УДК 004.056.5 DOI: 10.18372/2073-4751.86.21273

Дячук О. Ю.[1],
orcid.org/0000-0002-6996-4700,
e-mail: kkik_doyu@ztu.edu.ua,

Колощук М. С.[1],
orcid.org/0009-0001-5825-2054,
e-mail: kkik_kms@ztu.edu.ua,

Рудюк Б. М.[1],
orcid.org/0009-0005-0372-8886,
e-mail: kkik_rbm@ztu.edu.ua,

Квасніков В. П.[2], д.т.н.,
orcid.org/0000-0002-6525-9721,
e-mail: kvp@kai.edu.ua

ЕФЕКТИВНІСТЬ КРИПТОГРАФІЧНИХ ПРОТОКОЛІВ TLS 1.2 І TLS 1.3 У СУЧАСНИХ ВЕБ-ДОДАТКАХ: АНАЛІТИЧНИЙ ТА ПРАКТИЧНИЙ АСПЕКТИ

[1]Державний університет «Житомирська політехніка»
[2]Національний університет «Київський авіаційний інститут»

Вступ

У сучасному цифровому суспільстві веб-додатки є базовим середовищем для зберігання, обробки та обміну конфіденційними даними. В основі захисту цього середовища лежить криптографічний стек SSL/TLS (Secure Sockets Layer / Transport Layer Security), що забезпечує автентифікацію, конфіденційність і цілісність переданої інформації [13]. За даними Google Transparency Report, станом на 2024 рік понад 95 % трафіку Chrome передається через HTTPS [10], що підкреслює центральну роль TLS у сучасній цифровій інфраструктурі.

Питання ефективності SSL/TLS має подвійний характер: з одного боку – рівень криптографічного захисту, з іншого – вплив алгоритмів шифрування на продуктивність і масштабованість систем. У добу мікросервісних архітектур і хмарних платформ шифрування застосовується не лише між клієнтом і сервером, а й між усіма внутрішніми компонентами, включаючи пристрої Інтернету речей [9].

Еволюція протоколів демонструє поступовий перехід від реактивного усунення вразливостей до побудови превентивних криптографічних архітектур. SSL 3.0 виявився вразливим до атаки POODLE, TLS 1.0 та SSL 3.0 – до BEAST, атака CRIME уразила механізм компресії TLS, а Lucky13 продемонструвала структурні слабкості CBC-режимів, що зберігалися і в TLS 1.2. TLS 1.2 інтегрував алгоритми AEAD як відповідь на ці загрози, а TLS 1.3 (RFC 8446, IETF, 2018) [1] радикально переробив архітектуру протоколу: скоротив рукостискання до одного раунду, усунув статичний обмін ключами RSA та впровадив лише сучасні шифри.

Мета

Метою роботи є експериментальне порівняльне дослідження ефективності криптографічних протоколів TLS 1.2 та TLS 1.3 у сучасних веб-додатках за критеріями продуктивності та криптографічної стійкості, розробка практичного інструментарію безпечного розгортання TLS для DevSecOps-інженерів, а також оцінка перспектив переходу до квантово-стійких гібридних конфігурацій на основі актуальних стандартів NIST і IETF.

Аналіз останніх досліджень і публікацій

Питання ефективності протоколів TLS є предметом активних наукових досліджень, які систематизуються за трьома напрямками: стандартизація протоколів IETF, вимірювання масштабу розгортання в мережі Інтернет і оцінка продуктивності на різних апаратних платформах.

Rescorla (RFC 8446, 2018) [1] визначив архітектурні засади TLS 1.3: перехід до 1-RTT рукостискання, обов'язковий ECDHE-обмін ключами та виключно AEAD-шифри. Dierks та Rescorla (RFC 5246, 2008) [14] описали архітектуру TLS 1.2, що досі широко застосовується у виробничих середовищах. Tschofenig та Fossati [6] у RFC 7925 (IETF, 2016) визначили профілі TLS/DTLS для пристроїв Інтернету речей із обмеженими ресурсами, наголосивши на важливості ChaCha20-Poly1305 як оптимального для платформ без апаратного прискорення AES.

Holz et al. [2] (ACM SIGCOMM Computer Communication Review, 2020) виконали масштабне вимірювання розгортання TLS 1.3, встановивши, що TLS 1.3 розгортається значно швидше, ніж TLS 1.2 у свій час. Sullivan (2018) [5] надав детальний технічний аналіз архітектурних змін RFC 8446, а Mozilla TLS Telemetry зафіксувала стрімке зростання частки з'єднань через TLS 1.3 після його стандартизації. Aviram et al. [7] задокументували атаку DROWN (USENIX Security, 2016), що стала каталізатором відмови від SSLv2 та SSLv3 у глобальній TLS-екосистемі.

Hasan et al. [3] (JISIS, 2025) підтвердили, що TLS 1.3 перевершує всі конфігурації TLS 1.2 завдяки меншій кількості round-trips, а механізми OCSP stapling та session resumption додатково знижують TTFB у мобільних мережах. Restuccia et al. [4] експериментально встановили, що TLS 1.3 у певних конфігураціях знижує накладні витрати та енергоспоживання на мікроконтролерах. Clark та van Oorschot [8] (IEEE S&P, 2013) систематизували виклики безпеки SSL/HTTPS та моделі довіри сертифікатів, що сформували основу сучасних вимог до конфігурації TLS. Незважаючи на значний обсяг досліджень, більшість публікацій або обмежуються теоретичним аналізом, або використовують закриту хмарну інфраструктуру. Відсутність відкритих відтворюваних мікроекспериментів і зумовила проведення власного дослідження.

Основна частина

Апаратно-програмне середовище: операційна система – Ubuntu Server 22.04 LTS (x86-64); веб-сервер – Nginx 1.18.0 + OpenSSL 3.0.2 (Nginx встановлено з офіційного репозиторію Ubuntu 22.04 LTS; для отримання версії 1.22+ використовується репозиторій nginx.org); сертифікати – ECDSA P-256 (основний тест) та RSA 2048 (додатковий); інструменти – wrk v4.2 (HTTP-навантаження), Wireshark 4.0 (аналіз рукостискань), htop 3.2 (моніторинг CPU) – усі open-source; мережа – ізольована локальна мережа 1 Гбіт/с, 50 паралельних з'єднань (8 потоків wrk).

Конфігурації для порівняння: TLS 1.2 із набором шифрів ECDHE-ECDSA-AES256-GCM-SHA384; TLS 1.3 із шифром ChaCha20-Poly1305 (TLS_CHACHA20_POLY1305_SHA256) ; TLS 1.3 із шифром AES-256-GCM (TLS_AES_256_GCM_SHA384) – для порівняння алгоритмів усередині TLS 1.3.

Тестування проводилося на фізичному сервері з процесором класу Intel Core i7 з підтримкою апаратного прискорення AES-NI та оперативною пам'яттю обсягом 16 ГБ. Клієнтське навантаження генерувалося з окремої машини в межах локальної мережі для мінімізації мережевого джиттера. Перед кожною серією вимірювань виконувалося прогрівання тривалістю 30 с, сторонні процеси на сервері зупинялися. Навантажувальне тестування виконувалося командою: *wrk -t8 -c50 -d60s https://192.168.1.10/*. Час рукостискання вимірювався окремо командою: *openssl s_time -connect 192.168.1.10:443 -new -time 10*.

Для кожної конфігурації виконувалося три незалежні серії тестування тривалістю 60 с із 50 паралельними з'єднаннями та 8 потоками wrk. Навантаження генерувалося на статичну сторінку розміром 512 КБ; загальна кількість оброблених запитів фіксувалася за результатами wrk і в усіх серіях перевищувала 5 000. Зафіксовано: TLS handshake time – окремий замір через openssl s_time [15]; end-to-end latency (TCP + TLS handshake + обробка запиту + передача відповіді) – через wrk; throughput та CPU utilization. Для кожної конфігурації обчислювалося середнє арифметичне $\overline{M}$ трьох незалежних серій тривалістю 60 с, стандартне відхилення σ та довірчий інтервал за рівня довіри 95 % на основі t-розподілу Стьюдента ($df = n-1 = 2$, критичне значення $t_{0,025;\,2} = 4{,}303$):

$$\overline{M} \pm t_{0,025;2} \cdot \frac{\sigma}{\sqrt{n}} = \overline{M} \pm 4{,}303 \cdot \frac{\sigma}{\sqrt{3}},$$

де $n = 3$ – кількість серій. Як індикатор відтворюваності використовувався коефіцієнт варіації: $CV = \frac{\sigma}{\overline{M}} \cdot 100\%$. Значення CV не перевищувало 4–5 % для всіх метрик, що підтверджує стабільність вимірювального середовища.

Конфігурація «TLS 1.2 + AES-256-GCM» тестувалася у стандартному режимі з апаратним прискоренням AES-NI (baseline типового x86-серверу). Для конфігурацій «TLS 1.3 + ChaCha20-Poly1305» та «TLS 1.3 + AES-256-GCM», орієнтованих на мобільно-IoT-сценарій, AES-NI було програмно відключено через змінну середовища OPENSSL_ia32cap – стандартну методику OpenSSL для моделювання платформ без апаратного прискорення AES (мобільні пристрої, ARM-мікроконтролери без crypto extensions). Така постановка відображає реальну асиметрію розгортання: серверна сторона майже завжди має AES-NI, а клієнтська сторона часто ні. Для ізольованої оцінки внеску архітектурних змін протоколу TLS 1.3 слід порівнювати TLS 1.2 + AES-GCM з TLS 1.3 + AES-GCM в однакових умовах без AES-NI.

Результати трьох серій вимірювань систематизовано у табл. 1. Дані засвідчують суттєву перевагу TLS 1.3 за всіма вимірюваними показниками.

Таблиця 1. Порівняльні показники продуктивності TLS 1.2 та TLS 1.3

Показник	TLS 1.2 AES-256-GCM (з AES-NI, baseline)	TLS 1.3 ChaCha20-Poly1305 (без AES-NI)	TLS 1.3 AES-256-GCM (без AES-NI, control)	Зміна (1.2→1.3)
TLS handshake time (openssl s_time), мс	4,8 ± 0,2	3,0 ± 0,1	3,2 ± 0,1	−37 %
TTFB (wrk, 50 з'єднань), мс	48,3 ± 1,8	30,1 ± 1,2	31,8 ± 1,3	−37 %
Throughput, Мбіт/с	430 ± 12	515 ± 14	498 ± 13	+20 %
CPU utilization, %	~32	~26	~28	−18 %
Пакетів у handshake, шт.	7–9	5–7	5–7	−2 пак.
TTFB при 0-RTT/PSK resumption, мс	–	~16	~17	~2× швидше

Примітка. Порівняння «TLS 1.2 → TLS 1.3 Ch» є навмисно асиметричним і відображає типовий реальний розрив між серверним та мобільним розгортанням. Перехід 32 % → 28 % не слід трактувати як ізольований ефект TLS 1.3, оскільки конфігурації відрізняються не лише версією протоколу, а й режимом апаратного прискорення та вибором шифронабору. Це порівняння має індикативний характер і демонструє загальну різницю між обраними сценаріями. Перехід 28 % → 26 % (~7 %) відображає перевагу ChaCha20-Poly1305 над AES-GCM у програмному режимі – ізольованіше порівняння в однакових умовах без AES-NI.

Wireshark-аналіз підтвердив, що фаза рукостискання TLS 1.3 містить на два пакети менше, ніж TLS 1.2. Чистий час TLS-рукостискання склав 4,8 мс (TLS 1.2) проти 3,0 мс (TLS 1.3) – скорочення на 37 %. Абсолютна економія часу рукостискання складає 1,8 мс (4,8 − 3,0 мс), що пояснюється переходом до 1-RTT handshake та паралелізацією криптографічних операцій у TLS 1.3.

Загальна затримка TTFB зменшилась з 48,3 до 30,1 мс.

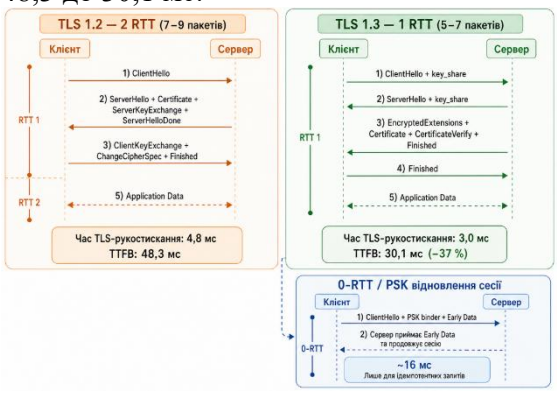

Рис. 1. Діаграма послідовності TLS-рукостискання: TLS 1.2 (2 RTT) vs TLS 1.3 (1 RTT)

Механізм 0-RTT Session Resumption дозволив відновити сесію за ~16 мс – у ~2 рази швидше повного TLS 1.3-рукостискання (30,1 мс TTFB) – без повторного узгодження ключів. Це особливо важливо для веб-додатків із великою кількістю короткострокових з'єднань – REST API, мікросервіси, WebSocket. Водночас 0-RTT доцільно застосовувати лише для ідемпотентних операцій, оскільки early data не забезпечує повного захисту від replay-атак без додаткових прикладних або інфраструктурних механізмів захисту. Порівняння двох шифрів усередині TLS 1.3 в однакових програмних умовах без AES-NI показало: ChaCha20 випереджає AES-GCM за throughput на ~3 % (515 vs. 498 Мбіт/с) та споживає на ~7 % менше ресурсів CPU.

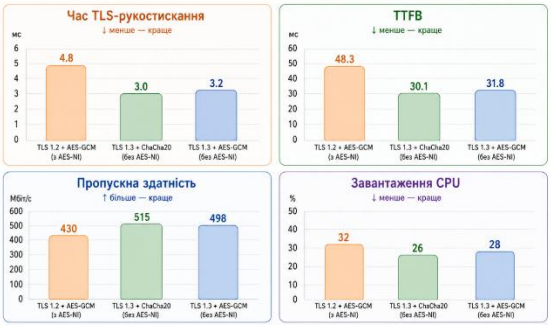

Рис. 2. Порівняльні показники продуктивності TLS 1.2 та TLS 1.3

Отримані результати підтверджують, що ефективність протоколів TLS визначається не лише алгоритмами шифрування, а й архітектурними рішеннями. Відповідно до RFC 7925 [6] та результатів Restuccia et al. [4], алгоритм ChaCha20-Poly1305 є оптимальним вибором для мобільних пристроїв і IoT. Натомість AES-256-GCM залишається ефективним рішенням для серверів із AES-NI, що підкреслює необхідність адаптивного підходу до конфігурації TLS.

Окремої уваги потребує питання стійкості до атак на сесійні ключі. Відмова від статичного RSA-обміну ключами у TLS 1.3 і обов'язкове використання ефемерних ключів ECDHE забезпечують властивість Perfect Forward Secrecy. У TLS 1.3 шифрується весь handshake після ServerHello, включно з сертифікатом і розширеннями, що значно ускладнює аналіз метаданих з'єднань для пасивного спостерігача. Механізм Encrypted Client Hello (ECH), що розвивається як окреме розширення TLS-екосистеми, потенційно дозволяє приховувати поле SNI, однак його практичне застосування залежить від підтримки клієнтами, серверами та DNS-інфраструктурою.

Наукова новизна роботи полягає у кількісній оцінці спільного впливу архітектури TLS 1.3, вибору шифронабору та наявності апаратного прискорення AES-NI на продуктивність веб-з'єднань. Через асиметричність тестових конфігурацій отримана декомпозиція має якісний, а не повністю ізольований експериментальний характер – що відсутнє у розглянутих публікаціях з даної тематики. Дослідження виконано на базі виключно open-source інструментарію (Nginx, OpenSSL, wrk, Wireshark) з повним розкриттям середовища, що забезпечує відтворюваність результатів. Додатково представлено кількісну оцінку постквантового overhead (ML-KEM-768: +2272 байти, <1 мс) у контексті практичного переходу на гібридні TLS-конфігурації відповідно до NIST FIPS 203 та IETF-драфтів.

Практичне значення результатів:

1. Оптимізація конфігурацій веб-серверів (Nginx, Apache, Caddy) через вибір шифрів відповідно до апаратного забезпечення.

2. Формування корпоративних TLS-політик відповідно до NIST SP 800-52r2 [12] та рекомендацій Mozilla [11].

3. Конфігурування криптографічного стека IoT-платформ з обмеженими ресурсами відповідно до [6], [9].

4. Вдосконалення DevSecOps-процесів через автоматизовану перевірку конфігурацій TLS у CI/CD-пайплайнах.

Для практичного впровадження розроблено чекліст безпечної конфігурації TLS (табл. 2), що охоплює 13 параметрів від версій протоколу до моніторингу в реальному часі.

Таблиця 2. Чекліст безпечної конфігурації TLS для веб-серверів (на прикладі Nginx)

Категорія	Перевірка	Цільове значення	Nginx директива	Ризик	Джерело
Версії протоколу	Відключити SSLv2/3, TLS 1.0/1.1	TLS 1.2 і TLS 1.3	ssl_protocols TLSv1.2 TLSv1.3;	POODLE, BEAST, DROWN	[1]
Версії протоколу	Пріоритет TLS 1.3	TLS 1.3 основний	ssl_prefer_server_ciphers off;	Втрата PFS	[1]
Набори шифрів	AEAD-шифри без CBC/RC4	AES-GCM, ChaCha20	ssl_ciphers ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384;	Lucky13, padding oracle	[12]
Обмін ключами	Лише ECDHE	ECDHE-ECDSA / ECDHE-RSA	ssl_ecdh_curve X25519:secp384r1;	Ретроспективне розшифрування	[1]
Сертифікати	Розмір ключа	RSA 3072 / ECDSA P-256	ssl_certificate_key server.key;	Підбір ключа	[12]
OCSP Stapling	Увімкнути	Активно	ssl_stapling on; ssl_stapling_verify on;	Витік метаданих	[11]
HSTS	Strict-Transport-Security	max-age≥31536000; includeSubDomains	add_header Strict-Transport-Security ...;	Downgrade-атаки	[24]
Сесійні ключі	Ротація session tickets	Добова ротація ключа	ssl_session_tickets on; ssl_session_timeout 1d;	Компрометація PSK	[1]
0-RTT	Тільки idempotent-запити	Валідація на рівні додатка	ssl_early_data on; (для класичного TLS 1.3 – працює; для HTTP/3/QUIC потрібен OpenSSL 3.5.1+ або quictls)	Replay-атаки	[1]
HTTP/2–HTTP/3	Сучасний транспорт	HTTP/3 або HTTP/2	listen 443 quic reuseport;	Втрата 0-RTT переваг	[25]
CSP	Content-Security-Policy	default-src 'self'	add_header Content-Security-Policy ...;	XSS через сторонній вміст	[26]
Моніторинг	Логування TLS-помилок	SIEM-інтеграція	error_log /var/log/nginx/ssl_error.log warn;	Непомітні атаки	[11]
Автоматизована перевірка	CI/CD-інтеграція	Nightly testssl.sh / sslyze	Окремий pipeline	Накопичення регресій	[11], [12]

Примітка. Цільові значення відповідають рекомендаціям NIST SP 800-52r2 [12] та Mozilla Server Side TLS (Intermediate profile) [11]. Директива ssl_ciphers керує шифрами лише для TLS 1.2; для конфігурації шифрів TLS 1.3 у Nginx 1.19.4+ використовується окрема директива ssl_conf_command Ciphersuites TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256;. За замовчуванням OpenSSL 3.0.2 застосовує безпечний набір усіх трьох стандартних шифрів TLS 1.3.

Рекомендовані конфігурації: для серверів із AES-NI – cipher suite TLS_AES_256_GCM_SHA384 з named group X25519 або secp256r1 (P-256) і ECDSA-сертифікатом P-256; для мобільних і IoT-систем – cipher suite TLS_CHACHA20_POLY1305_SHA256 з тими самими кривими та сертифікатом; для корпоративних середовищ – TLS 1.3 як пріоритет, TLS 1.2 як резерв сумісності, версії нижче TLS 1.2 відключити.

Перспективи квантово-стійкого TLS 1.3

У серпні 2024 року NIST опублікував стандарт FIPS 203 для алгоритму ML-KEM (CRYSTALS-Kyber) – першого стандартизованого механізму інкапсуляції ключів, стійкого до атак квантового комп'ютера [16]. Паралельно IETF розробляє проєкти draft-ietf-tls-mlkem [17] та draft-ietf-tls-hybrid-design [18], що вводять гібридні іменовані групи X25519MLKEM768 та SecP256r1MLKEM768 для TLS 1.3. Ключова загроза – harvest-now-decrypt-later: зловмисник записує зашифрований TLS-трафік сьогодні з наміром розшифрувати його після появи квантового комп'ютера.

Гібридний підхід комбінує класичний ECDHE-обмін з ML-KEM так, що спільний секрет сесії залишається безпечним, поки стійким є хоча б один з двох компонентів. Такий підхід обрали OpenJDK (JEP 527, 2025) [19], Cloudflare, Google Chrome та AWS. Згідно зі специфікацією IETF draft-ietf-tls-mlkem [17] та підтверджено у Zheng et al. (ESORICS 2024) [20], ML-KEM-768 додає близько 2272 байт до повідомлень ClientHello/ServerHello (1184 байти – encapsulation key у ClientHello, 1088 байт – ciphertext у ServerHello). Вимірювання Montenegro et al. [21] та Kampanakis і Childs-Klein [22] показують, що на стандартному x86-обладнанні гібридний handshake додає менше 1 мс латентності у локальних мережах. Інтеграція ML-DSA (FIPS 204) [27] для автентифікації сертифікатами разом з ML-KEM для обміну ключами забезпечує повне постквантове покриття, однак публічні центри сертифікації перебувають на етапі підготовки до масового розгортання постквантових CA.

Висновки та перспективи подальших досліджень

Комплексне порівняльне дослідження криптографічних протоколів TLS 1.2 та TLS 1.3 дозволило отримати обґрунтовані відповіді на питання ефективності сучасного захисту веб-додатків. За результатами мікроексперименту (Nginx 1.18, OpenSSL 3.0.2, wrk, Wireshark, три серії тривалістю 60 с із 50 паралельними з'єднаннями) зафіксовано стабільний приріст продуктивності TLS 1.3: час TLS-рукостискання скорочується на 37 % (з 4,8 до 3,0 мс), загальна затримка TTFB – на 37 % (з 48,3 до 30,1 мс), пропускна здатність зростає на 20 % (з 430 до 515 Мбіт/с). Зафіксовано, що перехід між конфігураціями супроводжується зміною CPU: 32 % → 28 % (індикативна різниця між TLS 1.2 з AES-NI та TLS 1.3 без AES-NI) і 28 % → 26 % (~7 %, ізольованіше порівняння шифронаборів в однакових програмних умовах). Через асиметричність базових конфігурацій повна ізоляція внеску архітектури TLS 1.3 та апаратного прискорення потребує додаткового тестування. Wireshark-аналіз підтвердив скорочення кількості пакетів рукостискання з 7–9 до 5–7, що безпосередньо корелює з виміряною економією латентності.

Аналіз літератури підтвердив, що отримані результати узгоджуються з висновками Holz et al. [2], Hasan et al. [3] та Restuccia et al. [4]. Відповідно до RFC 7925 [6] та даних Белей і Логутової [9], вибір оптимального шифронабору залежить від апаратної платформи: ChaCha20-Poly1305 є пріоритетним для мобільних і IoT-систем без апаратного

прискорення AES, тоді як AES-256-GCM залишається ефективнішим для серверів x86-64 із AES-NI, що підтверджує необхідність адаптивної, а не уніфікованої TLS-політики. Механізм 0-RTT PSK Session Resumption (~16 мс) є критичною перевагою для REST API, мікросервісів і мобільних клієнтів у нестабільних мережах, а TLS 1.3 забезпечує Perfect Forward Secrecy як обов'язкову архітектурну властивість, усуваючи цілий клас атак, можливих навіть у правильно налаштованому TLS 1.2. На основі аналізу стандартів NIST FIPS 203 [16] та IETF-драфтів [17, 18] встановлено, що overhead гібридного handshake з ML-KEM-768 складає близько 2 272 байт; за результатами вимірювань Montenegro et al. [21] та Kampanakis і Childs-Klein [22] додаткова латентність становить менше 1 мс на стандартному x86-обладнанні. Сукупно ці показники прийнятні для виробничих середовищ і свідчать про практичну готовність до постквантового переходу.

У роботі розроблено та запропоновано практичний чекліст із 13 параметрів безпечної конфігурації TLS на прикладі Nginx (табл. 2), цільові значення якого узгоджені з NIST SP 800-52r2 [12] та Mozilla Server Side TLS Intermediate profile [11]. Чекліст охоплює вибір версій протоколу, шифронаборів, керування сертифікатами, налаштування HSTS [24], HTTP/3 [25] та інтеграцію з CI/CD-пайплайнами, що робить його придатним для автоматизованої перевірки в DevSecOps-процесах. Сформульовано рекомендації щодо адаптивного вибору конфігурацій TLS залежно від апаратної платформи та типу розгортання: серверного, мобільного чи хмарного.

Перспективами подальших досліджень є оцінка енергоспоживання TLS-операцій у контейнеризованих середовищах (Docker, Kubernetes) з вимірюванням у ват-годинах на 1000 запитів; кросплатформне порівняння продуктивності TLS 1.3 на ARM-архітектурах з та без ARMv8 Cryptography Extensions; проведення власного benchmark постквантових гібридних конфігурацій через провайдер oqsprovider [23] з вимірюванням впливу на Time-to-Last-Byte; дослідження механізму Encrypted Client Hello (ECH) як засобу захисту метаданих SNI; розробка інтегрованої системи автоматизованої перевірки конфігурацій TLS у CI/CD-пайплайнах на основі запропонованого чекліста.

Література

1. Rescorla E. The Transport Layer Security (TLS) Protocol Version 1.3 : RFC 8446. RFC Editor, 2018. URL: https://doi.org/10.17487/rfc8446

2. Holz R., Hiller J., Amann J., Razaghpanah A., Jost T., Vallina-Rodriguez N., Hohlfeld O. Tracking the deployment of TLS 1.3 on the web. *ACM SIGCOMM Computer Communication Review*. 2020. Vol. 50, № 3. P. 3–15. URL: https://doi.org/10.1145/3411740.3411742

3. Hasan M. M., Rajkumar S., Ali S. A., Satyanarayana V., Jeyanthi S., Qodirova G. Analyzing SSL/TLS Handshake Latency in Modern Web Applications. *Journal of Internet Services and Information Security*. 2025. Vol. 15, № 4. P. 445–458. URL: https://doi.org/10.58346/jisis.2025.i4.032

4. Restuccia G., Tschofenig H., Baccelli E. Low-Power IoT Communication Security: On the Performance of DTLS and TLS 1.3. *2020 9th IFIP International Conference on Performance Evaluation and Modeling in Wireless Networks (PEMWN)*. IEEE, 2020. URL: https://doi.org/10.23919/pemwn50727.2020.9293085

5. Sullivan N. A Detailed Look at RFC 8446 (a.k.a. TLS 1.3). *The Cloudflare Blog*. 2018. URL: https://blog.cloudflare.com/rfc-8446-aka-tls-1-3

6. Fossati T., Tschofenig H. Transport Layer Security (TLS) / Datagram Transport Layer Security (DTLS) Profiles for the Internet of Things : RFC 7925. RFC Editor, 2016. URL: https://doi.org/10.17487/rfc7925

7. Aviram N., Schinzel S., Somorovsky J., Heninger N., Dankel M.,

Steube J. та ін. Drown: Breaking TLS using SSLv2. *Proceedings of the 25th USENIX Security Symposium*. 2016. P. 689–706.

8. Clark J., van Oorschot P. C. SoK: SSL and HTTPS: Revisiting Past Challenges and Evaluating Certificate Trust Model Enhancements. *2013 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2013. URL: https://doi.org/10.1109/sp.2013.41

9. Білей О., Логутова Т. Безпека передачі даних для інтернету речей. *Кібербезпека: освіта, наука, техніка*. 2019. № 2(6). С. 6–18. URL: https://doi.org/10.28925/2663-4023.2019.6.618

10. Шифрування HTTPS в Інтернеті. *Google Transparency Report*. URL: https://transparencyreport.google.com/https/overview

11. Security/Server Side TLS. *MozillaWiki*. 2026. URL: https://wiki.mozilla.org/Security/Server_Side_TLS

12. Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations : NIST SP 800-52 Rev. 2. National Institute of Standards and Technology, 2019. URL: https://doi.org/10.6028/NIST.SP.800-52r2

13. Oppliger R. *SSL and TLS: Theory and Practice*. 2nd ed. Artech House, 2016.

14. Dierks T., Rescorla E. The TLS Protocol Version 1.2 : RFC 5246. IETF, 2008. URL: https://doi.org/10.17487/rfc5246

15. Ericson J. TLS1.3. *GitHub*. 2025. URL: https://github.com/openssl/openssl/wiki/TLS1.3

16. Module-Lattice-Based Key-Encapsulation Mechanism Standard : FIPS 203. National Institute of Standards and Technology, 2024. URL: https://doi.org/10.6028/NIST.FIPS.203

17. Connolly D. ML-KEM Post-Quantum Key Agreement for TLS 1.3. *IETF Datatracker*. 2024. URL: https://datatracker.ietf.org/doc/draft-ietf-tls-mlkem/

18. Stebila D., Fluhrer S., Gueron S. Hybrid key exchange in TLS 1.3. *IETF Datatracker*. 2026. URL: https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/

19. Nimeh J. JEP 527: Post-Quantum Hybrid Key Exchange for TLS 1.3. *OpenJDK*. 2025. URL: https://openjdk.org/jeps/527

20. Zheng J., Zhu H., Dong Y., Song Z., Zhang Z., Yang Y., Zhao Y. Faster Post-quantum TLS 1.3 Based on ML-KEM: Implementation and Assessment. *Lecture Notes in Computer Science*. Springer Nature Switzerland, 2024. P. 123–143. URL: https://doi.org/10.1007/978-3-031-70890-9_7

21. Montenegro J. A., Rios R., López-Cerezo J. A Performance Evaluation Framework for Post-Quantum TLS. *Future Generation Computer Systems*. 2025. P. 108062. URL: https://doi.org/10.1016/j.future.2025.108062

22. Kampanakis P., Childs-Klein W. The impact of data-heavy, post-quantum TLS 1.3 on the Time-To-Last-Byte of Web connections. *Workshop on Measurements, Attacks, and Defenses for the Web*. Internet Society, 2024. URL: https://doi.org/10.14722/madweb.2024.23010

23. Open Quantum Safe provider for OpenSSL. *GitHub*. 2024. URL: https://github.com/open-quantum-safe/oqs-provider

24. Hodges J., Jackson C., Barth A. HTTP Strict Transport Security (HSTS) : RFC 6797. RFC Editor, 2012. URL: https://doi.org/10.17487/rfc6797

25. Bishop M. HTTP/3 : RFC 9114. RFC Editor, 2022. URL: https://doi.org/10.17487/rfc9114

26. OWASP Secure Headers Project. *OWASP Foundation*. 2024. URL: https://owasp.org/www-project-secure-headers/

27. Module-Lattice-Based Digital Signature Standard : FIPS 204. National Institute of Standards and Technology, 2024. URL: https://doi.org/10.6028/nist.fips.204

Дячук О. Ю., Колощук М.С., Рудюк Б. М., Кваснiков В. П.

ЕФЕКТИВНІСТЬ КРИПТОГРАФІЧНИХ ПРОТОКОЛІВ TLS 1.2 І TLS 1.3 У СУЧАСНИХ ВЕБ-ДОДАТКАХ: АНАЛІТИЧНИЙ ТА ПРАКТИЧНИЙ АСПЕКТИ

У роботі досліджено ефективність криптографічних протоколів TLS 1.2 та 1.3 у сучасних веб-додатках за двома вимірами: криптографічна стійкість та системна продуктивність. Виконано аналітичний огляд еволюції стека протоколів від SSL 3.0 до TLS 1.3, включаючи перехід від вразливих CBC-режимів до AEAD-алгоритмів (AES-256-GCM та ChaCha20-Poly1305), які з'явилися у TLS 1.2 і стали обов'язковими у TLS 1.3.

Методологія базується на порівнянні двох сценаріїв розгортання Nginx 1.18 з OpenSSL 3.0.2 на Ubuntu Server 22.04 LTS: типової конфігурації TLS 1.2 + AES-256-GCM з апаратним прискоренням AES-NI та конфігурації TLS 1.3 + ChaCha20-Poly1305, орієнтованої на мобільні та IoT-платформи без AES-NI. Вимірювання виконано інструментами wrk, Wireshark та openssl s_time.

Для TLS 1.3 зафіксовано скорочення часу рукостискання з 4,8 до 3,0 мс (−37 %), загальної затримки TTFB — з 48,3 до 30,1 мс, зростання пропускної здатності з 430 до 515 Мбіт/с (+20 %). Різниця у завантаженні CPU (−18 %) значною мірою зумовлена вибором шифронабору і платформи, що відображає типовий мобільно-серверний розрив. Додатково виміряно конфігурацію TLS 1.3 + AES-256-GCM без AES-NI як контрольну точку для порівняння шифронаборів.

Запропоновано рекомендації щодо оптимальних конфігурацій TLS 1.3 для серверних, мобільних та хмарних середовищ, а також 13-пунктовий чекліст безпечного розгортання для DevSecOps-інженерів. Окреслено перспективи переходу до квантово-стійких конфігурацій на основі NIST FIPS 203 (ML-KEM) та гібридних груп IETF (X25519MLKEM768, SecP256r1MLKEM768).

Ключові слова: *TLS 1.2; TLS 1.3; AES-GCM; ChaCha20-Poly1305; ECDHE; веб-безпека; HTTPS.*

Diachuk O. Yu., Koloshchuk M. S., Rudiuk B. M., Kvasnikov V. P.,

EFFICIENCY OF TLS 1.2 AND TLS 1.3 CRYPTOGRAPHIC PROTOCOLS IN MODERN WEB APPLICATIONS: ANALYTICAL AND PRACTICAL ASPECTS

This paper presents an empirical evaluation of TLS 1.2 and TLS 1.3 cryptographic protocols in modern web applications across two dimensions: cryptographic strength and system performance. An analytical review traces the protocol evolution from SSL 3.0 to TLS 1.3 (RFC 8446), documenting the transition from vulnerable CBC modes to AEAD algorithms (AES-256-GCM and ChaCha20-Poly1305) and the 1-RTT handshake with mandatory ECDHE key exchange.

The study compares two practical deployment scenarios of Nginx 1.18 with OpenSSL 3.0.2 on Ubuntu Server 22.04 LTS: a typical server-side configuration (TLS 1.2 + AES-256-GCM with AES-NI acceleration) and a mobile/IoT-oriented configuration (TLS 1.3 + ChaCha20-Poly1305 without AES-NI, emulated via OPENSSL_ia32cap). An additional control configuration (TLS 1.3 + AES-256-GCM without AES-NI) was included to compare cipher-suite behavior within TLS 1.3. Measurements were performed using open-source tools — wrk, Wireshark and openssl s_time.

The TLS 1.3 scenario demonstrated handshake time reduction from 4.8 to 3.0 ms (−37 %), TTFB reduction from 48.3 to 30.1 ms, and throughput increase from 430 to 515 Mbit/s (+20 %). The observed −18 % CPU utilization difference reflects the combined effect of protocol optimization, cipher choice and software-only execution — representative of the asymmetry between server and client-side TLS deployments. A practical 13-item secure TLS deployment checklist for DevSecOps engineers is proposed, aligned with NIST SP 800-52r2 and the Mozilla Server Side TLS Intermediate profile. The paper outlines the quantum-resistant transition path via NIST FIPS 203 (ML-KEM) standardization and IETF hybrid key exchange drafts (X25519MLKEM768, SecP256r1MLKEM768).

Keywords: *TLS 1.2; TLS 1.3; AES-GCM; ChaCha20-Poly1305; ECDHE; web security; HTTPS.*

Стаття подана до редакції: 12/05/2026
Стаття прийнята до опублікування: 15/05/2026
Стаття опублікована: 30/05/2026