority:{patient_id}»)
if cached_response:
return cached_response
else:
response = calculate_priority (patient_data)
redis. set (f'priority:{patient_id}», response) # без перевірки!
return response
```

https://doi.org/10.32782/mathematical-modelling/2025-8-2-26 ISSN 2618-0332 (print), 2618-0340 (online)

Цей фрагмент демонструє, як відсутність контролю над кешем дозволяє повторно використати результат для іншого запиту з підставним patient_id, що може змінити логіку системи. Таким чином, ми ідентифікували конкретні загрози, які мають бути враховані в моделі безпеки нашої системи.

Побудова архітектури безпеки хмарної інформаційної системи не може здійснюватися ізольовано від її логіки функціонування. У нашому випадку система не лише зберігає чутливі медичні дані, а й виконує управлінські дії на їх основі, застосовуючи метод AHP для ранжування звернень пацієнтів. Тому надійність, контроль і цілісність не є лише формальними вимогами, а критичними елементами правильності її функціонування [4].

На підставі цього сформовано п'ять принципів, якими повинна керуватися система безпеки:

1. Принцип мінімальних повноважень та чіткої ролі (RBAC). Усі модулі та користувачі системи повинні мати доступ лише до тих функцій і даних, які необхідні для виконання їхніх обов'язків. Наприклад:

– Адміністратор має право змінювати структуру критеріїв AHP, але не впливає на медичні записи.

– Медичний працівник має доступ лише до даних власних пацієнтів, без можливості перегляду аналітичної логіки системи.

– Аналітик має право бачити ваги критеріїв, але не може їх редагувати без додаткової авторизації.

2. Принцип журналювання управлінських дій. Зміна будь-якого елементу, який впливає на логіку прийняття рішень (вага критерію, оцінка альтернативи, структура ієрархії), повинна бути зафіксована у незалежному журналі змін з указанням: користувача, часу, старого і нового значення. Це дозволяє гарантувати обґрунтованість рішень системи і в разі потреби провести аудит.

Приклад реалізації на рівні СУБД (PostgreSQL):

```
CREATE TABLE criteria_weights_log
(id SERIAL PRIMARY KEY,
criteria_id INT,
old_weight NUMERIC,
new_weight NUMERIC,
changed_by TEXT,
changed_at TIMESTAMP DEFAULT current_timestamp);

CREATE OR REPLACE FUNCTION log_criteria_weight_change()
RETURNS TRIGGER AS $$
BEGIN
INSERT INTO criteria_weights_log (criteria_id, old_weight, new_weight, changed_by)
VALUES (OLD. id, OLD. weight, NEW. weight, current_user);
RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_log_criteria_weight
BEFORE UPDATE ON criteria_weights
FOR EACH ROW
WHEN (OLD. weight IS DISTINCT FROM NEW. weight)
EXECUTE FUNCTION log_criteria_weight_change();
```

Цей тригер забезпечує повний контроль над усіма змінами ваг критеріїв – ключового параметра в методі AHP, що прямо впливає на результат ранжування пацієнтів.

3. Принцип валідації правомірності управлінських рішень. У межах розробки пропонується реалізація внутрішньої перевірки «легітимності» управлінських рішень. Наприклад, кожна

https://doi.org/10.32782/mathematical-modelling/2025-8-2-26 ISSN 2618-0332 (print), 2618-0340 (online)

інтегральна оцінка повинна супроводжуватись контрольним хешем на основі структури ієрархії та актуальних ваг. У разі спроби маніпуляції – зміна хешу сигналізує про спотворення рішення.

4. Принцип незалежності критичних модулів. AHP-модуль ранжування має бути ізольованим від інтерфейсних або візуалізаційних модулів. Це означає, що навіть повний компроміс вебінтерфейсу не дасть змоги змінити структуру прийняття рішень без доступу до окремого обчислювального ядра або API.

5. Принцип прозорості дій адміністратора. Кожна адміністративна дія (створення нового критерію, оновлення ієрархії, перегляд ваг інших користувачів) повинна бути зафіксована з поясненням, що виводиться в окремому звіті для контролю управління.

Ці принципи ляжуть в основу архітектурної моделі захисту системи, яка буде розкрита в наступному блоці. Таким чином, безпека реалізується не як окрема функція, а як вбудований механізм гарантії справедливості, цілісності та об'єктивності у прийнятті рішень про пріоритет доступу до медичних послуг.

Отже, з метою запобігання загрозам, ідентифікованим на попередніх етапах, було сформовано комплекс методів захисту, що забезпечують цілісність, конфіденційність і контрольованість роботи хмарної системи управління медичними послугами. Ці методи не лише відповідають вимогам міжнародних стандартів безпеки (ISO / IEC 27001, HIPAA), але й адаптовані до архітектури алгоритмічного прийняття рішень, що є основою нашої системи.

Для захисту медичної інформації на рівні збереження й передачі ми використовуємо алгоритм симетричного шифрування AES – 256. Це дозволяє ефективно зашифрувати навіть великі обсяги даних з високим рівнем стійкості до злому.

Наведемо фрагмент реалізації (Python):

```
from Crypto. Cipher import AES
from Crypto. Random import get_random_bytes
import base64

def encrypt_data (data: str, key: bytes) -> str:
cipher = AES. new (key, AES. MODE_EAX)
nonce = cipher. nonce
ciphertext, tag = cipher. encrypt_and_digest (data. encode('utf – 8'))
return base64. b64encode (nonce + ciphertext). decode('utf – 8')

def decrypt_data (enc_data: str, key: bytes) -> str:
raw = base64. b64decode (enc_data)
nonce = raw[:16]
ciphertext = raw [16:]
cipher = AES. new (key, AES. MODE_EAX, nonce=nonce)
return cipher. decrypt (ciphertext). decode('utf – 8')

# Використання:
key = get_random_bytes (32) # 256-bit ключ
encrypted = encrypt_data («Медичний запис: ID_23456», key)
decrypted = decrypt_data (encrypted, key)
```

Цей код придатний для інтеграції з REST API або базою даних, і може застосовуватись для шифрування записів у таблицях, що зберігають діагнози, призначення, а також аналітичні оцінки.

Розробимо рольову модель доступу – RBAC, яка базується на чітко визначених ролях користувачів та їхніх обмеженнях у контексті функцій системи. Це дозволить мінімізувати ризики інсайдерських втручань, неконтрольованих змін у критеріях або розрахунках AHP.

https://doi.org/10.32782/mathematical-modelling/2025-8-2-26 ISSN 2618-0332 (print), 2618-0340 (online)

Матриця доступу є ключовим інструментом обмеження повноважень користувачів відповідно до їхніх функціональних ролей. Її мета – гарантувати, що кожен користувач має доступ лише до тих компонентів системи, які відповідають його посадовим обов'язкам, і не більше [3].

У процесі розробки виходили з принципу мінімальних привілеїв, відповідно до якого навіть адміністративний персонал не має безумовного доступу до всіх функцій. Це особливо важливо для медичної інформаційної системи, яка реалізує критичні алгоритми (зокрема, метод AHP) і приймає автоматизовані рішення щодо обслуговування пацієнтів [3].

У структурі доступу враховано не лише технічний складник, а й логіку управлінської відповідальності:

1. Адміністратор не має доступу до медичних записів пацієнтів, але повністю керує архітектурою моделі AHP.

2. Медичний працівник не може змінювати ваги критеріїв, однак має право ініціювати запити.

3. Аналітик може бачити все, що пов'язано з алгоритмами і трендами, але не втручається у самі дані.

Таким чином, запропонована матриця доступу дозволяє:

– захистити критичні дані та алгоритми від несанкціонованого впливу;

– ізолювати дії різних категорій користувачів у межах їхніх функцій;

– забезпечити простежуваність кожної дії у системі;

– мінімізувати ризики інсайдерських загроз і людських помилок.

Формування такої моделі є необхідною умовою не лише для відповідності системи стандартам безпеки, але й для підвищення надійності рішень, що приймаються на її основі.

Основні ролі:

– Адміністратор системи (Admin) – редагування критеріїв, запуск перерахунку моделі, контроль доступу.

– Лікар (Doctor) – доступ до медичних даних пацієнтів, ініціація запиту.

– Оператор (Staff) – реєстрація звернень, супровід.

– Аналітик (Analyst) – перегляд ваг, результатів, трендів, без права втручання.

Таблиця 2

RBAC-матриця доступу

Функція	**Admin**	**Doctor**	**Staff**	**Analyst**
Перегляд медичних даних	✕	✓	✓	✕
Редагування ваг AHP	✓	✕	✕	✕
Перегляд результатів AHP	✓	✓	✓	✓
Ініціація запиту	✕	✓	✓	✕
Перегляд логів змін	✓	✕	✕	✓

Реалізацію доцільно здійснити через JWT-токени із вбудованими ознаками ролей і контроль доступу на рівні middleware у бекенді.

Також доцільно реалізувати обов'язкове журналювання всіх дій, пов'язаних із доступом до критеріїв, змін у структурі AHP, перегляду медичних даних та ініціації обчислень. Логи зберігаються в базі даних або у форматі JSON-файлів із контролем цілісності (SHA – 256 хеш на кожен запис). Такі записи не лише підлягають аудиту, але й можуть бути автоматично аналізовані системами моніторингу (Zabbix, Prometheus) для виявлення аномалій [9].

https://doi.org/10.32782/mathematical-modelling/2025-8-2-26 ISSN 2618-0332 (print), 2618-0340 (online)

Фрагмент лог-файлу:

{
«timestamp»: «2025–08–27T11:40:32Z»,
«user»: «admin1»,
«action»: «update_weight»,
«criterion»: «Medical urgency»,
«old_value»: 0.314,
«new_value»: 0.270,
«ip»: «192.168.12.105»
}

Таким чином, методи захисту, впроваджені у проєктованій системі, не ізольовані від її логіки, а є невіддільною частиною алгоритмічного процесу пріоритетизації, що гарантує довіру до автоматизованих рішень у критично важливій сфері медичних послуг.

Припускаємо із загальновідомої логіки роботи аналогічних систем у світовій практиці, що захист на рівні програмного коду та баз даних є критично необхідним, але недостатнім. Безпека повинна бути підкріплена зовнішніми інфраструктурними рішеннями, які автоматично реагують на аномалії, блокують спроби вторгнення, захищають канали доступу до критичних модулів і дають змогу вести глибокий аудит [8].

Реалізуємо три ключові інтеграційні напрями, які дозволяють поєднати рівні безпеки – від прикладного до мережевого:

1. Захист API та інтерфейсів через Cloudflare. Для захисту відкритих API-ендпоінтів, які використовуються зовнішніми медичними системами (наприклад, лабораторіями чи державними реєстрами), застосовується Cloudflare як зворотний проксі та захисний шлюз. Він реалізує:

– WAF (Web Application Firewall) – автоматичне блокування відомих атак типу SQL-injection, XSS та обходу автентифікації;

– Rate limiting – обмеження кількості запитів на IP або токен протягом часу;

– Geo fencing – блокування доступу з небажаних регіонів.

Це дозволяє захистити як публічні інтерфейси системи, так і окремі компоненти, зокрема AHP-модуль, від масованого звернення (наприклад, DDoS через симуляцію пріоритетних запитів) [9].

2. Моніторинг подій безпеки через Zabbix. Для контролю критичних подій і своєчасного реагування на аномалії доцільно інтегрувати систему моніторингу Zabbix. Вона дозволяє:

– вести спостереження за логами (доступ до ваг, зміни ієрархій, помилки в запитах);

– налаштовувати алерти на спроби несанкціонованого доступу;

– відстежувати пікові навантаження на систему ранжування;

– інтегруватись із телеграм-ботами, поштою або SMS.

Таке рішення дозволяє оперативно втручатися у випадках, коли система потрапляє під цілеспрямовану атаку або коли внутрішні користувачі порушують регламент [9].

3. Автоматичне блокування атак через Fail2ban. На рівні сервера доцільно реалізувати систему динамічного блокування IP-адрес, з яких надходить підозріла активність. Інструмент Fail2ban аналізує лог-файли авторизації, запитів до критичних модулів, зокрема AHP та редагування ваг, і створює правила блокування [8].

Фрагмент конфігурації Fail2ban (auth. log + пріоритетизація):

[sshd]
enabled = true
port = ssh
logpath = /var / log / auth. log
maxretry = 5
findtime = 600
bantime = 3600

https://doi.org/10.32782/mathematical-modelling/2025-8-2-26 ISSN 2618-0332 (print), 2618-0340 (online)

```
[api-ahp]
enabled = true
port = http, https
filter = api-ahp
logpath = /var / log / ahp. log
maxretry = 3
findtime = 300
bantime = 7200
```

Цей механізм дозволяє блокувати користувача, який, наприклад, тричі спробував змінити ваги критеріїв або підробити структуру ієрархії, не маючи відповідних прав.

4. Інтеграція з SIEM-системами. У перспективі можлива інтеграція з централізованими системами аналізу подій безпеки (SIEM), зокрема ELK stack або Splunk, що дозволить:

– об'єднати логи із серверів, API, баз даних;

– автоматично виявляти пов'язані інциденти;

– формувати інциденти для реагування службою ІБ.

Таким чином, завдяки розробці система не обмежується вбудованим захистом, а інтегрується у середовище постійного моніторингу, блокування і профілактики загроз, що відповідає найвищим вимогам до безпеки критичних систем у сфері охорони здоров'я.

Також розробимо принципи безпеки кешованих обчислень, які будуть інтегровані у нашу систему: запобігання атакам повторного запиту (Replay) в Redis / Memcached.

У хмарній системі управління медичними послугами кеш використовується для прискорення обробки однотипних запитів, що особливо важливо в умовах високого навантаження. Для цього задіяно високопродуктивні сервіси кешування – Redis та Memcached. Вони зберігають частину проміжних обчислень алгоритму AHP (наприклад, попередньо згенеровані інтегральні оцінки) для повторного використання без повторного запуску складних розрахунків.

Однак використання кешу створює потенційно небезпечні вектори атак, зокрема replay-атаки, коли зловмисник повторно надсилає запит з тією ж сесією або зловживанням відкритим patient_id, щоб отримати «засвоєну» раніше відповідь із підвищеним пріоритетом [8].

Розробимо підхід до захисту кешу. На основі цього впроваджуємо двоетапну перевірку достовірності кешованих обчислень, яка базується на:

– nonce (одноразовому маркері) – кожен запит до системи має супроводжуватись унікальним токеном, який використовується лише один раз і зберігається у Redis як використаний;

– TTL (time-to-live) – обмеження часу життя кешованого результату, після чого він автоматично видаляється і має бути згенерований заново.

Фрагмент реалізації захисту у Redis (Python):

```
import redis
import uuid
import time

r = redis. Redis (host='localhost', port=6379, db=0)

def is_replay (nonce):
return r. exists (f'used_nonce:{nonce}»)

def store_nonce (nonce):
r. set (f'used_nonce:{nonce}», «1», ex=600) # TTL 10 хв.

def get_cached_priority (patient_id):
return r. get (f'priority:{patient_id}»)
```

https://doi.org/10.32782/mathematical-modelling/2025-8-2-26 ISSN 2618-0332 (print), 2618-0340 (online)

```
def store_priority (patient_id, value):
r. set (f'priority:{patient_id}», value, ex=300) # TTL 5 хв.

# Приклад обробки запиту
nonce = «abc123» # отримано з frontend / API
if is_replay (nonce):
raise Exception («Replay-attack detected!»)
else:
store_nonce (nonce)
cached = get_cached_priority («patient_456»)
if cached:
return cached
else:
result = calculate_priority (data)
store_priority («patient_456», result)
return result
```

У цьому блоці коду:

– перевіряємо, чи nonce вже використовувався;

– застосовуємо TTL як до пріоритету, так і до nonce;

– блокуємо повторний доступ, навіть якщо ідентифікатор пацієнта однаковий.

Пропонуються інші технічні заходи для Redis / Memcached:

– ізоляція кешу за користувачами (namespace або ключі user_id: patient_id);

– шифрування чутливих даних у кеші;

– журналювання критичних запитів;

– обмеження доступу до портів Redis / Memcached лише з довірених IP.

Таким чином, впроваджено адаптивну політику захисту кешованих обчислень, яка унеможливлює використання кешу як інструменту маніпуляції логікою пріоритетів. Це забезпечує не лише стабільність продуктивності, але й інваріантність управлінських рішень у хмарному середовищі, що є елементом наукової новизни цієї роботи.

Було сформовано власну концепцію побудови безпечної хмарної інформаційної системи управління медичними послугами, яка враховує як загальноприйняті принципи захисту інформації, так і специфіку алгоритмічного компонента – багатокритеріальної моделі прийняття рішень на основі методу AHP. Здійснено класифікацію типів даних, що обробляються системою, з урахуванням їх критичності: медичні, аналітичні, службові, адміністративні. Особливу увагу приділено захисту вагових коефіцієнтів та структури критеріїв, оскільки саме вони формують підґрунтя для пріоритетизації обслуговування пацієнтів. Проаналізовано типові загрози, притаманні хмарному середовищу, зокрема атаки повторного запиту (replay), компрометацію API, несанкціоновані зміни логіки моделі. На основі цього запропоновано принципи побудови безпечної системи, включаючи:

– мінімізацію повноважень користувачів через рольову модель (RBAC);

– журналювання змін у структурі рішень (SQL-тригери для ваг AHP);

– захист кешованих обчислень від маніпуляцій (nonce + TTL у Redis);

– контроль легітимності управлінських рішень.

Запропоновано до реалізації приклад шифрування медичних записів за алгоритмом AES – 256, побудовано матрицю доступу користувачів та інтегровано інструменти інфраструктурного захисту – Cloudflare, Fail2ban, Zabbix. Завдяки запропонованим рішенням забезпечується не лише захист даних, а й цілісність логіки системи прийняття рішень, що є критичним у медичній сфері. Таким чином, реалізовано модель комплексної, адаптивної та алгоритмічно інтегрованої інформаційної безпеки, яка стане підґрунтям для технічного впровадження в межах архітектури та реалізації системи.

https://doi.org/10.32782/mathematical-modelling/2025-8-2-26 ISSN 2618-0332 (print), 2618-0340 (online)

Висновки

У результаті проведеного дослідження сформовано концептуальні та практичні засади побудови моделі комплексної інформаційної безпеки хмарної системи управління медичними послугами. Визначено, що поєднання методів аналітичної ієрархії (AHP) із сучасними механізмами контролю доступу (RBAC) та криптографічного захисту (AES-256) забезпечує високий рівень стійкості системи до внутрішніх і зовнішніх загроз.

Запропонована модель передбачає інтеграцію безпекових політик безпосередньо в процес прийняття рішень, що дозволяє зменшити ризик несанкціонованої модифікації параметрів управління та підвищує довіру до автоматизованих рішень у медичній сфері.

Практичне впровадження моделі можливе на базі поширених інструментів хмарної інфраструктури, таких як Redis і PostgreSQL, з урахуванням принципів журналювання, контролю кешу та моніторингу подій безпеки.

Список використаної літератури

1. Griebel L., Prokosch H.-U., Köpcke F., et al. A scoping review of cloud computing in healthcare. *BMC Medical Informatics and Decision Making*. 2015. Vol. 15. Article 17. DOI: 10.1186/s12911-015-0145-7.
2. Gao F., Sunyaev A. Context Matters: A review of the determinant factors in the decision to adopt cloud computing in healthcare. *International Journal of Information Management*. 2019. Vol. 48. P. 120–138.
3. Putzier M., Kuhlmann L., Müller M., et al. Implementation of cloud computing in the German healthcare system. *npj Digital Medicine*. 2024. Vol. 7. Article 12.
4. Moumtzi G., Mavridis I., Vergidis K. A security framework for healthcare cloud computing. *International Journal of Reliable and Quality E-Healthcare*. 2015. Vol. 4. No. 2. P. 1–12.
5. Al-Dmour A., Al-Dmour R., Shannak R. The effect of cloud computing on improving the quality of healthcare services. *Journal of Theoretical and Applied Information Technology*. 2020. Vol. 98. No. 23. P. 3807–3820.
6. Javaid M., Haleem A., Singh R.P., Khan S., Suman R. Evolutionary trends in progressive cloud computing based healthcare: Ideas, enablers, and barriers. *International Journal of Cognitive Computing in Engineering*. 2022. Vol. 3. P. 124–135.
7. Kuo A.M. Opportunities and challenges of cloud computing to improve health care services. *Journal of Medical Internet Research*. 2011. Vol. 13. No. 3. e67.
8. Sajid A., Abbas H. Data Privacy in Cloud-assisted Healthcare Systems: State of the Art and Future Challenges. *Journal of Medical Systems*. 2016. Vol. 40. No. 6. Article 155.
9. Chenthara S., Ahmed K., Wang H., Whittaker F. Security and Privacy-Preserving Challenges of e-Health Solutions in Cloud Computing. *IEEE Access*. 2019. Vol. 7. P. 74361–74382.
10. Dolan J.G. Shared decision-making – transferring research into practice: The Analytic Hierarchy Process (AHP). *Patient Education and Counseling*. 2008. Vol. 73. No. 3. P. 418–425.
11. Saaty T.L. The Analytic Hierarchy Process: Planning, Priority Setting, Resource Allocation. New York : McGraw-Hill, 1980.

References

1. Griebel, L. et al. (2015). A scoping review of cloud computing in healthcare. *BMC Medical Informatics and Decision Making*, 15 (17). https://doi.org/10.1186/s12911–015–0145–7 [in English].
2. Gao, F., & Sunyaev, A. (2019). Context Matters: A review of the determinant factors in the decision to adopt cloud computing in healthcare. *International Journal of Information Management*, 48, 120–138 [in English].

3. Putzier, M. et al. (2024). Implementation of cloud computing in the German healthcare system. *npj Digital Medicine*, 7 (12) [in English].
4. Moumtzi, G., Mavridis, I., & Vergidis, K. (2015). A security framework for healthcare cloud computing. *International Journal of Reliable and Quality E-Healthcare*, 4(2), 1–12 [in English].
5. Al-Dmour, A., Al-Dmour, R., & Shannak, R. (2020). The effect of cloud computing on improving the quality of healthcare services. *Journal of Theoretical and Applied Information Technology*, 98(23), 3807–3820 [in English].
6. Javaid, M., Haleem, A., Singh, R.P., Khan, S., & Suman, R. (2022). Evolutionary trends in progressive cloud computing based healthcare: Ideas, enablers, and barriers. *International Journal of Cognitive Computing in Engineering*, 3, 124–135 [in English].
7. Kuo, A.M. (2011). Opportunities and challenges of cloud computing to improve health care services. *Journal of Medical Internet Research*, 13(3), e67 [in English].
8. Sajid, A., & Abbas, H. (2016). Data Privacy in Cloud-assisted Healthcare Systems: State of the Art and Future Challenges. *Journal of Medical Systems*, 40(6), 155 [in English].
9. Chenthara, S., Ahmed, K., Wang, H., & Whittaker, F. (2019). Security and Privacy-Preserving Challenges of e-Health Solutions in Cloud Computing. *IEEE Access*, 7, 74361–74382 [in English].
10. Dolan, J.G. (2008). Shared decision-making – transferring research into practice: The Analytic Hierarchy Process (AHP). *Patient Education and Counseling*, 73(3), 418–425 [in English].
11. Saaty, T.L. (1980). The Analytic Hierarchy Process: Planning, Priority Setting, Resource Allocation. McGraw-Hill [in English].

Семчишин Володимир Миколайович – аспірант кафедри програмної інженерії Тернопільського національного технічного університету імені Івана Пулюя. E-mail: volodymyr_semchyshyn9391@tntu.edu.ua, ORCID: 0009-0008-9206-8657.

Михалик Дмитро Михайлович – к.т.н., доцент кафедри програмної інженерії Тернопільського національного технічного університету імені Івана Пулюя. E-mail: myhalyk_d@tntu.edu.ua, ORCID: 0000-0001-9032-695X.

Semchyshyn Volodymyr Mykolayovych – Postgraduate Student at the Department of Software Engineering of Ivan Pulyuy Ternopil National Technical University. E-mail: volodymyr_semchyshyn9391@tntu.edu.ua, ORCID: 0009-0008-9206-8657.

Mykhalyk Dmytro Mykhaylovych – Candidate of Technical Sciences, Associate Professor at the Department of Software Engineering of Ivan Pulyuy Ternopil National Technical University. E-mail: myhalyk_d@tntu.edu.ua, ORCID: 0000-0001-9032-695X.

Отримано: 05.11.2025
Рекомендовано: 09.12.2025
Опубліковано: 30.12.2025