DOI: https://doi.org/10.36910/6775-2524-0560-2026-62-18
УДК 004.056.5
Андрущак Ігор Євгенович, д.т.н., професор
http://orcid.org/0000-0002-8751-4420
Колошко Олександр Володимирович, аспірант
https://orcid.org/0009-0001-9325-9542
Луцький національний технічний університет, м. Луцьк, Україна

ДЕТЕРМІНОВАНІ AEAD ДЛЯ ІДЕНТИЧНИХ ПОВІДОМЛЕНЬ: РИЗИКИ Й ЕКОНОМІЯ

Андрущак І.Є., Колошко О.В. Детерміновані AEAD для ідентичних повідомлень: ризики й економія. У статті досліджено особливості застосування детермінованих AEAD (Authenticated Encryption with Associated Data) алгоритмів для шифрування ідентичних повідомлень у високонавантажених інформаційних системах реального часу. Метою роботи є аналіз компромісу між безпекою та ефективністю при використанні детермінованого шифрування, де відсутність випадкових ініціалізаційних векторів забезпечує відтворюваність результатів, але водночас породжує ризики витоку структурної інформації. Розглянуто теоретичні основи детермінованих AEAD-схем, зокрема AES-SIV та AES-GCM-SIV, і проведено їх порівняння з недетермінованими режимами AES-GCM та ChaCha20-Poly1305 за ключовими показниками продуктивності, безпеки та накладних витрат. Проаналізовано практичні сценарії застосування у системах з високою частотою повторюваних запитів: IoT-телеметрія, кешовані транзакції, мобільні сервіси та платіжні системи. Окремо розглянуто модель семантичної безпеки IND-CPA та умови її порушення при повторенні шифротекстів. Розроблено модель кількісної оцінки ресурсної економії, що враховує витрати на генерацію nonce, обсяги службових даних і споживання енергії. Результати експериментальної оцінки на платформі ARM Cortex-A53 підтвердили можливість досягнення до 15–20% економії обчислювальних ресурсів та скорочення допоміжних даних майже на 40%. Для мінімізації ризиків повторюваних шифротекстів систематизовано десять методів захисту, включаючи контекстну унікалізацію пов'язаних даних, доменне розділення ключів, введення випадкового наповнення та регулярну ротацію ключів. Встановлено, що детерміновані AEAD є доцільними у спеціалізованих середовищах з обмеженими ресурсами, однак вимагають суворого контролю контексту застосування для запобігання витоку структурної інформації.
Ключові слова: детерміноване шифрування, AEAD, автентифікація даних, криптографічна безпека, AES-GCM-SIV, конфіденційність, продуктивність.

Andrushchak I., Koloshko O. Deterministic AEADs for identical messages: risks and savings. The article examines the peculiarities of using deterministic AEAD (Authenticated Encryption with Associated Data) algorithms for encrypting identical messages in highly loaded real-time information systems. The aim of the work is to analyse the trade-off between security and efficiency when using deterministic encryption, where the absence of random initialisation vectors ensures reproducibility of results but at the same time creates risks of structural information leakage. The theoretical foundations of deterministic AEAD schemes, in particular AES-SIV and AES-GCM-SIV, are considered and compared with non-deterministic modes AES-GCM and ChaCha20-Poly1305 in terms of key performance indicators, security, and overhead. Practical application scenarios in systems with a high frequency of repeated requests are analysed: IoT telemetry, cached transactions, mobile services, and payment systems. A model for quantitative assessment of resource savings has been developed, taking into account the costs of nonce generation, service data volumes, and energy consumption. To minimise the risks of repeated ciphertexts, ten protection methods have been systematised, including contextual uniqueness of related data, domain separation of keys, introduction of random padding, and regular key rotation. It has been established that deterministic AEADs are appropriate in specialised environments with limited resources, but require strict control of the application context to prevent the leakage of structural information.
Keywords: deterministic encryption, AEAD, data authentication, cryptographic security, AES-GCM-SIV, confidentiality, performance.

Постановка проблеми. У сучасних інформаційних системах, що працюють у режимі реального часу, швидкість обробки та обсяг переданих даних зростають експоненційно. Сотні мільйонів транзакцій, запитів до баз даних, повідомлень від IoT-пристроїв і токенів автентифікації щосекунди генеруються в інфраструктурах хмарних сервісів, платіжних систем і мобільних додатків. У таких умовах криптографічні операції мають забезпечувати не лише високий рівень безпеки, але й оптимальну ефективність. Саме тому виникає інтерес до детермінованих алгоритмів AEAD, які усувають необхідність у випадкових ініціалізаційних векторах (nonce) і гарантують однаковий вихід для однакових вхідних даних [1].

Класичні недетерміновані схеми AEAD, такі як AES-GCM або ChaCha20-Poly1305, використовують унікальний nonce для кожної операції шифрування. Це запобігає повторенню шифротекстів і унеможливлює статистичний аналіз. Однак така стратегія має недолік: потребує постійного генерування, зберігання або передачі унікальних nonce, що створює додаткове навантаження на систему та збільшує ризик помилок при їх повторному використанні. Відомо, що багато компрометацій шифрування в реальних продуктах пов'язано саме з неправильним керуванням nonce або його колізіями [2]. Детерміновані AEAD усувають цей фактор, вони не

покладаються на випадковість, забезпечуючи стабільне, відтворюване шифрування. Цей підхід особливо привабливий у високопродуктивних системах, де однакові повідомлення (наприклад, токени доступу, повідомлення з кешу, хешовані запити до API чи інваріантні конфігурації) можуть повторюватися тисячі разів. Відсутність генерації nonce зменшує затрати пам'яті, спрощує архітектуру протоколу й дає економію до 15–20% обчислювальних ресурсів у системах із великим обсягом однотипних даних [3]. Однак детермінізм породжує і нові ризики. Якщо противник спостерігає ідентичні шифротексти, він може зробити висновок про повторення вхідних повідомлень, навіть не знаючи їхнього змісту. Це може призвести до витоку структурної інформації, що у криптографії вважається серйозним порушенням моделі семантичної безпеки, застосування детермінованих AEAD вимагає суворого дотримання контекстних обмежень. Наприклад, унікалізації пов'язаних даних (Associated Data), введення контекстних тегів або ізоляції доменів ключів [4].

Отже, проблема полягає у визначенні безпечних умов і практичних меж використання детермінованих AEAD для ідентичних повідомлень, де досягається баланс між зниженням навантаження на систему та мінімізацією ризиків для конфіденційності даних.

Аналіз актуальних досліджень та публікацій. У контексті дослідження детермінованих режимів AEAD з особливим акцентом на повідомлення, що можуть бути ідентичними, важливо звернути увагу як на теоретичні основи, так і на практичні впровадження. Нижче наведено огляд ключових робіт.

Одна з ранніх фундаментальних праць «Deterministic Authenticated-Encryption: A Provable-Security Treatment of the Key-Wrap Problem» розглядає модель детермінованої аутентифікованої енкрипції (Deterministic AE), та показує, що коли входи повторюються, шифртекст у детермінованих схемах також буде повторюватися, що створює додатковий ризик інформаційного витоку. Ця робота встановлює, що детермінованість усуває потребу у випадковому nonce, але за рахунок того, що однакові plaintext + ключ дають однаковий ciphertext, зберігається вразливість у вигляді виявлення повторень [5].

Практичні режими AEAD з детермінованістю або стійкістю до повторного nonce, режим AES-GCM-SIV документований у «AES-GCM-SIV: Specification and Analysis» є прикладом AEAD-схеми, що забезпечує стійкість до повторного використання nonce (nonce-misuse resistant) [6].

У праці «Revisiting AES-GCM-SIV: Multi-user Security, Faster Key Derivation» автори аналізують безпеку цього режиму у мультикористувацькому середовищі та намагаються визначити обмеження щодо кількості повідомлень під одним ключем, а також нюанси реалізації. Додатково, праця «Stronger Security Variants of GCM-SIV» досліджує вдосконалення GCM-SIV для досягнення «beyond-birthday-bound» безпеки [7-8]. Ці дослідження чітко підкреслюють, що навіть режими із стійкістю до повторного nonce не знімають повністю ризику пов'язаної з детермінованістю (наприклад, коли plaintext і nonce ті самі, ciphertext буде ідентичним).

Спеціалізовані аспекти повторення повідомлень, в роботі «How Subtle AEAD Differences can impact Protocol Security» аналізуються конкретні протокольні реалізації AEAD-схем, зокрема вплив того, що однакові повідомлення зашифровані під однаковими ключами/nonce мають ідентичні результати. Автори показують, що це дає можливість зловмиснику виявляти повтори, будувати профілі активності або власне компрометувати метадані. Коли повідомлення ідентичні, детерміновані AEAD-режими можуть давати «двійники» ciphertext, що створює ризик інформаційного витоку, навіть якщо сам plaintext не розкрити [9].

Узагальнюючи проведений аналіз, слід підкреслити, що існуючі дослідження охоплюють численні аспекти реалізації та безпеки режимів AEAD, зокрема таких як AES-GCM-SIV та інші детерміновані схеми. Проте, виявлено суттєвий дефіцит праць, які досліджують їх застосування саме для ідентичних повідомлень, з точки зору економії ресурсів і ризиків витоку інформації. Отже, виникає потреба у цілеспрямованому дослідженні, що формалізує межі застосування детермінованих AEAD-рішень, оцінює їхню продуктивність та визначає критерії допустимості в реальних сценаріях.

Метою роботи є дослідження особливостей застосування детермінованих AEAD-алгоритмів для шифрування ідентичних повідомлень, із фокусом на оцінці компромісу між безпекою та ефективністю. У рамках дослідження передбачається виявити ризики витоку структурної інформації, що виникають через повторюваність шифротекстів, та визначити межі безпечного використання детермінованих AEAD у системах із високою частотою однакових запитів. Крім того, мета включає порівняння продуктивності детермінованих і недетермінованих

режимів (AES-GCM-SIV, AES-SIV, ChaCha20-Poly1305) і розроблення рекомендацій щодо їх доцільного застосування у практичних середовищах, де важливими є як збереження автентичності, так і економія обчислювальних ресурсів.

Виклад основного матеріалу. Сучасні криптографічні системи захисту даних у більшості випадків спираються на концепцію AEAD – аутентифікованого шифрування з пов'язаними даними. Основна мета цього підходу полягає в одночасному забезпеченні конфіденційності (приховування змісту повідомлення) та цілісності (гарантія того, що дані не були змінені під час передавання або зберігання). AEAD поєднує в собі переваги симетричного шифрування (AES, ChaCha20) і механізмів перевірки автентичності (HMAC, Poly1305), що робить його базовим стандартом у більшості протоколів безпеки: TLS 1.3, QUIC, WireGuard тощо [10]. У класичній моделі AEAD процес шифрування визначається функцією:

$$C = AEAD\_Encrypt\ (K, N, A, P) \tag{1}$$

де:

K – секретний ключ,

N – ініціалізаційний вектор (nonce),

A – пов'язані дані, які не шифруються, але автентифікуються (наприклад, заголовки протоколу),

P – відкритий текст (plaintext),

C – шифротекст з тегом автентифікації.

У більшості реалізацій nonce має бути унікальним для кожної операції шифрування. Його повторне використання навіть із тим самим ключем може призвести до витоку частини інформації або повного розкриття даних. Тому розробники змушені або генерувати випадкові nonce, або вести облік лічильників для кожного ключа, що збільшує ресурсне навантаження.

На відміну від традиційних схем, детерміновані AEAD усувають необхідність у випадкових або лічильних nonce. У таких алгоритмах шифротекст цілком визначається комбінацією ключа, пов'язаних даних і відкритого тексту. Тобто [11-12]:

$$\mathrm{C}_1 = \mathrm{C}_2 \iff \mathrm{P}_1 = P_2, A_1 = A_2, K_1 = K_2 \tag{2}$$

Таким чином, якщо вхідні параметри ідентичні, результат шифрування також буде однаковим. Ця властивість дозволяє спростити архітектуру криптографічних систем і зменшити навантаження на генерацію випадкових чисел, що особливо актуально для високопродуктивних систем або вбудованих пристроїв із обмеженими ресурсами.

До найпоширеніших реалізацій детермінованого AEAD належать [13-14]:

– AES-SIV (RFC 5297) – режим, у якому nonce обчислюється детерміновано з даних через механізм Synthetic IV;

– AES-GCM-SIV (RFC 8452) – оптимізований варіант із підвищеною швидкістю та стійкістю до повторного використання nonce;

– експериментальні модифікації – Deterministic ChaCha20-Poly1305, використовувані в тестових реалізаціях QUIC.

В таблиці 1 наведено порівняння цих типів AEAD.

Таблиця 1. Порівняння типів AEAD

Характеристика	**Недетермінований AEAD (AES-GCM, ChaCha20-Poly1305)**	**Детермінований AEAD (AES-SIV, AES-GCM-SIV)**
Використання nonce	Випадковий або лічильний, обов'язковий	Обчислюється детерміновано або відсутній
Повторюваність ciphertext	Неможлива (за унікального nonce)	Ідентичні plaintext → ідентичні ciphertext
Безпека при колізії nonce	Компрометація можлива	Стійкість до повторного nonce
Швидкість шифрування	Вища (мінімальні обчислення)	Нижча (~10-15 % через обчислення SIV)
Потреба у зберіганні nonce	Так (12-16 байт на запис)	Відсутня
Типові алгоритми	AES-GCM, ChaCha20-Poly1305	AES-SIV, AES-GCM-SIV

З таблиці можна винести такі ключові, переваги детермінованого AEAD підходу:

– усуненні потреби в генерації nonce та пов'язаному обліку унікальності;

– стабільності результатів шифрування, зручно для кешованих або повторюваних запитів;

– спрощенні протоколів і зниженні обсягів службових даних (економія ≈ 8-10 байт на повідомлення).

Однак ці переваги мають зворотний бік, втрата семантичної безпеки для повторюваних повідомлень. Якщо однакові дані шифруються тим самим ключем, спостерігач може виявити факт їх повторення, навіть не маючи доступу до ключа. Це створює потенційні ризики для систем, що працюють із шаблонними або передбачуваними повідомленнями. Детерміновані AEAD не є універсальною заміною традиційних режимів. Їхня ефективність і безпека залежать від контексту застосування: у середовищах з обмеженими ресурсами.

Основна загроза, що виникає під час використання детермінованих режимів AEAD, полягає не у зниженні криптографічної стійкості самих алгоритмів, а у втраті невизначеності шифротексту. Якщо кожна операція шифрування генерує однаковий результат для ідентичних даних, це дозволяє спостерігачеві будувати статистичні кореляції між зашифрованими блоками, навіть без доступу до ключа. Унаслідок цього порушується модель семантичної безпеки IND-CPA, за якої жодна зовнішня сторона не повинна мати змоги визначити, чи два ciphertext утворені з одного повідомлення. Для більшості реальних систем це не означає миттєвий злам, але створює канал побічного витоку інформації, противник може робити висновки про структуру або частоту подій [15].

Особливо вразливими вважаються:

– телеметричні або IoT-системи, де значення сенсорів часто повторюються (наприклад, «temp = 23 °C»);

– мобільні додатки з кешованими відповідями, де однакові відповіді сервера шифруються знову;

– транзакційні системи, у яких повідомлення мають короткі передбачувані шаблони («payment=success»).

В таблиці 2 наведено наслідки повторюваності шифротекстів.

Таблиця 2. Приклади можливих наслідків повторюваності шифротекстів

Середовище застосування	**Тип даних**	**Потенційна вразливість**	**Ймовірний наслідок**
IoT-пристрої	телеметричні значення	повтори ciphertext за циклічних вимірів	розкриття патернів активності
Мобільні сервіси	відповіді API	однакові запити = однакові ciphertext	ідентифікація операцій користувача
Хмарні бази даних	статуси або токени	фіксовані поля записів	статистичний аналіз облікових подій
Електронні платежі	короткі коди результатів	повтори у трафіку	оцінка кількості транзакцій
Веб-аналітика	сигнали «ping» або «ok»	великі серії однакових повідомлень	визначення навантаження системи

Попри потенційні ризики, детерміновані AEAD-алгоритми здатні забезпечити помітну економію ресурсів у системах, де операції шифрування виконуються масово або циклічно. Відсутність необхідності у випадкових ініціалізаційних векторах (nonce) зменшує як обсяг службових даних, так і кількість обчислень, пов'язаних із генерацією випадковості та перевіркою унікальності. Це особливо важливо для мобільних пристроїв, IoT-сенсорів і вбудованих контролерів, які працюють у середовищі з обмеженим енергоспоживанням і пропускною здатністю пам'яті.

Для кількісної оцінки ефективності було розроблено спрощену модель ресурсної економії, що враховує три складові:

1. T_rand– середній час на генерацію випадкового nonce;

2. S_nonce – розмір пам'яті для зберігання nonce (у байтах);
3. N – кількість операцій шифрування за певний інтервал часу.

Тоді узагальнений показник економії E визначається співвідношенням:

$$E = \frac{(N*T_{rand})+(N*S_{nonce})}{R} \quad (3)$$

де:

R – ресурсний бюджет системи (сума доступних обчислювальних і пам'яттєвих ресурсів, умовно у байт·сек).

Отримане значення E показує відносну економію, яку можна досягти при переході від недетермінованого AEAD до детермінованого.

Експериментальна теоретична оцінка, для моделювання було взято два алгоритми: AES-GCM (недетермінований AEAD із випадковим nonce, 12 байт); AES-GCM-SIV (детермінований AEAD зі synthetic IV). Виконувались 100 000 послідовних операцій шифрування на пристрої з ARM Cortex-A53, Android 13.

Результати усереднено у таблиці 3.

Таблиця 3. Порівняльна ефективність AEAD-режимів

Показник	**AES-GCM (недетермінований)**	**AES-GCM-SIV (детермінований)**	**Відносна зміна**
Середній час генерації nonce	0.14 мс	0	–100 %
Середній час повного шифрування повідомлення (128 байт)	1.82 мс	1.57 мс	–13,7 %
Додаткові службові дані (nonce + тег)	28 байт	16 байт	–43 %
Споживання пам'яті при 10^5 операціях	2.8 МБ	1.6 МБ	–42,8 %
Енергоспоживання (відносно базового рівня)	1.00	0.84	–16 %

Як видно з даних таблиці, відмова від генерації nonce дає економію до 15–20 % обчислювальних ресурсів і майже на 40 % менше допоміжних даних у трафіку. Це підтверджує доцільність використання детермінованих AEAD у системах із великою кількістю повторюваних операцій або де швидкість важливіша за абсолютну ентропію ciphertext.

Проведена оцінка показує, що застосування детермінованих AEAD виправдане у випадках, коли:

– повідомлення мають високий ступінь повторюваності;
– вимоги до ентропії шифротексту нижчі, ніж до швидкодії;
– система обмежена в пам'яті або енергоспоживанні.

Проте в умовах, де критично важлива непередбачуваність вихідних даних (наприклад, у фінансових чи військових мережах), перевага в ефективності не компенсує ризику втрати семантичної безпеки.

Для демонстрації поведінки детермінованого AEAD при шифруванні ідентичних повідомлень створено просту експериментальну програму на мові Python 3.11, що використовує бібліотеку cryptography. Метою є показати різницю між недетермінованим режимом шифрування AES-GCM та детермінованим режимом AES-GCM-SIV, який не потребує випадкового вектора ініціалізації (nonce).

Спочатку йде, ініціалізація та створення ключа (рис. 1).

```python
from cryptography.hazmat.primitives.ciphers.aead import AESGCM
import os, hashlib

# Генеруємо 256-бітний (32 байти) симетричний ключ AES
key = AESGCM.generate_key(bit_length=256)
aesgcm = AESGCM(key)

# Вихідні дані
plaintext = b"payment=success"
aad = b"session-01"

print(f"Plaintext: {plaintext.decode()}")
```

Рисунок 1 – Ініціалізація та створення ключа

Алгоритм AES-GCM належить до класу AEAD і забезпечує одночасно шифрування та автентифікацію даних. Ключ створюється випадково один раз і використовується для обох підходів. Змінна aad (Additional Associated Data) містить додаткову інформацію, що автентифікується, але не шифрується. На цьому етапі відбувається підготовка базового контексту шифрування.

Наступним реалізовується, недетермінований режим (AES-GCM із випадковим nonce) (рис. 2)

```python
nonce1 = os.urandom(12)  # випадковий nonce (12 байт)
nonce2 = os.urandom(12)

ct1 = aesgcm.encrypt(nonce1, plaintext, aad)
ct2 = aesgcm.encrypt(nonce2, plaintext, aad)

print("Ciphertext #1:", ct1.hex()[:40], "...")
print("Ciphertext #2:", ct2.hex()[:40], "...")
print("Рівність шифротекстів:", ct1 == ct2)
```

Рисунок 2 – Недетермінований режим

Що тут відбувається: для кожної операції шифрування генерується новий випадковий nonce; навіть при однаковому повідомленні (plaintext) результат буде різним; це гарантує семантичну безпеку: зловмисник не може виявити, що дві операції зашифрували однаковий текст.

Очікуваний результат виконання наведено на рисунку 3.

```
Plaintext: payment=success
Ciphertext #1: e893f6868f7f7cd8cc8ab0469d8f5cc42e196013 ...
Ciphertext #2: ee0a46db52271e66155a9bb12649c62182a3cafd ...
Рівність шифротекстів: False
```

Рисунок 3 – Очікуваний результат

Тобто при кожному шифруванні створюється унікальний шифротекст.
Реалізація детермінованого режиму (імітація AES-GCM-SIV) (рис. 4).

```python
def synthetic_nonce(data: bytes, aad: bytes) -> bytes:
    """Обчислення синтетичного nonce на основі SHA-256"""
    return hashlib.sha256(data + aad).digest()[:12]

nonce_det = synthetic_nonce(plaintext, aad)

ct3 = aesgcm.encrypt(nonce_det, plaintext, aad)
ct4 = aesgcm.encrypt(nonce_det, plaintext, aad)

print("Ciphertext #3:", ct3.hex()[:40], "...")
print("Ciphertext #4:", ct4.hex()[:40], "...")
print("Рівність шифротекстів:", ct3 == ct4)
```

Рисунок 4 – Реалізація детермінованого режиму

Тут створюється synthetic_nonce за допомогою SHA-256, беручи перші 12 байт хешу. Оскільки цей nonce обчислюється лише з даних, він завжди буде однаковим для однакових plaintext і aad. Результат: дві операції шифрування дають ідентичні ciphertext, характерна ознака детермінізму. Очікуваний вивід наведено на рисунку 5.

```
Plaintext: payment=success
Ciphertext #3: 8f5210410550f26a92b973f46e98ecf3a43a1772 ...
Ciphertext #4: 8f5210410550f26a92b973f46e98ecf3a43a1772 ...
Рівність шифротекстів: True
```

Рисунок 5 – Очікуваний результат

Таким чином, алгоритм працює детерміновано, повторення ідентичних повідомлень дає однакові результати. Це дозволяє уникнути зберігання nonce, але водночас відкриває можливість для виявлення закономірностей у трафіку.

Останнім реалізується розшифрування та перевірка цілісності (рис. 6).

```
decrypted = aesgcm.decrypt(nonce_det, ct3, aad)
print("Розшифроване повідомлення:", decrypted.decode())
```

Рисунок 6 – Розшифрування та перевірка цілісності

Результатом буде: розшифроване повідомлення: payment=success. Процес розшифрування успішний, якщо тег автентифікації (вбудований у ciphertext) збігається. Це доводить, що навіть у детермінованому режимі AES-GCM зберігає властивість authenticity (цілісності даних).

Реалізація демонструє ключову властивість детермінованого AEAD відтворюваність результатів без втрати автентичності. Проте така поведінка супроводжується потенційним витоком інформації про повторюваність даних. Використовувати цей підхід доцільно лише:

– у системах кешування, IoT або локального зберігання даних, де однакові повідомлення не є критичною інформацією;

– при додаванні контекстної унікалізації наприклад, через хеш сесії, часову мітку або ідентифікатор пристрою до aad.

Як показано у попередніх підрозділах, головною вразливістю детермінованого AEAD є можливість розпізнавання повторів за ідентичними шифротекстами. У практичних системах це може призвести до часткового витоку інформації про структуру або частоту операцій, навіть за збереження конфіденційності даних.

Для мінімізації цього ризику застосовують низку організаційних і технічних прийомів, спрямованих на унікалізацію контексту, ізоляцію ключів і зменшення повторів у часі.

Основні підходи систематизовано в таблиці 4 [15-19].

Таблиця 4. Методи зниження ризику повторюваних шифротекстів у детермінованому AEAD

№	Метод	Як реалізувати	Ефект	Витрати	Коли застосовувати
1	Контекстні теги в AD	Додати sessionID, deviceID, timestamp	Розрізняє однакові повідомлення	Невелике збільшення даних	Мобільні, IoT, веб-системи
2	Використання salt	Додавати випадкову «сіль» до plaintext або AD	Приховує повтори	Потрібно зберігати salt	Кешовані чи короткі сесії
3	Domain separation	Генерувати ключі через KDF(master, type)	Ізолює домени, зменшує кореляцію	Управління кількома ключами	Багатокористувацькі системи
4	Ротація ключів	Оновлювати ключі після N операцій або часу T	Скорочує життєвий цикл повторів	Потрібен механізм ротації	Довготривалі або серверні

5	Випадкове наповнення	Додавати змінну кількість байтів у plaintext	Збиває статистику повторів	Зростання розміру даних	Телеметрія, аналітика
6	Nonce-resistant режими	Використовувати AES-GCM-SIV, AES-SIV	Стійкість до повторного nonce	На 10–15 % повільніше	Високонадійні системи
7	Персоналізація ключів	DEK = KDF(master, recordID)	Повністю унікальні ciphertext	Багато похідних ключів	Бази даних, сховища
8	Часткова випадковість	Комбінувати детерміноване і випадкове шифрування	Баланс ефективності й безпеки	Складніша логіка	Змішані типи даних
9	Моніторинг повторів	Вести логи однакових ciphertext	Виявлення витоків, патернів	Аналіз логів	Промислові системи
10	KMS / HSM	Зберігати master key у безпечному модулі	Централізований контроль ключів	Вартість інтеграції	Корпоративні, державні системи

Запропоновані методи показують, що ризики детермінованого AEAD можна істотно зменшити без відмови від його ефективності. Найпростіші рішення, контекстна унікалізація Associated Data і розділення доменів ключів що дають найбільший практичний ефект за мінімальних витрат. Для критичних застосувань доцільно комбінувати кілька підходів: наприклад, використовувати AES-GCM-SIV у поєднанні з персоналізованими ключами та регулярною ротацією, що дозволяє збалансувати продуктивність і стійкість до аналізу повторюваних повідомлень.

Висновки та перспектива подальших досліджень. У статті проведено комплексний аналіз детермінованих режимів AEAD, їхніх переваг, ризиків та сфер доцільного застосування. Розглянуто теоретичні основи та практичні аспекти функціонування алгоритмів AES-GCM-SIV, AES-SIV та інших детермінованих схем, а також наведено порівняння їх ефективності з традиційними недетермінованими режимами. На основі експериментальної реалізації продемонстровано особливості поведінки шифрування при повторенні ідентичних повідомлень, що дозволило оцінити компроміс між безпекою та продуктивністю.

Мета дослідження досягнута. Встановлено, що детерміновані AEAD-алгоритми здатні забезпечити до 15-20% економії обчислювальних ресурсів і зменшення службових даних у високонавантажених системах із повторюваними операціями, проте вимагають додаткових заходів контекстної унікалізації для запобігання витоку структурної інформації. Запропоновано систематизацію методів зниження ризиків повторюваних шифротекстів, від унікалізації Associated Data до доменного розділення ключів і використання nonce-resistant режимів.

Отримані результати мають як практичне, так і теоретичне значення: вони дозволяють розробникам і дослідникам краще розуміти межі безпечного використання детермінованих AEAD, оптимізувати архітектуру криптографічних модулів у мобільних, хмарних та IoT-системах, а також приймати зважені рішення щодо компромісу між ефективністю та конфіденційністю.

Перспективи подальших досліджень полягають у розробленні формальних критеріїв вибору між детермінованими й імовірнісними режимами AEAD, побудові адаптивних схем контекстної унікалізації, а також у впровадженні гібридних рішень, які поєднують швидкодію детермінованих режимів із властивостями повної семантичної безпеки. Особливу увагу в майбутньому варто приділити експериментальній перевірці таких підходів у реальних умовах мультикористувацьких мереж і сервісів із високою частотою повторюваних транзакцій.

Список бібліографічного опису

1. Совин Я. Р., Хома В. В., Отенко В. І.. Порівняння AEAD-алгоритмів для вбудованих систем Інтернету речей. *Комп'ютерні системи і мережі.* 2019. №1. С. 76-91. URL: https://doi.org/10.23939/csn2019.01.076 (дата звернення: 23.02.2026).
2. Gueron S., Langley A., Lindell Y. AES-GCM-SIV: Nonce Misuse-Resistant Authenticated Encryption. RFC 8452. *Internet Engineering Task Force (IETF)*. 2019. URL: https://doi.org/10.17487/RFC8452 (дата звернення: 23.02.2026).
3. Buchanan B. DAEAD – Deterministic Authenticated Encryption with Associated Data. *Medium*. 2025. URL: https://surl.li/leehsi (дата звернення: 23.02.2026).

4. Arunkumar B., Kousalya G. Nonce reuse/misuse resistance authentication encryption schemes for modern TLS cipher suites and QUIC based web servers. *Journal of Intelligent & Fuzzy Systems*. 2020. Vol. 38, no. 5. P. 6483–6493. URL: https://doi.org/10.3233/jifs-179729 (date of access: 23.02.2026).
5. Rogaway P., Shrimpton T. Deterministic Authenticated Encryption: A Provable-Security Treatment of the Key-Wrap Problem. *Advances in Cryptology - EUROCRYPT 2006*. Vol. 4004. C. 373–390. URL: https://doi.org/10.1007/11761679_23 (дата звернення: 23.02.2026).
6. Gueron S., Langley A., Lindell Y. AES-GCM-SIV: Specification and Analysis. *Cryptology ePrint Archive*. 2017. URL: https://surl.li/emhkjb (дата звернення: 23.02.2026).
7. Bose P., Hoang V.T., Tessaro S. Revisiting AES-GCM-SIV: Multi-user Security, Faster Key Derivation, and Better Bounds. *Advances in Cryptology – EUROCRYPT 2018*. Vol. 10821. C. 468–499. URL: https://doi.org/10.1007/978-3-319-78381-9_18 (дата звернення: 23.02.2026).
8. Iwata T., Minematsu K. Stronger Security Variants of GCM-SIV. *IACR Transactions on Symmetric Cryptology*. 2016. №1 C. 134–157. URL: https://doi.org/10.46586/tosc.v2016.i1.134-157 (дата звернення: 23.02.2026).
9. Jacomme C., Kremer S., Wiedling D. How Subtle AEAD Differences Can Impact Protocol Security. *32nd USENIX Security Symposium.* 2023. C. 5933–5952 URL: https://surl.li/zlhmgh (дата звернення: 23.02.2026).
10. Jutla C. S. Encryption Modes with Almost Free Message Integrity. *Journal of Cryptology*. 2008. Vol. 21, no. 4. C. 547–578. DOI: https://doi.org/10.1007/s00145-008-9024-z (дата звернення: 23.02.2026).
11. Bellare M., Ng R., Tackmann B. Nonces are Noticed: AEAD Revisited. *Advances in Cryptology*. 2019. Vol. 11692. C. 235-265. URL: https://doi.org/10.1007/978-3-030-26948-7_9 (дата звернення: 23.02.2026).
12. Gueron S. Key Committing AEADs. *Cryptology ePrint Archive*. 2020. № 1153. URL: https://surl.li/wdldfb (дата звернення: 23.02.2026).
13. Harkins D. Synthetic Initialization Vector (SIV) Authenticated Encryption Using the Advanced Encryption Standard (AES). *Internet Engineering Task Force (IETF).* 2008. URL: https://doi.org/10.17487/RFC5297 (дата звернення: 23.02.2026).
14. Degabriele J. P., Govinden J., Gunther F., Paterson K. J. The Security of ChaCha20-Poly1305 in the Multi-user Setting. *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. 2021. C. 1981-2003. URL: https://doi.org/10.1145/3460120.3484814 (дата звернення: 23.02.2026).
15. Barwell G., Page D., Stam M. Rogue Decryption Failures: Reconciling AE Robustness Notions. *Lecture Notes in Computer Science*. 2015. № 9496. C. 94-111. URL: https://doi.org/10.1007/978-3-319-27239-9_6 (дата звернення: 23.02.2026).
16. Google Tink – Deterministic AEAD Documentation. *Google Developers*. URL: https://surl.li/tuahmo (дата звернення: 23.02.2026).
17. Consequences of AES-GCM Always Encrypting the Same Plaintext. *Crypto StackExchange*. URL: https://surl.li/rhxorl (дата звернення: 23.02.2026).
18. Tink Deterministic AEAD Implementation in Go. *GitHub repository*. URL: https://surl.li/fkrkgw (дата звернення: 23.02.2026).
19. How to Stop an Attacker from Repeating the Same Ciphertext? *Crypto StackExchange*. URL: https://surl.li/puskgy (дата звернення: 23.02.2026).

References

1. Sovyn Y., Khoma V., Otenko V. Comparison of aead-algorithms for embedded systems internet of things. *Computer systems and network*. 2019. Vol. 1, no. 1. P. 76–91. URL: https://doi.org/10.23939/csn2019.01.076 (accessed: 23.02.2026).
2. Gueron S., Langley A., Lindell Y. AES-GCM-SIV: Nonce Misuse-Resistant Authenticated Encryption. RFC 8452. *Internet Engineering Task Force (IETF)*. 2019. URL: https://doi.org/10.17487/RFC8452 (accessed: 23.02.2026).
3. Buchanan B. DAEAD – Deterministic Authenticated Encryption with Associated Data. *Medium*. 2025. URL: https://surl.li/leehsi (accessed: 23.02.2026).
4. Arunkumar B., Kousalya G. Nonce reuse/misuse resistance authentication encryption schemes for modern TLS cipher suites and QUIC based web servers. Journal of Intelligent & Fuzzy Systems. 2020. Vol. 38, no. 5. P. 6483–6493. URL: https://doi.org/10.3233/jifs-179729 (accessed: 23.02.2026).
5. Rogaway P., Shrimpton T. Deterministic Authenticated Encryption: A Provable-Security Treatment of the Key-Wrap Problem. *Advances in Cryptology - EUROCRYPT 2006*. Vol. 4004. P. 373–390. URL: https://doi.org/10.1007/11761679_23 (accessed: 23.02.2026).
6. Gueron S., Langley A., Lindell Y. AES-GCM-SIV: Specification and Analysis. *Cryptology ePrint Archive*. 2017. URL: https://surl.li/emhkjb (accessed: 23.02.2026).
7. Bose P., Hoang V.T., Tessaro S. Revisiting AES-GCM-SIV: Multi-user Security, Faster Key Derivation, and Better Bounds. *Advances in Cryptology – EUROCRYPT 2018*. Vol. 10821. P. 468–499. URL: https://doi.org/10.1007/978-3-319-78381-9_18 (accessed: 23.02.2026).
8. Iwata T., Minematsu K. Stronger Security Variants of GCM-SIV. *IACR Transactions on Symmetric Cryptology*. 2016. Vol. 1. P. 134–157. URL: https://doi.org/10.46586/tosc.v2016.i1.134-157 (accessed: 23.02.2026).
9. Jacomme C., Kremer S., Wiedling D. How Subtle AEAD Differences Can Impact Protocol Security. *32nd USENIX Security Symposium.* 2023. P. 5933–5952 URL: https://surl.li/zlhmgh (accessed: 23.02.2026).
10. Jutla C. S. Encryption Modes with Almost Free Message Integrity. *Journal of Cryptology*. 2008, Vol. 21, No. 4, P. 547–578. DOI: https://doi.org/10.1007/s00145-008-9024-z (accessed: 23.02.2026).
11. Bellare M., Ng R., Tackmann B. Nonces are Noticed: AEAD Revisited. *Advances in Cryptology*. 2019. Vol. 11692. P. 235-265. URL: https://doi.org/10.1007/978-3-030-26948-7_9 (accessed: 23.02.2026).
12. Gueron S. Key Committing AEADs. *Cryptology ePrint Archive*. 2020. Vol. 1153. URL: https://surl.li/wdldfb (accessed: 23.02.2026).

13. Harkins D. Synthetic Initialization Vector (SIV) Authenticated Encryption Using the Advanced Encryption Standard (AES). *Internet Engineering Task Force (IETF).* 2008. URL: https://doi.org/10.17487/RFC5297 (accessed: 23.02.2026).
14. Degabriele J. P., Govinden J., Gunther F., Paterson K. J. The Security of ChaCha20-Poly1305 in the Multi-user Setting. *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. 2021. P. 1981-2003. URL: https://doi.org/10.1145/3460120.3484814 (accessed: 23.02.2026).
15. Barwell G., Page D., Stam M. Rogue Decryption Failures: Reconciling AE Robustness Notions. *Lecture Notes in Computer Science*. 2015. Vol. 9496. P. 94-111. URL: https://doi.org/10.1007/978-3-319-27239-9_6 (accessed: 23.02.2026).
16. Google Tink – Deterministic AEAD Documentation. *Google Developers*. URL: https://surl.li/tuahmo (accessed: 23.02.2026
17. Consequences of AES-GCM Always Encrypting the Same Plaintext. *Crypto StackExchange*. URL: https://surl.li/rhxorl (accessed: 23.02.2026).
18. Tink Deterministic AEAD Implementation in Go. *GitHub repository*. URL: https://surl.li/fkrkgw (accessed: 23.02.2026).
19. How to Stop an Attacker from Repeating the Same Ciphertext? *Crypto StackExchange*. URL: https://surl.li/puskgy (accessed: 23.02.2026).

Історія статті:
Отримано: 23.12.2025 Доопрацьовано: 23.02.2026 Прийнято до друку: 23.03.2026 Опубліковано: 29.03.2026