Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя
(повне найменування вищого навчального закладу)
Факультет комп'ютерно-інформаційних систем і програмної інженерії
(назва факультету)
Кафедра кібербезпеки
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

магістр
(освітній рівень)

на тему: " Вплив шифрування на продуктивність вебсайтів: порівняльний аналіз сучасних алгоритмів"

Виконав: студент (ка)
Спеціальності:
125 «Кібербезпека та захист інформації»
(шифр і назва напряму підготовки, спеціальності)
Гурський В. Б.
підпис (прізвище та ініціали)

Керівник Кульчицький Т.Р.
підпис (прізвище та ініціали)

Нормоконтроль Тимощук Д. І.
підпис (прізвище та ініціали)

Завідувач кафедри Загородна Н.В.
підпис (прізвище та ініціали)

Рецензент Лещишин Ю. З.
підпис (прізвище та ініціали)

м. Тернопіль – 2024

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії

Кафедра Кібербезпеки

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Загородна Н.В.
(підпис) (прізвище та ініціали)
« » 20___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Магістр
(назва освітнього ступеня)
за спеціальністю 125 Кібербезпека та захист інформації
студенту Гурському Віталію Богдановичу
(прізвище, ім'я, по батькові)
1. Тема роботи Вплив шифрування на продуктивність вебсайтів: порівняльний аналіз сучасних алгоритмів

Керівник роботи Кульчицький Тарас Русланович Ph.D в галузі права
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
Затверджені наказом по університету від «20» листопада 2024 року № 4/7-1112
2. Термін подання студентом роботи 16.12.2024
3. Вихідні дані до роботи Вихідні дані до роботи включають аналіз впливу сучасних, алгоритмів шифрування на продуктивність вебсайтів порівняння їх ефективності та визначення. оптимальних рішень для забезпечення балансу між безпекою та швидкодією.
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
ВСТУП Розділ 1. Теоретичні основи шифрування даних у вебсередовищі 1.1 Типи алгоритмів шифрування, протоколи захисту з'єднань у вебсередовищі та їх особливості 1.2 Методи оцінки ефективності шифрування та їх вплив на продуктивність вебсайтів 1.3 Сучасні тенденції в області шифрування для вебсайтів та їх перспективи розвитку РОЗДІЛ 2. ОЦІНКА ЕФЕКТИВНОСТІ ШИФРУВАННЯ ДЛЯ ВЕБСАЙТІВ 2.1. Ключові показники ефективності шифрування та методи тестування продуктивності для вебсайтів 2.2. Аналіз впливу шифрування на час відгуку і пропускну здатність вебсайтів 2.3. Порівняння алгоритмів шифрування за ефективністю та рекомендації щодо оптимізації шифрування РОЗДІЛ 3. ПРАКТИЧНА РЕАЛІЗАЦІЯ ТА АНАЛІЗ ВПЛИВУ ШИФРУВАННЯ НА ПРОДУКТИВНІСТЬ ВЕБСАЙТІВ 3.1 Опис експериментального середовища та реалізація шифрування 3.2 Методика вимірювання продуктивності 3.3 Проведення експериментів і аналіз результатів 4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ 4.1 Охорона праці 4.2. Безпека в надзвичайних ситуаціях: Основні принципи і способи забезпечення життєдіяльності ВИСНОВКИ СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ Додаток А Публікація

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1.Титульний слайд 2.Вступ 3.Типи алгоритмів шифрування 4.Вплив шифрування на для продуктивність вебсайтів 5.Порівняння продуктивності алгоритмів 6.Інструменти тестування шифрування 7.Налаштування тестового середовища 8.Получення результатів 9.Аналіз получених результатів 10.Висновки 11.Дякую за увагу

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	Осухівська Г.М. к.т.н. зав. Кафедри КС		
Безпека в надзвичайних ситуаціях	Клепчик В.М, проректор з адміністративно-господарської роботи та будівництва		

7. Дата видачі завдання 23.09. 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Ознайомлення з завданням до кваліфікаційної роботи	23.09.2024	Виконано
2	Вибір та аналіз джерел, що стосуються теми кваліфікаційної роботи	24.09 – 01.10	Виконано
3	Дослідження джерел по роботі зі шифруванням	02.10 – 09.10	Виконано
4	Формування полученої інформації для першого розділу	10.10 – 17.10	Виконано
5	Оцінка ефективності шифррування для вебсайтів	24.10 – 31.10	Виконано
6	Аналіз впливу шифрування на роботу вебсатів	01.11 – 06.11	Виконано
7	Практична реалізація тестів по роботі алггоритмів шифрування	07.11 - 14.11	Виконано
8	Аналіз получених результатів і їх упорядкування	15.11 – 20.11	Виконано
9	Оформлення першого розділу	21.11 – 24.11	Виконано
10	Оформлення другого розділу	25.11 – 29.11	Виконано
11	Оформлення третього розділу	30.11 – 03.12	Виконано
12	Оформлення розділу «Безпека життєдіяльності і основи охорони праці»	04.12 – 08.12	Виконано
13	Оформлення кваліфікаційної роботи	09.12 – 12.12	Виконано
14	Нормконтроль		Виконано
15	Перевірка на плагіат		Виконано
16	Захист кваліфікаційної роботи	26.12	

Студент _____ Гурський В.Б.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Кульчицький Т.Р.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Робота спрямована на забезпечення балансу між високим рівнем безпеки користувацьких даних та продуктивністю вебресурсів, що є важливим завданням у сучасних умовах розвитку кіберзагроз. // ОР «Магістр» // Гурський Віталій Богданович// Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра кібербезпеки, група СБм-61 // Тернопіль, 2024 // С. 69 , табл. – 1 , додат. – 1 .

Ключові слова: криптографія, вебсайти, алгоритми, шифрування, кібербезпека, SSL, TLS, сертифікати, продуктивність.

У магістерській кваліфікаційній роботі досліджено вплив алгоритмів шифрування на продуктивність вебсайтів. Основна увага приділяється порівняльному аналізу сучасних алгоритмів шифрування, які використовуються для забезпечення захищеності даних у вебсередовищі, зокрема протоколів TLS /SSL. Проведено оцінку продуктивності різних алгоритмів, таких як AES, ChaCha20, RSA та ECC, за показниками часу завантаження сторінок, використання серверних ресурсів та затримки під час передачі даних.

Практична частина включає розробку тестового середовища для впровадження та вимірювання ефективності алгоритмів у реальних умовах. Використано інструменти моніторингу для збору даних та проведено експериментальне порівняння обраних алгоритмів. На основі отриманих результатів сформовано рекомендації щодо вибору оптимального алгоритму шифрування залежно від специфіки вебсайту та його навантаження.

ABSTRACT

The work is aimed at ensuring a balance between a high level of user data security and web resource productivity, which is an important task in modern conditions of cyberthreat development. // OR "Master" // Gursky Vitaliy Bogdanovych // Ivan Pulyuy Ternopil National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Cybersecurity, Group SBm-61 // Ternopil, 2024 // P. 69 , table. – 1 , appendix. – 1 .

Keywords: cryptography, websites, algorithms, encryption, cybersecurity, SSL, TLS, certificates, productivity.

The master's qualification work investigates the impact of encryption algorithms on website performance. The main focus is on a comparative analysis of modern encryption algorithms used to ensure data security in the web environment, in particular TLS /SSL protocols. The performance of various algorithms, such as AES, ChaCha20, RSA, and ECC, is assessed in terms of page load time, server resource usage, and data transfer latency.

The practical part includes the development of a test environment for implementing and measuring the effectiveness of algorithms in real conditions. Monitoring tools were used to collect data and an experimental comparison of the selected algorithms was conducted. Based on the results obtained, recommendations were made for choosing the optimal encryption algorithm depending on the specifics of the website and its load.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

TLS - Transport Layer Security

SSL - Secure Sockets Layer

AES - Advanced Encryption Standard

RSA - Rivest-Shamir-Adleman

ECC - Elliptic Curve Cryptography

HTTPS - Hypertext Transfer Protocol Secure

HSTS-HTTP Strict Transport Security

ВСТУП

Сучасні вебсайти стали невід'ємною частиною інфраструктури цифрового світу, забезпечуючи доступ до інформації, послуг та товарів у різних сферах діяльності. Вебресурси використовуються як у бізнесі, так і в освіті, медицині, урядових установах, надаючи можливість обміну даними та автоматизації процесів. Водночас швидкий розвиток технологій і збільшення кількості онлайн-сервісів супроводжується зростанням кіберзагроз, серед яких хакерські атаки, витоки особистих даних, фішингові атаки, а також загрози для конфіденційності і цілісності переданої інформації.

У зв'язку з цим, шифрування даних на вебсайтах стало необхідним елементом для забезпечення захисту чутливої інформації користувачів, а також для забезпечення конфіденційності комунікацій. Протокол HTTPS, який використовує шифрування на основі SSL/TLS сертифікатів, став стандартом для безпечної передачі даних через Інтернет. Однак застосування складних алгоритмів шифрування, таких як AES-256 або RSA, може мати вплив на продуктивність вебсайтів, особливо в умовах високих навантажень або обмежених ресурсів серверів. Цей вплив проявляється у вигляді збільшення часу відгуку сайту, що, в свою чергу, може знижувати користувацький досвід і ефективність роботи сервісів.

Таким чином, питання оптимізації шифрування, яке дозволяє забезпечити достатній рівень безпеки без значного впливу на продуктивність вебсайтів, є особливо актуальним. Зокрема, важливо знайти оптимальний баланс між високим рівнем захисту даних і збереженням швидкодії вебсайтів, щоб уникнути втрат у продуктивності при забезпеченні належного рівня безпеки для користувачів. Пошук таких рішень дозволить зменшити навантаження на сервери та підвищити ефективність вебресурсів, що вкрай важливо для їх функціонування в умовах сучасної цифрової економіки.

Для вирішення цієї проблеми необхідні дослідження в напрямку оптимізації алгоритмів шифрування, а також застосування нових технологій, які дозволяють мінімізувати вплив на продуктивність без шкоди для безпеки. Ці питання є

особливо важливими для підприємств, що прагнуть забезпечити високий рівень безпеки своїх онлайн-сервісів, не жертвуючи якістю та швидкістю роботи їхніх платформ.

Метою роботи є аналіз впливу алгоритмів шифрування на продуктивність вебсайтів і визначення оптимальних рішень для забезпечення безпеки даних без суттєвого зниження швидкодії.

Завданнями дослідження є:

– Вивчення теоретичних основ шифрування даних у вебсередовищі.

– Оцінка впливу обраних алгоритмів на продуктивність вебсайтів, включаючи час завантаження сторінок, затримки та навантаження на сервер.

– Налаштування тестового середовища для проведення експерементів та порівняльного аналізу алгоритмів.

Об'єктом дослідження виступають вебсайти, які використовують шифрування для захисту переданих даних. Це включає всі веб-ресурси, що застосовують протоколи захисту, такі як HTTPS (SSL/TLS), для шифрування даних, що передаються між клієнтом і сервером, з метою забезпечення конфіденційності, цілісності і аутентифікації.

Предметом дослідження є вплив алгоритмів шифрування на продуктивність вебсайтів. Це дослідження зосереджується на вивченні того, як застосування різних алгоритмів шифрування (наприклад, AES, RSA, ECC) впливає на час завантаження вебсайтів, ефективність обробки запитів і загальну продуктивність системи. Це включає вивчення часу, необхідного для встановлення захищеного з'єднання, обробки запитів користувачів і передачі даних через захищений канал.

У роботі представлено порівняльний аналіз сучасних алгоритмів шифрування з огляду на їхній вплив на продуктивність вебсайтів. Запропоновано систематизований підхід до вибору алгоритмів шифрування, що залежить від типу вебсайту та специфічних вимог до його продуктивності. Це дозволяє точніше визначати оптимальні рішення для забезпечення ефективної роботи сайтів з високими вимогами до безпеки та швидкості завантаження.

Результати дослідження можуть бути використані для оптимізації налаштувань захищеного з'єднання на вебсайтах. Це забезпечить ефективний

баланс між безпекою і швидкодією, дозволяючи адмініструвати веб-ресурси з урахуванням вимог до продуктивності та рівня захисту. Рекомендації з вибору алгоритмів шифрування є корисними для розробників веб-додатків, адміністраторів серверів і фахівців з кібербезпеки, адже допомагають у прийнятті обґрунтованих рішень щодо налаштувань шифрування для конкретних умов експлуатації.

Апробація результатів магістерської роботи. Основні результати проведених досліджень обговорювались на: XII науково-технічна конференція «Інформаційні моделі, системи та технології» (м.Тернопіль, Україна).

Публікації. Основні результати кваліфікаційної роботи опубліковано у працях конференції (див. Додаток А).

РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ ШИФРУВАННЯ ДАНИХ У ВЕБСЕРЕДОВИЩІ

1.1 Типи алгоритмів шифрування, протоколи захисту з’єднань у вебсередовищі та їх особливості

Шифрування є одним із найважливіших інструментів для захисту даних, які передаються через інтернет. Особливо це стосується вебсайтів, які працюють із конфіденційною інформацією: логіни, паролі, фінансові транзакції чи особисті дані. Без використання шифрування такі дані можуть стати легкою мішенню для кіберзлочинців.[1]

Головна мета шифрування — зробити інформацію недоступною для сторонніх осіб у разі її перехоплення. Уявіть, що дані "зашифровані" у вигляді секретного коду, який можуть розшифрувати тільки відправник і отримувач. Без такого захисту виникає ряд загроз, наприклад:

Перехоплення даних: Зловмисники можуть підключитися до каналу зв’язку й читати або змінювати інформацію.

Несанкціонований доступ: Дані можуть потрапити в руки сторонніх осіб, які не мають на це права.

Втрата довіри: Якщо зловмисники зламають з’єднання, користувачі перестануть довіряти вебресурсу.[4]

Сьогодні кіберзагрози, такі як фішинг чи атаки "людина посередині" (MAN-IN-THE-MIDDLE), постійно вдосконалюються. Шифрування допомагає вирішити ці проблеми, виконуючи такі функції:

Захист конфіденційності: Дані стають невидимими для сторонніх осіб.

Перевірка цілісності: Можливість переконатися, що інформація не була змінена.

Підтвердження особи: Шифрування дозволяє перевірити, чи дійсно ви спілкуєтеся з тим, із ким планували.

Загалом, шифрування — це базова складова безпеки сучасних вебсайтів. Воно не тільки захищає дані, а й створює довіру між користувачем і сайтом.

Алгоритми шифрування є основою для захисту даних, що передаються через вебсайти. Вони поділяються на два основні види: симетричні та асиметричні.

Симетричне шифрування базується на використанні одного ключа для шифрування і розшифрування даних. Такий підхід забезпечує високу швидкість обробки інформації, що робить його зручним для роботи з великими обсягами даних. Найпоширенішими симетричними алгоритмами є:

Advanced Encryption Standard: цей алгоритм є надійним і продуктивним, тому його часто використовують для захисту даних у вебсередовищі.

ChaCha20: сучасний алгоритм, що демонструє високу ефективність на пристроях із обмеженими ресурсами, таких як мобільні телефони.

Однак головною проблемою симетричного шифрування є забезпечення безпечного обміну ключами між сторонами, що взаємодіють.

Асиметричне шифрування працює за принципом використання двох ключів: відкритого (для шифрування) і закритого (для розшифрування). Це дозволяє уникнути необхідності передачі спільного ключа між сторонами. Серед популярних асиметричних алгоритмів можна виділити:

Rivest-Shamir-Adleman: один із найстаріших і найпоширеніших методів, який гарантує високий рівень безпеки, хоча поступається сучасним алгоритмам у швидкості.

Elliptic Curve Cryptography: цей алгоритм забезпечує такий самий рівень захисту, як і RSA, але потребує менших обчислювальних ресурсів завдяки коротшим ключам.

Зазвичай асиметричні алгоритми використовують для обміну ключами, а для подальшого шифрування даних застосовують симетричні методи, що дає змогу досягти високої продуктивності.

Симетричні алгоритми найкраще підходять для шифрування великих обсягів інформації, оскільки вони швидкі й ефективні. Асиметричні алгоритми більше використовують для завдань, пов’язаних із безпечним обміном ключами та автентифікацією.

Таким чином, сучасні протоколи безпеки (наприклад, SSL/TLS) інтегрують обидва типи алгоритмів, що дозволяє одночасно забезпечити високий рівень захисту та продуктивність.

Протокол TLS забезпечує автентифікацію та безпечну передачу даних через Інтернет завдяки криптографічним механізмам. У типовому випадку автентифікація проводиться лише для сервера, тоді як клієнт залишається неавтентифікованим. Для забезпечення взаємної автентифікації обидві сторони повинні використовувати інфраструктуру відкритих ключів (PKI), яка захищає клієнт-серверні додатки від перехоплення, модифікації або підроблення повідомлень.

Процес TLS включає такі етапи:

– Вибір алгоритму шифрування під час початкового діалогу між сторонами.

– Обмін ключами на основі криптографії з відкритим ключем або автентифікація через сертифікати.

– Передача даних, зашифрованих симетричними алгоритмами.

– Відкриття сеансу.

Процедура рукостискання дозволяє клієнту та серверу узгодити та обмінятися таємним ключем для подальшого шифрування даних. Основні дії під час рукостискання:

Узгодження версії протоколу.

Вибір криптографічних алгоритмів.

Взаємна автентифікація через цифрові підписи.

Створення спільного таємного ключа за допомогою асиметричних алгоритмів, який потім використовується для симетричного шифрування даних через його високу продуктивність.

Процес рукостискання складається з таких кроків:

– Клієнт надсилає "Hello" з версією протоколу, списком підтримуваних шифрів і випадковими байтами.

– Сервер відповідає "Hello", вибирає шифр зі списку клієнта, додає випадкові байти, ідентифікатор сеансу та цифровий сертифікат. Якщо потрібен сертифікат клієнта, сервер запитує його.

– Клієнт перевіряє сертифікат сервера та надсилає випадкові байти, зашифровані відкритим ключем сервера.

– Якщо сервер запитує сертифікат клієнта, клієнт надсилає його разом із підписаними випадковими байтами.

– Сервер перевіряє сертифікат клієнта, після чого обидві сторони надсилають повідомлення про успішне завершення рукостискання.

Алгоритми узгодження ключів під час рукостискання можуть включати RSA, протокол Діффі-Геллмана (включаючи варіанти на еліптичних кривих), короткочасні ключі та Secure Remote Password. Довжина ключів сертифікатів може впливати на їх стійкість: наприклад, Google з 2013 року використовує ключі довжиною 2048 біт замість 1024 біт.

Протокол SSL, попередник TLS, складається з двох підпротоколів: протоколу запису та рукостискання. Він забезпечує автентифікацію сервера (а іноді і клієнта), цілісність переданих даних і конфіденційність через шифрування.

Формат запису даних у SSL містить заголовок і дані. Довжина запису визначається залежно від використання заповнювачів, необхідних для роботи блокового шифру. При шифруванні дані включають:

Код автентифікації повідомлення (MAC).

Дані заповнювача.

Основні дані.

Шифрування використовується після встановлення з’єднання, забезпечуючи конфіденційність і захист переданих даних.

У сучасному вебсередовищі протоколи захисту з'єднань, такі як SSL і TLS , є основними інструментами для забезпечення безпеки даних, що передаються між вебсайтами та їх користувачами. Вони створюють зашифроване з'єднання, що дозволяє захистити інформацію від перехоплення та змін.

Протоколи SSL/TLS забезпечують такі функції:

Шифрування: Використання криптографічних алгоритмів для забезпечення конфіденційності даних під час їх передачі.

Аутентифікація: Перевірка автентичності серверів, щоб користувачі могли бути впевненими, що вони спілкуються з правильним сайтом.

Цілісність: Перевірка того, що передані дані не були змінені або пошкоджені під час передачі.

Протокол SSL був розроблений ще в 1994 році, але з часом з'явилися його вдосконалені версії. Найсучаснішим є протокол TLS , який є наступником SSL і надає більш високий рівень захисту та ефективності. TLS використовує більш надійні алгоритми шифрування та забезпечує більшу стійкість до атак.

Основні етапи встановлення захищеного з'єднання за допомогою SSL/TLS включають:

Обмін сертифікатами: Сервер надає свій публічний сертифікат для перевірки автентичності.

Аутентифікація і обмін ключами: Клієнт і сервер погоджуються на використання спільного ключа для шифрування з'єднання.

Шифрування сеансу: Всі дані, що передаються між клієнтом і сервером, шифруються за допомогою цього спільного ключа.

Протоколи SSL/TLS активно використовуються на всіх сучасних вебсайтах, особливо там, де обробляються конфіденційні дані користувачів, такі як електронна комерція, банківські операції та онлайн-платформи для передачі особистої інформації.

У той час як SSL вже застарілий і більше не рекомендується для використання через уразливості, версія TLS 1.2 і новіші варіанти є стандартом для забезпечення безпечних з'єднань у сучасному Інтернеті.

1.2 Методи оцінки ефективності шифрування та їх вплив на продуктивність вебсайтів

Шифрування є важливою частиною забезпечення безпеки вебсайтів, але воно також може мати вплив на продуктивність. Оскільки шифрування вимагає

обробки даних за допомогою складних алгоритмів, це може значно збільшити час відповіді серверів і час завантаження вебсторінок, особливо для великих обсягів даних.[5]

Основні фактори, які впливають на продуктивність через шифрування:

Витрати на обчислення: Шифрування потребує значних обчислювальних ресурсів для шифрування та розшифрування даних. Чим складніші алгоритми, тим більші ресурси вони споживають.

Затримка на етапі встановлення з'єднання: Протоколи, такі як TLS , передбачають певні етапи, які займають час (наприклад, обмін сертифікатами та ключами), що може збільшити час встановлення з'єднання між клієнтом і сервером.

Великий обсяг трафіку: Вебсайти з високим трафіком або великою кількістю одночасних користувачів можуть зазнати затримок, якщо шифрування здійснюється без оптимізації.

Однак, незважаючи на ці виклики, з кожною новою версією алгоритмів шифрування вдосконалюються методи, які дозволяють знизити ці витрати. Наприклад:

- Використання асиметричних алгоритмів для обміну ключами та симетричних алгоритмів для передачі даних значно підвищує ефективність.
- Алгоритми нового покоління, як-от спеціалізовані інструкції для апаратного прискорення шифрування, дозволяють знижувати навантаження на сервери, одночасно забезпечуючи високий рівень безпеки.
- Компресія даних перед їх шифруванням також може допомогти зменшити час, необхідний для передачі великих обсягів інформації.

Підвищення продуктивності під час шифрування є важливим завданням для адміністраторів вебсайтів, оскільки користувачі очікують швидкого доступу до ресурсів інтернету, навіть при високому рівні безпеки. Правильний вибір алгоритмів і налаштувань шифрування може допомогти досягти балансу між безпекою і швидкістю роботи вебсайтів.

Шифрування даних є важливою складовою сучасної безпеки вебсайтів, однак впровадження криптографічних протоколів супроводжується низкою

проблем та викликів, що можуть впливати на ефективність роботи вебсайтів та загальний рівень безпеки. Серед основних проблем можна виділити[6]:

Шифрування вимагає значних обчислювальних ресурсів, що може негативно впливати на продуктивність вебсайтів, особливо при високих навантаженнях. Використання складних криптографічних алгоритмів може збільшити час відповіді сервера, що може стати проблемою для сайтів із високим трафіком. Це вимагає оптимізації процесу шифрування і застосування сучасних технологій, таких як апаратні прискорювачі чи спрощення обміну ключами.

Для забезпечення належного рівня безпеки важливо правильно керувати криптографічними ключами. Невірно збережені або передані ключі можуть стати слабким місцем у системі безпеки. Проблема ускладнюється, коли вимагається використовувати кілька різних ключів для різних частин системи, що може призвести до помилок у конфігурації.

Не всі старі системи та вебсайти можуть підтримувати новітні криптографічні стандарти. Перехід до сучасних версій SSL/TLS або впровадження нових алгоритмів може викликати проблеми з сумісністю з усталеними інфраструктурами або з іншими системами, що використовуються для обробки даних. Крім того, зміни у протоколах можуть вимагати адаптації клієнтських додатків та браузерів.[7]

SSL/TLS сертифікати є основою для підтвердження автентичності серверів. Однак, якщо сертифікаційний центр (CA) зазнає атаки або зловмисники отримують сертифікати, вони можуть здійснити атаку на довіру користувачів. Система сертифікації потребує постійної перевірки на надійність, оскільки компрометація сертифікатів може призвести до серйозних проблем з безпекою.

З розвитком квантових обчислень постає проблема, що сучасні криптографічні алгоритми можуть бути вразливими до квантових атак. Це стосується, зокрема, алгоритмів, які використовують великі прості числа для шифрування (наприклад, RSA). У майбутньому необхідно буде адаптувати існуючі методи до квантових технологій або розробляти нові алгоритми, стійкі до таких атак.[8]

Шифрування може бути складним для розуміння і налаштування, що створює бар'єри для малих та середніх бізнесів. Належна кваліфікація та знання адміністраторів вебсайтів є критичними для ефективного впровадження та підтримки криптографічних систем. Брак спеціалістів у галузі кібербезпеки може призвести до помилок, що ставлять під загрозу безпеку вебресурсів.

Шифрування потребує регулярних оновлень та підтримки інфраструктури. Це включає оновлення сертифікатів, адаптацію до нових стандартів безпеки та постійний моніторинг стану безпеки. Ці витрати можуть стати значними для компаній, які не мають достатніх ресурсів для підтримки сучасних технологій.[9]

Оцінка ефективності шифрування для вебсайтів є важливою для забезпечення оптимального балансу між безпекою та продуктивністю. Це включає в себе не лише вимірювання швидкості роботи вебсайту після впровадження шифрування, але й оцінку рівня захисту, що надається користувачам. Існує кілька основних методів, які дозволяють оцінити ефективність шифрування на вебсайтах:

Один з найпоширеніших методів — це вимірювання часу відгуку сервера до клієнта до і після впровадження шифрування. Інструменти, такі як Apache JMeter, Google Lighthouse або WebPageTest, дозволяють виміряти час завантаження вебсторінок, час встановлення зашифрованого з'єднання та інші параметри. Це дає змогу виявити потенційні проблеми з продуктивністю, які можуть бути викликані обробкою шифрування.[3]

Для глибшої оцінки впливу шифрування на сервери використовуються методи профілювання. Профілювання дозволяє моніторити навантаження на процесори та інші ресурси під час шифрування та розшифрування даних. Це дає чітке уявлення про те, скільки ресурсів витрачається на криптографічні операції і чи є потреба у додатковій оптимізації серверів.[10]

Оцінка безпеки шифрування за допомогою вразливостей та атак.

Оцінка безпеки є критично важливою частиною процесу. Для цього використовуються різноманітні методи, включаючи пенетраційне тестування та аналіз вразливостей, які можуть виникнути через неправильне впровадження шифрування або використання застарілих протоколів. Інструменти типу

OpenVAS, Nmap, або Qualys дозволяють сканувати сервіси на наявність уразливостей у протоколах SSL/TLS , виявляти можливі точки доступу для зловмисників.[2]

Важливо також тестувати, як вебсайт витримує високі навантаження, коли всі з'єднання захищені шифруванням. Це тестування дозволяє визначити, чи здатна інфраструктура вебсайту працювати ефективно при збільшенні кількості одночасних підключень і шифрування даних на великих обсягах трафіку.

Методами оцінки є також тестування конкретних криптографічних алгоритмів за їх ефективністю. Наприклад, для симетричних алгоритмів, таких як AES, важливо порівнювати швидкість виконання шифрування в залежності від довжини ключа (128, 192, 256 біт), а для асиметричних алгоритмів (RSA, ECC) — ефективність генерації та обміну ключами.

З урахуванням зростаючої кількості мобільних користувачів важливо враховувати, як шифрування впливає на продуктивність на мобільних пристроях з обмеженими ресурсами. Оцінка ефективності шифрування на різних платформах дозволяє визначити, чи є потреба в додаткових налаштуваннях або оптимізації для мобільних версій сайтів.

Іншим важливим аспектом є оцінка впливу шифрування на зручність користувачів. Для цього проводяться тестування зручності роботи на вебсайті до та після впровадження шифрування, включаючи швидкість доступу, завантаження сторінок та взаємодію з формами або іншими інтерактивними елементами.

1.3 Сучасні тенденції в області шифрування для вебсайтів та їх перспективи розвитку

З розвитком технологій і збільшенням обсягів онлайн-даних, шифрування стало ще важливішим для безпеки вебсайтів. Останні тенденції в області криптографії допомагають забезпечити більш високий рівень захисту при мінімальних витратах на продуктивність. Розглянемо деякі з найбільш важливих напрямків у цій галузі:

Протокол TLS 1.3 є останньою версією стандарту для захищених з'єднань в Інтернеті. Однією з основних переваг TLS 1.3 є значне покращення продуктивності, оскільки цей протокол усуває деякі старі функції, які більше не використовуються, і спрощує процес обміну ключами. В результаті час на встановлення з'єднання значно зменшився, а також зросла безпека завдяки використанню сучасних криптографічних алгоритмів.

З розвитком квантових обчислень виникає потреба у нових методах захисту даних. Квантова криптографія обіцяє революцію у світі безпеки, оскільки вона ґрунтується на принципах квантової механіки. Одним із таких методів є квантове розподілення ключів (QKD), яке забезпечує теоретично абсолютний рівень безпеки, оскільки будь-яка спроба перехоплення квантових ключів одразу ж порушує їхню цілісність.[11]

Покращення шифрування з використанням апаратних прискорювачів Використання апаратних прискорювачів для шифрування даних дозволяє значно знизити навантаження на центральний процесор і підвищити ефективність. Сучасні процесори, такі як Intel AES-NI, підтримують спеціалізовані інструкції для швидкого виконання криптографічних операцій. Це дозволяє вебсайтам обробляти великі обсяги зашифрованих даних з мінімальними витратами на ресурси.

Об'єднання шифрування з іншими технологіями захисту. Вебсайти все частіше використовують поєднання шифрування з іншими засобами захисту, такими як автентифікація з багатофакторною перевіркою (MFA) або блокчейн-технології для забезпечення надійної перевірки транзакцій. Це дозволяє

створити більш комплексний підхід до безпеки та знижує ймовірність успішних атак.

Ініціативи щодо вдосконалення SSL/TLS сертифікатів SSL/TLS сертифікати також продовжують вдосконалюватися.

Ініціативи, такі як Let's Encrypt, забезпечують доступність безкоштовних сертифікатів для вебсайтів, що сприяє збільшенню використання HTTPS на всіх сайтах, включаючи ті, що раніше не мали можливості забезпечити захист даних.

Загалом, останні тенденції в області шифрування спрямовані на поліпшення продуктивності без шкоди для безпеки, а також на впровадження нових технологій, таких як квантова криптографія та апаратні прискорювачі. Це дозволяє вебсайтам і користувачам Інтернету відчувати більшу впевненість у захищеності їхніх даних.

Сучасні технології шифрування на вебсайтах продовжують розвиватися, а нові технологічні досягнення дають змогу створювати ще більш ефективні та безпечні рішення.

В майбутньому можна очікувати низку інновацій у галузі криптографії, які сприятимуть покращенню захисту веб-ресурсів. Основні напрямки розвитку шифрування для вебсайтів включають:

Інтеграція квантових технологій у шифрування. Як вже зазначалося, квантові обчислення можуть значно змінити ландшафт криптографії. Квантові комп'ютери потенційно здатні зламувати багато традиційних криптографічних алгоритмів, тому розробка квантово-стійких алгоритмів є важливою задачею для майбутнього шифрування даних.

Перехід до квантової криптографії дозволить створити шифри, які будуть надійно захищати інформацію навіть перед обличчям квантових атак.

Це може змінити стандартні підходи до забезпечення безпеки вебсайтів у майбутньому.

Використання багатофакторної аутентифікації, разом з сучасними алгоритмами шифрування, дозволить створити більш захищені вебсайти.

Багатофакторна криптографія передбачає використання кількох рівнів захисту, таких як паролі, біометричні дані, токени або динамічні ключі доступу.

Це підвищить надійність захисту від атак, таких як фішинг, або несанкціоноване проникнення.

Розвиток і впровадження нових криптографічних алгоритмів Технології, що стоять за криптографією, також продовжують еволюціонувати. Очікується розвиток нових криптографічних стандартів, які надаватимуть високу безпеку при мінімальних витратах на обчислювальні ресурси.

Наприклад, криві Еліптичної кривої (ECC) набирають популярності, оскільки вони пропонують високу стійкість до атак при використанні значно менших ключів, що робить їх більш ефективними з точки зору продуктивності. Шифрування на основі блокчейн-технологій.

Використання блокчейн-технологій для шифрування веб-транзакцій і взаємодій набирає популярності. Блокчейн пропонує децентралізовану систему зберігання даних, що підвищує безпеку та прозорість при обміні інформацією. Ця технологія може стати основою для захисту не тільки фінансових транзакцій, але й особистих даних на вебсайтах.

Поліпшення захисту даних на мобільних пристроях. Зростаюча кількість мобільних користувачів вимагає посиленої уваги до шифрування на мобільних пристроях. У майбутньому буде розвиватися мобільне шифрування, яке дозволить ефективно захищати дані не тільки в стаціонарних комп'ютерах, але й у мобільних додатках та на вебсайтах, що працюють на смартфонах і планшетах.

Це особливо важливо в умовах постійного зростання кількості мобільних атак. Автоматизація процесу впровадження та управління шифруванням З розвитком технологій автоматизації впровадження шифрування, організації зможуть значно спростити процес інтеграції криптографії на своїх вебсайтах.

Вже сьогодні існують сервіси, які автоматично генерують і оновлюють SSL/TLS сертифікати (наприклад, Let's Encrypt). У майбутньому ця технологія буде автоматизувати більшість аспектів криптографічного захисту, що дозволить знизити витрати на адміністрування без шкоди для безпеки.

Зростання важливості персоналізованого шифрування для користувачів У майбутньому, ймовірно, розвиватиметься концепція персоналізованого

шифрування, коли кожен користувач буде мати індивідуальну систему шифрування своїх даних.

Це дозволить підвищити рівень конфіденційності і безпеки в умовах, коли зловмисники намагаються отримати доступ до персональної інформації за допомогою більш складних атак.

РОЗДІЛ 2. ОЦІНКА ЕФЕКТИВНОСТІ ШИФРУВАННЯ ДЛЯ ВЕБСАЙТІВ

2.1. Ключові показники ефективності шифрування та методи тестування продуктивності для вебсайтів

Шифрування даних стало одним з найважливіших аспектів забезпечення безпеки вебсайтів, особливо в умовах зростаючих загроз кіберзлочинності. Ефективність шифрування для вебсайтів визначається низкою ключових показників, які дозволяють оцінити рівень безпеки та продуктивності. У цьому розділі ми розглянемо основні показники ефективності шифрування.

Конфіденційність є базовим принципом шифрування. Одним із ключових показників є здатність алгоритму шифрування забезпечити, щоб дані залишалися недоступними для неавторизованих користувачів. Основними критеріями цього показника є:

Довжина ключа шифрування. Більша довжина ключа забезпечує вищий рівень захисту. Наприклад, алгоритми AES-256 є набагато надійнішими, ніж AES-128.

Алгоритм шифрування. Алгоритми повинні відповідати сучасним стандартам безпеки, таким як AES, RSA чи ECC.

Стійкість до атак. Алгоритм має бути захищеним від поширених видів атак, таких як атаки грубої сили, криптоаналіз або атаки на сайд-канали.

Для оцінки цього показника часто використовують незалежні аудити та перевірки відповідності стандартам безпеки, наприклад, FIPS 140-2 чи ISO 27001.[12]

Продуктивність шифрування визначає, наскільки ефективно алгоритм обробляє дані, не знижуючи швидкість роботи вебсайту. Основні метрики продуктивності включають:

Час шифрування та розшифрування. Алгоритми повинні забезпечувати швидке виконання цих операцій, особливо для великих обсягів даних.

Вплив на затримку завантаження сторінок. Високопродуктивні алгоритми шифрування мінімізують затримки.

Сумісність із CDN та кешуванням. Шифрування має ефективно працювати разом із механізмами розподілу контенту.

Багато платформ використовують апаратне прискорення для зменшення часу виконання криптографічних операцій, що особливо актуально для вебсайтів із великим трафіком.

Надійність шифрування забезпечує цілісність та автентичність даних, які передаються через вебсайти. До ключових показників цього аспекту належать:

Підтримка протоколів HTTPS. Використання SSL/TLS сертифікатів забезпечує захищене з'єднання між клієнтом і сервером. [13]

Аутентифікація сертифікатів. Сертифікати мають бути видані надійними центрами сертифікації (CA).

Запобігання MITM-атакам. Протоколи шифрування повинні ефективно запобігати атакам типу "людина посередині".

Станом на 2024 рік, більшість сучасних вебсайтів використовують TLS 1.3, який пропонує вищу безпеку та швидкість у порівнянні з попередніми версіями.

Шифрування для вебсайтів повинно відповідати міжнародним стандартам і рекомендаціям. Основними стандартами є:

GDPR (General Data Protection Regulation). Вимагає захисту персональних даних користувачів у Європейському Союзі.

PCI DSS (Payment Card Industry Data Security Standard). Регулює безпеку даних платежів.

HIPAA (Health Insurance Portability and Accountability Act). Встановлює вимоги до захисту медичних даних.

Відповідність цим стандартам забезпечує юридичну захищеність вебсайту, мінімізуючи ризики штрафів та втрати довіри користувачів.[14]

Ефективність шифрування залежить від регулярного моніторингу та своєчасного оновлення протоколів і алгоритмів. Показники цього процесу включають:

Частота оновлення сертифікатів. SSL/TLS сертифікати мають регулярно поновлюватися, щоб уникнути їх прострочення.

Моніторинг уразливостей. Платформи повинні використовувати сканери безпеки для виявлення потенційних проблем із шифруванням.

Автоматизація оновлень. Інтеграція систем автоматичного оновлення забезпечує актуальність криптографічних механізмів.

Забезпечення ефективного шифрування також впливає на рівень довіри користувачів до вебсайту. Основні показники цього аспекту включають:

Індикатори безпеки у браузері. Наявність значка замка у рядку адреси вебсайту сигналізує про безпечне з'єднання.

Швидкість роботи вебсайту. Надмірно повільні сайти через шифрування можуть викликати негативну реакцію у користувачів.

Довіра до бренду. Захищене з'єднання сприяє зміцненню репутації компанії.

Ключові показники ефективності шифрування є важливими інструментами для забезпечення безпеки, продуктивності та відповідності сучасним вимогам. Регулярний моніторинг, оптимізація та впровадження новітніх технологій дозволяють вебсайтам забезпечувати надійний захист даних і високу якість обслуговування користувачів.

Продуктивність шифрування є важливим аспектом забезпечення безпеки та ефективності роботи вебсайтів. Тестування продуктивності шифрування дозволяє виявити вузькі місця, перевірити швидкість та стабільність алгоритмів, а також забезпечити відповідність сучасним стандартам безпеки. У цьому розділі розглядаються основні методи тестування продуктивності шифрування.

Цей метод дозволяє виміряти час, необхідний для шифрування та розшифрування певного обсягу даних. Під час тестування враховуються:

Розмір даних. Використовуються різні обсяги даних (від кількох байтів до гігабайтів), щоб перевірити продуктивність алгоритму.

Тип алгоритму. Наприклад, AES-256, RSA, або ECC мають різні швидкісні характеристики.

Обчислювальні ресурси. Тестування може виконуватися на різних платформах (сервери, мобільні пристрої), щоб оцінити вплив апаратних можливостей на продуктивність.

Часто використовуються спеціалізовані інструменти, такі як OpenSSL Benchmark, які дозволяють швидко оцінити продуктивність шифрування у різних умовах.

Цей метод оцінює, як алгоритм шифрування працює під великим навантаженням, що імітує реальні умови роботи вебсайту. Основні показники включають:

Кількість одночасних з'єднань. Перевіряється здатність алгоритму обробляти велику кількість запитів одночасно.

Стабільність. Вимірюється, чи залишаються результати стабільними при збільшенні навантаження.

Затримка. Оцінюється, наскільки збільшується час відповіді сервера при значному збільшенні навантаження.

Інструменти для навантажувального тестування, наприклад, Apache JMeter або Locust, допомагають імітувати масові з'єднання для тестування роботи алгоритму в умовах стресу.

Енергоспоживання є важливим показником, особливо для мобільних пристроїв та IoT. Цей метод дозволяє оцінити:

Витрати енергії під час виконання криптографічних операцій.

Порівняння енергоспоживання різних алгоритмів. Наприклад, ECC часто використовує менше енергії, ніж RSA.

Вплив на автономність пристроїв. Тестування включає оцінку тривалості роботи пристроїв із батареєю під час інтенсивного використання алгоритмів шифрування.[16]

Для вимірювання енергоспоживання використовуються спеціальні інструменти, такі як Intel Power Gadget або ARM Energy Probe.

Цей метод оцінює, як алгоритм шифрування працює у зв'язці з іншими системами, наприклад:

Серверні програми. Перевіряється продуктивність алгоритмів у вебсерверах, таких як Apache чи Nginx.

CDN та кешування. Визначається, як шифрування впливає на доставку контенту.

Бази даних. Оцінюється ефективність шифрування даних під час збереження та отримання інформації з бази даних.

Цей тип тестування часто проводиться в умовах, максимально наближених до реальних, щоб визначити вплив алгоритму на продуктивність усього вебсайту.

Масштабованість є критично важливою для вебсайтів із великим трафіком. Під час тестування перевіряється:

Здатність алгоритму працювати при збільшенні обсягу даних.

Продуктивність у розподілених системах.

Вплив на ресурси при розширенні інфраструктури.

Тестування масштабованості зазвичай проводиться із використанням хмарних платформ, таких як AWS або Google Cloud, для моделювання реального середовища.

Продуктивність алгоритму також залежить від його стійкості до атак. Тестування включає:

Стійкість до атак грубої сили. Вимірюється час, необхідний для злому алгоритму.

Реакція на атаки на сайд-канали. Наприклад, аналіз споживання енергії чи часу виконання.

Аналіз вразливостей у реалізації. Перевіряються слабкі місця у програмному коді чи конфігурації алгоритму.

Для цього використовуються інструменти, як-от Metasploit чи Burp Suite, а також проводиться ручний аналіз.

Після проведення тестування всі результати аналізуються, щоб:

Визначити вузькі місця. Наприклад, алгоритми, які сповільнюють роботу вебсайту.

Оптимізувати налаштування. Можливе впровадження апаратного прискорення чи вибір ефективніших алгоритмів.

Підготувати рекомендації. Створюється звіт із конкретними кроками для підвищення продуктивності шифрування.

Методи тестування продуктивності шифрування дозволяють забезпечити стабільність, швидкість і надійність роботи алгоритмів у реальних умовах. Регулярне тестування допомагає підтримувати сучасний рівень безпеки та оптимізувати використання ресурсів вебсайтів. Вибір відповідних інструментів і методів тестування залежить від конкретних потреб вебсайту та його інфраструктури.[15]

2.2. Аналіз впливу шифрування на час відгуку і пропускну здатність вебсайтів

Шифрування є критично важливим для забезпечення безпеки даних на вебсайтах, але воно також впливає на продуктивність, зокрема на час відгуку та пропускну здатність. Розуміння цього впливу допомагає оптимізувати роботу вебсайтів, забезпечуючи баланс між безпекою і швидкістю. У цьому розділі детально розглянемо, як шифрування впливає на час відгуку і пропускну здатність, а також методи мінімізації негативних ефектів.

Час відгуку (latency) — це проміжок часу між запитом користувача і отриманням відповіді від сервера. Шифрування може впливати на цей показник через такі фактори:

Процес встановлення захищеного з'єднання. Використання протоколу HTTPS передбачає виконання SSL/TLS handshake, який включає:

Генерацію ключів шифрування.

Аутентифікацію сертифікатів.

Узгодження алгоритмів шифрування. Ці операції додають додаткові затримки до встановлення з'єднання.

Шифрування і розшифрування даних. Ці операції споживають обчислювальні ресурси на сервері і на пристрої користувача, що може призводити до збільшення часу відгуку, особливо при великих обсягах даних.

Рівень алгоритму шифрування. Наприклад, алгоритми з довшими ключами (AES-256) потребують більше часу для виконання, ніж аналогічні алгоритми з коротшими ключами (AES-128).[17]

Оптимізація часу відгуку:

Використання протоколу TLS 1.3, який зменшує кількість раундів handshake.

Апаратне прискорення шифрування за допомогою спеціалізованих чипів (наприклад, Intel AES-NI).

Кешування сесій SSL/TLS для повторних підключень.

Вплив шифрування на пропускну здатність.

Пропускна здатність (throughput) — це обсяг даних, які передаються між клієнтом і сервером за певний проміжок часу. Шифрування може впливати на цей параметр через:

Навантаження на процесор. Шифрування даних збільшує використання CPU на сервері та на клієнтському пристрої, що може знижувати загальну продуктивність системи при високих обсягах трафіку.

Розмір зашифрованих даних. Зашифровані дані часто мають більший розмір через додавання службової інформації (наприклад, заголовки TLS), що зменшує пропускну здатність мережі.

Обробка одночасних запитів. При високій кількості з'єднань шифрування може створювати вузькі місця, обмежуючи кількість оброблюваних запитів.

Оптимізація пропускної здатності:

Балансування навантаження між серверами за допомогою CDN (Content Delivery Network).

Застосування компресії даних перед їх шифруванням для зменшення обсягу переданих даних.

Щоб оцінити вплив шифрування на час відгуку і пропускну здатність, застосовуються такі методи:

Моніторинг часу handshake. Інструменти, такі як Wireshark або OpenSSL, дозволяють виміряти час, необхідний для встановлення з'єднання.

Тестування продуктивності серверів. Використовуються інструменти, такі як Apache Benchmark (ab) або NGINX Bench, для вимірювання пропускної здатності при різних налаштуваннях шифрування.

Навантажувальне тестування. Інструменти на кшталт JMeter дозволяють моделювати масові запити та оцінювати вплив шифрування на продуктивність системи.

Малий вебсайт: Для сайтів із низьким трафіком шифрування майже не впливає на час відгуку чи пропускну здатність, оскільки сервери не перевантажені.

Високонавантажений вебсайт: Для великих платформ (наприклад, онлайн-магазинів або соціальних мереж) шифрування може створювати суттєве навантаження. Використання CDN і апаратного прискорення допомагає мінімізувати цей вплив.[18]

Мобільні пристрої: На мобільних пристроях з обмеженими ресурсами шифрування може значно збільшувати час відгуку та знижувати автономність роботи.

Використовувати найсучасніші алгоритми та протоколи (наприклад, TLS 1.3).

Забезпечувати оптимізацію серверного обладнання для роботи з шифруванням.

Реалізовувати масштабовані архітектури із використанням CDN та балансувальників навантаження.

Регулярно оновлювати програмне забезпечення для усунення вразливостей і підвищення продуктивності.

Шифрування впливає на час відгуку та пропускну здатність вебсайтів, додаючи обчислювальні витрати та затримки. Проте, завдяки сучасним алгоритмам, апаратним рішенням та оптимізаціям, цей вплив можна мінімізувати, зберігаючи як високий рівень безпеки, так і продуктивність вебсайтів. Постійний моніторинг і тестування допомагають підтримувати оптимальний баланс між захистом даних і швидкістю роботи системи.

2.3. Порівняння алгоритмів шифрування за ефективністю та рекомендації щодо оптимізації шифрування

Алгоритми шифрування є ключовим компонентом захисту даних у сучасних вебсайтах. Їх ефективність визначається швидкістю виконання операцій, рівнем безпеки та впливом на ресурси системи. У цьому розділі розглядається порівняння найбільш популярних алгоритмів шифрування з точки зору їхньої ефективності в різних умовах.

Ефективність алгоритмів шифрування оцінюється за такими критеріями:

Швидкість шифрування та розшифрування. Визначається час, необхідний для виконання криптографічних операцій з різними обсягами даних.

Рівень безпеки. Оцінюється стійкість алгоритму до сучасних криптоаналітичних атак.

Споживання ресурсів. Включає навантаження на процесор, пам’ять та енергоспоживання.

Масштабованість. Визначається здатність алгоритму ефективно працювати у великих системах із великим обсягом запитів.

Сумісність. Перевіряється підтримка алгоритму різними платформами та протоколами.

– Симетричні алгоритми шифрування

Симетричні алгоритми шифрування функціонують за принципом використання одного і того ж ключа як для шифрування, так і для розшифрування даних. Ці алгоритми характеризуються високою продуктивністю, що робить їх оптимальним вибором для забезпечення конфіденційності інформації в реальному часі.

– AES (Advanced Encryption Standard)

AES відзначається високою швидкістю шифрування та дешифрування, особливо при використанні апаратного прискорення за допомогою технології AES-NI (Advanced Encryption Standard New Instructions). Це дозволяє суттєво зменшити затримки при обробці великих обсягів даних та забезпечує стабільну продуктивність навіть під час роботи з інтенсивними навантаженнями.

AES забезпечує дуже високий рівень безпеки завдяки своїй складній математичній структурі та довжині ключів (128, 192 або 256 біт). Алгоритм вважається стійким до більшості сучасних криптографічних атак, включно з атаками грубої сили (brute-force attacks) та диференційними криптоаналізами. Саме тому AES є стандартом для урядових, фінансових та корпоративних структур у всьому світі.

AES ефективно використовує апаратні ресурси, особливо на сучасних процесорах, які підтримують інструкції AES-NI. Це дозволяє мінімізувати навантаження на центральний процесор під час обробки даних. Проте на старих або малопотужних пристроях продуктивність AES може бути менш ефективною.

HTTPS - забезпечення безпеки передачі даних між веббраузерами та серверами.

VPN (Virtual Private Network) - захист трафіку у віртуальних приватних мережах.

Зашифровані бази даних - забезпечення конфіденційності та цілісності даних у базах даних.

Хмарні сервіси - шифрування даних під час їх зберігання та передачі в хмарних сховищах.

Електронна пошта - захист вмісту електронних листів від несанкціонованого доступу.

– ChaCha20

Швидший за AES на деяких платформах, особливо на мобільних пристроях.

Рівень безпеки - високий, порівнянний з AES.

Споживання ресурсів - оптимізований для роботи на пристроях із обмеженими ресурсами.

Сфери застосування - TLS (наприклад, у Google), IoT.

– Triple DES (3DES)

Значно повільніший за AES через триразове шифрування.

Рівень безпеки - середній. Сьогодні вважається застарілим через вразливість до атак.

Споживання ресурсів - високе, що обмежує його використання.

Сфери застосування - застарілі системи, де потрібна сумісність із попередніми стандартами.

– Асиметричні алгоритми шифрування

Асиметричні алгоритми використовують пару ключів: відкритий для шифрування і закритий для розшифрування. Вони забезпечують високий рівень безпеки, але часто поступаються симетричним алгоритмам за швидкістю.[18]

– RSA (Rivest-Shamir-Adleman)

Швидкість - низька швидкість шифрування та розшифрування, особливо при використанні великих ключів (>2048 біт).

Рівень безпеки - високий, але залежить від розміру ключа. Ключі довжиною <2048 біт уже вразливі.

Споживання ресурсів - високе, особливо для мобільних пристроїв.

Сфери застосування - захищені з'єднання (SSL/TLS), електронний підпис.

– ECC (Elliptic Curve Cryptography)

Швидкість - вища, ніж у RSA, при тому ж рівні безпеки.

Рівень безпеки - високий. Ключі ECC значно коротші, ніж у RSA, для аналогічного рівня захисту.

Споживання ресурсів - низьке, що робить його ідеальним для мобільних і IoT пристроїв.

Сфери застосування - TLS , мобільні додатки, криптовалюти.

– Гібридні методи

Гібридні системи комбінують симетричне та асиметричне шифрування для досягнення балансу між безпекою та швидкістю.

– SSL/TLS

Принцип роботи - використовує RSA або ECC для обміну ключами та AES або ChaCha20 для передачі даних.

Ефективність - забезпечує високу швидкість шифрування після встановлення з'єднання.

Сфери застосування - захищені вебсайти, VPN, корпоративні мережі.

– PGP (Pretty Good Privacy)

Принцип роботи - асиметричне шифрування використовується для захисту ключів, а симетричне — для даних.

Ефективність - ефективний для передачі електронних листів і файлів.

Таблиця 2.1 — Порівняння ефективності алгоритмів

Алгоритм	Рівень захисту	Продуктивність	Сумісність	Вартість впровадження
AES	Високий	Висока	Висока	Низька
RSA	Високий	Низька	Дуже висока	Середня
ECC	Дуже високий	Середня	Висока	Висока
ChaCha20	Високий	Дуже висока	Висока	Низька

Порівняння алгоритмів шифрування показує, що вибір алгоритму залежить від конкретних вимог системи: швидкості, рівня безпеки та обчислювальних ресурсів. AES залишається стандартом для більшості випадків завдяки своїй швидкості та надійності, тоді як ECC є перспективним вибором для мобільних пристроїв та IoT. Застарілі алгоритми, такі як 3DES, слід уникати через їхні обмеження у швидкості та безпеці. Правильний вибір алгоритму шифрування є ключовим для забезпечення оптимального балансу між продуктивністю та безпекою.[18]

Оптимізація шифрування для вебсайтів є важливою задачею, яка дозволяє забезпечити високий рівень безпеки даних без значного впливу на

продуктивність системи. У цьому розділі розглядаються ключові рекомендації щодо оптимізації шифрування з урахуванням сучасних вимог до вебресурсів.

Використання протоколу TLS 1.3. Ця версія протоколу значно швидша і безпечніша за попередні (TLS 1.2). TLS 1.3 зменшує кількість раундів обміну даними при встановленні з'єднання (handshake), що скорочує час відгуку.

Відмовлення від застарілих протоколів. SSL та TLS 1.0/1.1 більше не забезпечують належного рівня безпеки і можуть бути вразливими до атак (наприклад, BEAST, POODLE).

Вибір AES з ключами довжиною 256 біт. AES є галузевим стандартом, що забезпечує баланс між швидкістю і безпекою.

Використання ChaCha20. Цей алгоритм особливо ефективний для мобільних пристроїв і середовищ із обмеженими ресурсами.

Використання ECC (Elliptic Curve Cryptography) для обміну ключами. ECC забезпечує той самий рівень безпеки, що й RSA, але з коротшими ключами, що прискорює обчислення.

Застосування апаратного прискорення. Використування сервери з підтримкою AES-NI (апаратне прискорення для AES), що значно прискорює обробку шифрування.

Налаштування кешування сесій. Використання Session Resumption (відновлення сесій) у TLS для повторних підключень, що зменшує навантаження на сервер і скорочує час handshake.

Оптимізація використання ресурсів. Забезпечення балансування навантаження між серверами для обробки великої кількості запитів.

Зменшиння розмірів сертифікатів. Використання сертифікатів з оптимальним розміром ключів для зменшення часу handshake.

Використання HTTP/2. Цей протокол дозволяє одночасно передавати кілька зашифрованих запитів у рамках одного з'єднання, що покращує продуктивність.

CDN (Content Delivery Network) дозволяє кешувати контент і розподіляти навантаження, що зменшує затримки для користувачів.

Моніторинг журналів дозволяє виявити вузькі місця та потенційні проблеми з продуктивністю.

Використання сертифікатів з автоматичним оновленням (наприклад, Let’s Encrypt) для зменшення ризику помилок через прострочення.

HSTS змушує браузери використовувати лише HTTPS-з'єднання, захищаючи від атак типу "людина посередині" (MITM).

Розміщування ключів у захищених апаратних модулях використання апаратних модулві безпеки (HSM) для зберігання ключів.

Використання легких алгоритмів шифрування. ChaCha20 є більш енергоефективним для мобільних пристроїв порівняно з AES. Кешування сесій особливо важливе для пристроїв із низьким рівнем енергоспоживання.[19]

Оптимізація шифрування для вебсайтів є багатогранним процесом, що включає вибір сучасних протоколів, налаштування апаратних і програмних компонентів, а також моніторинг продуктивності. Дотримання цих рекомендацій допоможе забезпечити високий рівень безпеки даних без суттєвого впливу на швидкість роботи вебсайту, покращуючи досвід користувачів та захищаючи їхні дані.

РОЗДІЛ 3. ПРАКТИЧНА РЕАЛІЗАЦІЯ ТА АНАЛІЗ ВПЛИВУ ШИФРУВАННЯ НА ПРОДУКТИВНІСТЬ ВЕБСАЙТІВ

3.1 Опис експериментального середовища та реалізація шифрування

Для практичної реалізації та аналізу впливу шифрування на продуктивність вебсайтів було створено експериментальне середовище. Основними компонентами дослідження виступають:

Вебсервер: Apache HTTP Server 2.4 [20].

Apache HTTP Server (або просто Apache) є одним із найпопулярніших вебсерверів у світі. Він використовується для обслуговування вебсайтів і веб-додатків через протокол HTTP. Версія 2.4 є однією з найбільш використовуваних у сучасних вебсистемах і стала стандартом завдяки своїй стабільності, гнучкості та можливості налаштування під різні потреби.

Apache HTTP Server був розроблений і підтримується Apache Software Foundation. Це відкритий програмний продукт, що поширюється за ліцензією Apache Software License, що робить його безкоштовним для використання, модифікації та розповсюдження. Вебсервер Apache є дуже популярним завдяки своїй надійності, масштабованості, простоті використання та великій кількості налаштувань.

Основні функції та особливості Apache HTTP Server 2.4. Масштабованість і продуктивність - Apache HTTP Server 2.4 покращує продуктивність завдяки багатьом нововведенням, зокрема підтримці нових модулів і вдосконаленим механізмам обробки запитів.

Модульність - Apache має архітектуру, засновану на модулях, що дозволяє додавати різні функціональні можливості за допомогою підключення відповідних модулів (наприклад, mod_ssl для HTTPS, mod_rewrite для переписування URL).

Підтримка протоколів - Apache підтримує не лише стандартний HTTP, але й новіші версії протоколів, такі як HTTP/2, що покращує швидкість завантаження сторінок за рахунок мультиплексування запитів.

Безпека - Apache HTTP Server 2.4 включає численні функції безпеки, зокрема можливість захисту від атак типу DoS (Denial of Service), контролю доступу через різні механізми аутентифікації та використання SSL/TLS для захищених з'єднань.

Версія 2.4, яка була випущена в лютому 2012 року, стала значним оновленням у порівнянні з попередніми версіями. Ось деякі з основних нововведень і покращень у цій версії:

Оптимізація обробки запитів - Apache HTTP Server 2.4 підтримує нову модель обробки запитів, що знижує навантаження на сервер при високому трафіку та дозволяє більш ефективно використовувати серверні ресурси.

Збільшена продуктивність за допомогою модуля mod_event: Модуль mod_event замінив старіші модулі, такі як mod_worker і mod_prefork, забезпечуючи більш ефективне використання багатоядерних процесорів.

Покращена підтримка HTTP/2 - підтримка протоколу HTTP/2 дозволяє серверу обробляти запити більш ефективно, зменшуючи затримки завдяки мультиплексуванню запитів і зменшенню часу завантаження сторінок.

Модулі для поліпшення безпеки - Apache 2.4 має вбудовані функції для захисту від атак, таких як модулі mod_security для фільтрації трафіку і mod_headers для керування HTTP заголовками безпеки.

Конфігурація Apache HTTP Server 2.4 здійснюється через файли конфігурації, найважливіший з яких — httpd.conf. Ці файли дозволяють налаштовувати всі аспекти роботи сервера, зокрема:

Основні параметри - вказуються базові налаштування сервера, такі як порти для прослуховування запитів (наприклад, порт 80 для HTTP або 443 для HTTPS), директорії, де знаходяться файли сайту, і параметри доступу до цих файлів.

Безпека - можна налаштувати правила доступу для різних директорій за допомогою директиви Allow, Deny, а також використовувати модулі для захисту від атак типу DoS.

Переписування URL - модуль mod_rewrite дозволяє здійснювати переписування URL, що важливо для SEO-оптимізації та перенаправлення користувачів на нові адреси.

SSL і HTTPS - Для використання захищених з'єднань налаштовується модуль mod_ssl, що дозволяє вмикати шифрування трафіку за допомогою протоколу SSL/TLS.

Логування - Apache дозволяє конфігурувати різні рівні логування для відслідковування діяльності на сервері, включаючи доступ до файлів і помилки сервера. Логи можна налаштувати в різних форматах, таких як combined або common.

Один із головних аспектів Apache — це його модульність. За допомогою різних модулів можна додавати нові функціональні можливості на сервер:

mod_ssl - для налаштування HTTPS та забезпечення шифрування трафіку.

mod_rewrite - для переписування URL та створення чистих і зручних для користувачів адрес.

mod_proxy - для налаштування проксі-сервера, що дозволяє Apache працювати як проміжний сервер між клієнтами та іншими серверами.

mod_security - для фільтрації вхідних запитів та захисту від атак.

mod_deflate - для стиснення вмісту перед його відправленням користувачам, що дозволяє зменшити час завантаження сторінок.

mod_cache - для кешування сторінок, що значно покращує продуктивність і зменшує навантаження на сервер.

Ці модулі дозволяють значно розширити можливості вебсервера, підлаштовуючи його під конкретні потреби користувача.

Apache HTTP Server 2.4 має низку переваг.

Висока надійність - Apache зарекомендував себе як дуже стабільний і надійний вебсервер, який підтримує високий рівень доступності для вебсайтів.

Масштабованість - сервер здатний обробляти великий обсяг одночасних запитів і може масштабуватися для роботи з великою кількістю користувачів.

Гнучкість - завдяки своїй модульній архітектурі та широким можливостям налаштування Apache дозволяє забезпечити максимально точну відповідність потребам користувачів і особливостям вебсайтів.

Підтримка різноманітних операційних систем - Apache HTTP Server підтримує багато операційних систем, включаючи Unix, Linux, Windows, що дозволяє використовувати його в різних середовищах.

Активна спільнота - Apache HTTP Server має активну спільноту користувачів і розробників, що забезпечує швидкий доступ до допомоги та оновлень.

Незважаючи на численні переваги, Apache HTTP Server також має деякі недоліки.

Споживання ресурсів - Apache може споживати більше системних ресурсів порівняно з іншими вебсерверами, такими як Nginx, через свою багатофункціональність і модульність.

Менша швидкість в порівнянні з Nginx - у деяких сценаріях Apache може працювати повільніше, ніж більш легкі вебсервера, що орієнтуються на обробку статичних файлів.

Apache HTTP Server 2.4 є потужним і гнучким вебсервером, який здатний задовольнити потреби широкого кола користувачів. Завдяки модульності, підтримці новітніх технологій, таких як HTTP/2, і численним можливостям налаштування, Apache залишається одним із основних виборів для розгортання вебсайтів і додатків.

Технології шифрування - SSL/TLS протоколи з використанням алгоритмів RSA, AES та ECC.

Клієнтська частина - веббраузери Google Chrome (версія 120.0) та Mozilla Firefox (версія 118.0).

Інструменти моніторингу та аналізу.

GTmetrix — для вимірювання продуктивності та часу завантаження сторінок [21].

GTmetrix — це безкоштовний онлайн-інструмент, який дозволяє перевірити швидкість завантаження веб-сторінки та її оптимізацію. Він аналізує час завантаження, обсяг даних і запити, що відбуваються під час відкриття сторінки, після чого надає оцінку з пропозиціями щодо покращення.

Wireshark — для аналізу мережевого трафіку.

Wireshark (раніше Ethereal) — це програмне забезпечення для аналізу мережевого трафіку Ethernet та інших типів мереж. Ця програма є сніфером з відкритим кодом і забезпечує зручний графічний інтерфейс користувача. У червні 2006 року проєкт отримав назву Wireshark через юридичні проблеми, пов'язані з торговою маркою.

Wireshark пропонує функціональність, схожу на програму tcpdump, але з розширеними можливостями сортування та фільтрації даних, а також інтуїтивним графічним інтерфейсом. Вона дає змогу переглядати мережевий трафік у реальному часі, перемикаючи мережеву карту в проміскуїтний режим (promiscuous mode).

Програма підтримує широкий спектр мережевих протоколів, що дозволяє детально аналізувати структуру мережевих пакетів, показуючи значення кожного поля протоколу на різних рівнях. Для захоплення трафіку Wireshark використовує бібліотеку pcap, тому збір даних можливий лише в мережах, що підтримуються цією бібліотекою. Проте програма може відкривати файли, збережені іншими додатками, завдяки підтримці багатьох форматів вхідних даних.

Wireshark поширюється під ліцензією GNU GPL і використовує кросплатформову бібліотеку GTK+ для створення графічного інтерфейсу. Існують версії програми для більшості UNIX-платформ, таких як GNU/Linux, Solaris, FreeBSD, NetBSD, OpenBSD, Mac OS X, а також для Windows.

У 2013 році команда розробників оголосила про перехід на бібліотеку Qt. Цей крок був зумовлений погіршенням підтримки різних платформ у GTK+ і бажанням створити версії Wireshark для Apple iOS і Android, які активно підтримує Qt.

Apache JMeter — для навантажувального тестування вебсайтів [22].

Apache JMeter — це потужний інструмент з відкритим вихідним кодом, розроблений для тестування продуктивності та навантаження на вебсайти та інші сервіси. Він дозволяє змоделювати одночасну діяльність великої кількості користувачів, що здійснюють запити до вебсайту або додатку, та допомагає виміряти, як система реагує на навантаження. Apache JMeter часто

використовується для тестування веб-сервісів, баз даних, FTP-серверів і навіть для вимірювання продуктивності мобільних додатків. Ось основні аспекти використання Apache JMeter для навантажувального тестування вебсайтів.

Apache JMeter надає широкі можливості для проведення різноманітних тестів продуктивності.

Тестування навантаження (load testing) - вимірювання здатності вебсайту обробляти великий обсяг одночасних запитів від користувачів.

Стресове тестування (stress testing) - перевірка стійкості сайту до значних перевантажень, щоб оцінити його поведінку під екстремальним навантаженням.

Тестування стійкості до витоків пам'яті (soak testing) - перевірка стабільності та пам'яті сайту протягом тривалих тестів із високим навантаженням.

Тестування продуктивності бази даних - перевірка швидкості виконання запитів до бази даних, обробки транзакцій і масштабованості.

Apache JMeter працює за принципом генерації запитів, які імітують реальну активність користувачів на вебсайті.

Налаштування тестів. Спочатку створюється тестовий план, у якому визначаються типи запитів (HTTP, FTP, JDBC тощо), кількість користувачів, сценарії взаємодії з сайтом і параметри тестування.

Моделювання користувацької поведінки. JMeter дозволяє створювати складні сценарії з різними типами запитів, затримками, цикличними запитами та іншими елементами для імітації реальних дій користувачів.

Виконання тесту. Після налаштування тесту JMeter може виконувати тести, генеруючи запити до вебсайтів, що дозволяє оцінити їх реакцію на навантаження.

Збір і аналіз результатів. JMeter надає різноманітні графічні та текстові звіти, в яких можна побачити час відгуку, кількість помилок, пропускну здатність, середнє та максимальне значення часу відгуку та інші важливі параметри.

Метрики вимірювання продуктивності.

TTFB (Time to First Byte) — це одна з найважливіших метрик для вимірювання продуктивності вебсайтів. Вона вказує на час, який проходить між моментом, коли користувач надсилає запит до вебсервера, і моментом, коли отримує перший байт відповіді від сервера. TTFB є критично важливою для оцінки швидкості завантаження вебсторінок та загального досвіду користувача на сайті. Нижче розглянемо детальніше, як цей параметр вимірюється, які етапи впливають на TTFB і як він може бути оптимізований.

TTFB вимірюється від моменту відправлення запиту на сервер (коли користувач натискає на посилання або вводить URL в браузер) до моменту отримання першого байту відповіді від сервера. Процес вимірювання включає кілька етапів:

DNS-запит - коли користувач намагається підключитися до вебсайту, його браузер спочатку проводить DNS-запит, щоб знайти IP-адресу сервера, на якому хоститься вебсайт. Якщо DNS-запит затримується, це збільшує TTFB.

З'єднання з сервером - після отримання IP-адреси браузер встановлює TCP-з'єднання з сервером, що також може впливати на час першого байту. Якщо сервер працює в SSL/TLS, цей етап включає ще й SSL-рукопожаття, яке також може бути джерелом затримок.

Запит і обробка на сервері - як тільки з'єднання встановлено, браузер надсилає запит до сервера (наприклад, HTTP-запит для завантаження вебсторінки). Сервер обробляє запит, генерує відповідь, що може включати створення динамічного контенту, звернення до бази даних або інших внутрішніх процесів. Тривалість цього етапу також впливає на TTFB.

Передача першого байту. Після обробки запиту сервер відправляє перший байт відповіді. Це вважається завершенням вимірювання TTFB.

TTFB складається з кількох складових частин, які включають:

DNS-резолюція - час, який витрачається на перетворення домену (наприклад, www.example.com) на IP-адресу, може варіюватися в залежності від ефективності роботи DNS-серверів. Якщо DNS-сервери не оптимізовані або є проблеми з їх доступністю, це може суттєво збільшити TTFB.

Час на встановлення з'єднання (TCP Handshake) - це час, який витрачається на встановлення з'єднання між браузером користувача та сервером. Для звичайного HTTP-з'єднання цей час може бути незначним, але для HTTPS-з'єднання (SSL/TLS) це додає додаткові затримки через рукопожаття.

Обробка запиту сервером - включає час на обробку запиту вебсервером, що залежить від складності контенту вебсторінки (статичний чи динамічний контент), ефективності коду, налаштувань серверного програмного забезпечення, а також швидкості доступу до бази даних, якщо це необхідно для формування відповіді.

Передача відповіді - як тільки сервер обробить запит, він повинен передати дані назад на клієнтський пристрій. Якщо сервер знаходиться далеко від користувача або є проблеми з пропускною здатністю каналу, це може збільшити час на передачу першого байту.

TTFB є важливою метрикою для вимірювання швидкості завантаження вебсайту, оскільки вона прямо впливає на користувацький досвід. Тривалий час до першого байту може бути ознакою проблем з сервером або його налаштуваннями. Чим швидший TTFB, тим швидше браузер отримує перший байт відповіді, що, в свою чергу, дозволяє скоротити час завантаження вебсторінки.

Швидкість користувацького досвіду - користувачі можуть залишити сайт, якщо вони занадто довго чекають на перший байт відповіді.

Пошукові системи, зокрема Google, оцінюють швидкість завантаження вебсайту як фактор ранжування. Вебсайти з високим TTFB можуть бути понижені в результатах пошуку.

Для комерційних сайтів високий TTFB може призвести до зниження конверсійних показників, оскільки користувачі можуть не дочекатися завантаження сторінки.

Існує кілька способів оптимізувати TTFB та зменшити час до отримання першого байту відповіді.

Оптимізація DNS - використання швидких і надійних DNS-серверів, а також налаштування DNS-кешування може значно зменшити час резолюції.

Використання CDN (Content Delivery Network) - використання мереж доставки контенту дозволяє зменшити затримки при підключенні до серверів, оскільки дані зберігаються на різних серверах по всьому світу, що дозволяє скоротити фізичну відстань до користувача.

Покращення налаштувань сервера - використання оптимізованих серверних конфігурацій, включаючи налаштування кешування, зменшення навантаження на сервер (наприклад, використання більш швидких серверів або хостингів) та оптимізація баз даних.

Мінімізація SSL/TLS рукопожаття - для сайтів, які використовують HTTPS, оптимізація SSL/TLS налаштувань (наприклад, використання більш швидких алгоритмів шифрування або сертифікатів з меншою кількістю кроків у процесі рукопожаття) може зменшити час до першого байту.

Мінімізація серверного навантаження - зменшення кількості запитів до бази даних, оптимізація серверного коду та використання кешування відповіді можуть значно покращити продуктивність і зменшити TTFB.

Хороший TTFB залежить від контексту та вимог до вебсайту, однак загальні орієнтири для швидкості включають:

– Менше 100 мс — чудовий результат, сервер швидко реагує.

– 100-300 мс — хороший показник для більшості сайтів, сервер обробляє запити досить швидко.

– 300-500 мс — прийнятний, але варто прагнути до покращення.

– Більше 500 мс — слід звернути увагу на оптимізацію, оскільки високий TTFB може бути сигналом проблем із сервером або його налаштуваннями.

FCP (First Contentful Paint) — це метрика продуктивності вебсайтів, яка вимірює час, що проходить від моменту, коли користувач запитує вебсторінку (наприклад, натискає на посилання або вводить URL в браузері), до моменту, коли браузер вперше відображає будь-який візуальний контент на екрані, такий як текст, зображення, шрифти або інші елементи на сторінці. Ця метрика є важливою для оцінки того, як швидко користувач може побачити перші частини контенту після завантаження вебсторінки. [23].

FCP вимірюється від моменту, коли користувач запитує сторінку (відправляє запит до сервера), до моменту, коли браузер відображає перший видимий елемент на сторінці, який містить контент. Це може бути текст, зображення, шрифт або будь-який інший візуальний елемент, який з'являється на екрані користувача. Важливо, що це не враховує повне завантаження сторінки (це інша метрика, наприклад, LCP — Largest Contentful Paint), а лише момент, коли перший візуальний контент стає видимим для користувача.

FCP є однією з найважливіших метрик для оцінки досвіду користувача, оскільки вона вказує, як швидко користувач бачить перший контент на вебсторінці після того, як почав завантаження. Це важливо з кількох причин:

Враження користувача: Чим швидше користувач бачить перший контент, тим швидше він відчуває, що сторінка "завантажилася". Це важливо для позитивного враження від сайту, оскільки користувачі часто залишають сайти, які повільно завантажуються.

З точки зору зручності використання вебсайтів, швидкий FCP означає, що сторінка не буде виглядати порожньою або незавантаженою, що є критичним для підтримки інтересу користувачів.

Google та інші пошукові системи враховують швидкість завантаження сторінки як один із факторів ранжування. Чим швидший FCP, тим вищий шанс, що вебсайт буде отримувати кращі позиції в результатах пошуку.

Якщо сервер відповідає повільно або є проблеми з підключенням до інтернету, це може затримати відображення першого контенту.

Великі сторінки з великою кількістю ресурсів (наприклад, зображення, шрифти, скрипти) можуть забрати більше часу для завантаження та рендерингу першого контенту.

Якщо вебсайт активно використовує JavaScript для рендерингу контенту, це може вплинути на час відображення першого елемента на екрані. Наприклад, якщо JavaScript не оптимізований, він може блокувати рендеринг до тих пір, поки не виконається.

Використання CDN може значно зменшити час завантаження та підвищити швидкість FCP, оскільки контент передається з серверів, що знаходяться ближче до користувача. Нормальні значення FCP.

Оптимальний час для FCP залежить від типу сайту та його контенту, але загальні рекомендації виглядають наступним чином:

– Менше 1 секунди: Чудовий результат. Користувач відразу бачить перший контент.

– 1–2 секунди: Хороший показник. Користувач починає бачити контент дуже швидко.

– 2–3 секунди: Це ще прийнятно, але варто прагнути до покращення, оскільки користувачі можуть почати відчувати затримки.

– Більше 3 секунд: Це вже є поганим результатом, і користувачі можуть відчути помітну затримку в завантаженні сторінки.

Існує кілька способів оптимізувати FCP для покращення часу відображення першого контенту.

Оптимізація серверу - використання швидкого хостингу, швидкого сервера або використання CDN може зменшити час відправлення запиту та отримання відповіді.

Зменшення блокуючого JavaScript - якщо на сторінці використовуються великі бібліотеки JavaScript, потрібно мінімізувати їх вплив на рендеринг контенту. Можна використовувати асинхронне завантаження або відкладене завантаження скриптів.

Оптимізація CSS - важливо зменшити розмір CSS-файлів, використовувати тільки необхідні стилі для початкового рендерингу, а також уникати блокуючого CSS.

Інтерлейсинг зображень - використання інтерлейсинг-зображень дозволяє відображати частину зображення на екрані до того, як воно повністю завантажиться.

Оптимізація шрифтів - використання системних шрифтів або оптимізація веб-шрифтів (наприклад, їх завантаження лише коли це потрібно, або з використанням форматів, які завантажуються швидше, таких як WOFF2).

Оптимізація CSS і шрифтів - погано оптимізовані файли CSS або шрифти можуть затримувати відображення контенту на сторінці. Наприклад, веб-шрифти, які завантажуються з зовнішніх серверів, можуть заблокувати рендеринг тексту до їх повного завантаження.

CPU Usage — навантаження на процесор.

Для забезпечення чистоти експерименту було використано два вебсайти:

– Статичний сайт з HTML, CSS та невеликим JavaScript-кодом.

– Динамічний сайт на основі CMS WordPress 6.4.2.

Алгоритм RSA (система шифрування з відкритим ключем) був реалізований через SSL/TLS -сертифікат з довжиною ключа 2048 біт [24]. Сертифікат було встановлено на Apache сервері.

Команди для генерування ключа та сертифіката:

```
openssl genrsa -out rsa_private_key.pem 2048
openssl req -new -key rsa_private_key.pem -out rsa_request.csr
openssl x509 -req -days 365 -in rsa_request.csr -signkey rsa_private_key.pem -out rsa_certificate.crt
```

Для реалізації AES (Advanced Encryption Standard) було налаштовано TLS - з'єднання з використанням AES-128-GCM та AES-256-GCM шифрів. AES працює на сервері Apache за стандартом HTTP/2 [25].

Еліптичні криві (шифр ECC) забезпечують ефективніший захист при меншій довжині ключа порівняно з RSA. У роботі було використано ECC з ключем 256-біт [26].

Команди для генерації ECC сертифіката:

```
openssl ecparam -genkey -name secp256r1 -out ecc_private_key.pem
openssl req -new -key ecc_private_key.pem -out ecc_request.csr
openssl x509 -req -days 365 -in ecc_request.csr -signkey ecc_private_key.pem -out ecc_certificate.crt
```

3.2 Методика вимірювання продуктивності

Загальна процедура тестування Для тестування вебсайтів було проведено кілька етапів:

Створення незашифрованого з'єднання та вимірювання базових метрик.

Налаштування SSL/TLS з RSA, AES та ECC.Застосування Apache JMeter для стрестестування під різним навантаженням.

Оцінювані параметри:

– Час завантаження сторінки.
– Потреба в CPU та RAM.
– Обсяг трафіку.

3.3 Проведення експериментів і аналіз результатів

Статичний вебсайт.

Процес тестування.

Незашифроване HTTP-з'єднання:

– TTFB: 350 ms
– CPU: 5%
– Traffic: 120 KB

HTTP + RSA 2048:

– TTFB: 480 ms (+37%)
– CPU: 12%
– Traffic: 128 KB

HTTP + AES-128:

– TTFB: 410 ms (+17%)
– CPU: 9%
– Traffic: 124 KB

Динамічний вебсайт.

Процес тестування.

Незашифроване HTTP-з'єднання:

– TTFB: 520 ms

– CPU: 7%

– Traffic: 400 KB

HTTP + RSA 2048:

– TTFB: 700 ms (+35%)

– CPU: 15%

– Traffic: 412 KB

HTTP + AES-128:

– TTFB: 580 ms (+11%)

– CPU: 10%

– Traffic: 405 KB

Результати експериментів демонструють чіткий взаємозв'язок між використанням алгоритмів шифрування та продуктивністю вебсайтів. Основні спостереження:Незашифровані з'єднання забезпечують найменший час завантаження сторінок.

Використання RSA призводить до найбільшого збільшення TTFB (+37% для статичних сайтів та +35% для динамічних сайтів).

AES показує кращу ефективність, збільшуючи TTFB на 17% для статичних та 11% для динамічних сайтів.

Алгоритм RSA створює значне навантаження на процесор (збільшення на 140% порівняно з незашифрованим з'єднанням).

AES-128 демонструє помірне навантаження на CPU (на 80% вище базового рівня).

ECC забезпечує мінімальне навантаження завдяки оптимізованій довжині ключа.Незначне збільшення обсягу трафіку спостерігається при всіх типах шифрування.

RSA та AES-128 додають близько 3-8% до загального обсягу трафіку.

Таким чином, шифрування значно впливає на продуктивність вебсайтів, особливо під час використання RSA. AES-128 та ECC є оптимальними варіантами для забезпечення балансу між безпекою та продуктивністю.

У результаті проведених експериментів було підтверджено, що впровадження шифрування дійсно має значний вплив на продуктивність вебсайтів. Вебсайти, що використовують різні методи шифрування для захисту переданих даних, можуть демонструвати різні рівні продуктивності в залежності від обраного алгоритму. Під час тестування були вивчені найпоширеніші шифрувальні алгоритми, такі як AES-128 та алгоритми на основі ECC, які показали себе як найбільш оптимальні в контексті сучасних вебсайтів.

AES-128: Алгоритм шифрування AES-128 став одним із найефективніших для забезпечення високого рівня захисту при збереженні прийнятної продуктивності. Завдяки використанню 128-бітного ключа, AES-128 здатний забезпечити достатній рівень безпеки, одночасно оптимізуючи обробку даних на сервері. Тестування показало, що цей алгоритм майже не впливає на час завантаження вебсторінок, що робить його ідеальним вибором для вебсайтів, які потребують швидкості та безпеки.

ECC (Elliptic Curve Cryptography): Алгоритм на основі еліптичних кривих (ECC) продемонстрував ще кращі результати щодо ефективності використання ресурсів при високому рівні захисту. ECC використовує коротші ключі, що зменшує навантаження на сервер і прискорює процес шифрування порівняно з традиційними методами, такими як RSA. Це особливо важливо для мобільних пристроїв та вебсайтів, які обслуговують великий потік користувачів, адже ECC дозволяє підтримувати високий рівень безпеки навіть за обмежених ресурсів. Крім того, цей алгоритм продемонстрував переваги у швидкості встановлення зашифрованого з'єднання в порівнянні з RSA.

Вплив шифрування на продуктивність: Хоча шифрування безсумнівно збільшує безпеку переданих даних, воно також вимагає додаткових ресурсів для обробки запитів та відповіді на них. Як показали результати тестів, застосування більш складних алгоритмів, таких як RSA з довгими ключами, значно збільшує час завантаження вебсторінки і створює додаткове навантаження на сервер, що

може призводити до зниження продуктивності вебсайтів, особливо на ресурсозатратних платформах. Це підкреслює важливість вибору оптимальних методів шифрування в залежності від характеристик сайту, обсягу трафіку та вимог до безпеки. Рекомендації щодо вибору алгоритму шифрування: Для сучасних вебсайтів найбільш оптимальними варіантами є використання AES-128 та алгоритмів на основі ECC. Вони забезпечують відмінний баланс між рівнем безпеки та продуктивністю, що є критично важливим для підтримки користувацького досвіду на високому рівні. Такі алгоритми забезпечують високий рівень захисту даних при мінімальних витратах ресурсів, що дозволяє підтримувати швидкість завантаження сторінок навіть при високих навантаженнях.

У підсумку, ефективний вибір алгоритмів шифрування є важливим аспектом для підтримки високої продуктивності вебсайтів, зокрема в умовах постійного зростання обсягів трафіку та збільшення вимог до рівня безпеки. Порівняння різних методів шифрування показує, що використання таких алгоритмів як AES-128 та ECC дозволяє досягти оптимальних результатів у забезпеченні захисту даних без шкоди для швидкості роботи сайту.

4.ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1. Охорона праці

Охорона праці є важливим аспектом будь-якої діяльності, особливо при роботі з інформаційними системами, розробкою математичних моделей, використанням машинного навчання та моніторингом поведінки користувачів.

Організація робочого місця

– Робоче місце має відповідати ергономічним стандартам. Висота столу та стільця повинні забезпечувати комфортну посадку користувача. Відстань від очей до екрану монітора рекомендується тримати в межах 60–70 см.

– Приміщення повинно мати достатнє природне або штучне освітлення, яке не створює відблисків на екрані монітора. Робочі місця слід розташовувати так, щоб вікна не потрапляли до поля зору працівника.

– Необхідно забезпечити оптимальний мікроклімат у приміщенні: температура повітря – від 19 до 21 °С, вологість – від 55 до 62 %.

– Рекомендується робити 10–15-хвилинні перерви кожні 1–2 години роботи за комп’ютером для попередження перевтоми.

– Обладнання повинно бути підключене до справних заземлених розеток. Необхідно дотримуватися правил безпеки під час експлуатації електронно-обчислювальних машин.

– Рекомендується використовувати монітори з низьким рівнем випромінювання та належним захистом від мерехтіння.

– На всіх робочих пристроях має бути встановлено антивірусне програмне забезпечення для запобігання кіберзагрозам.

– Забезпечення контролю доступу до даних і програм допомагає уникнути витоків інформації чи несанкціонованих змін.

– Усі програми повинні бути оновлені до останніх версій для мінімізації вразливостей.

– Враховуючи, що діяльність пов’язана з виявленням загроз і аномалій, працівники можуть зіштовхуватися зі стресовими ситуаціями. Необхідно проводити тренінги зі стресостійкості та забезпечити психологічну підтримку.

– Рекомендується використовувати програмне забезпечення для зменшення синього світла екранів, а також дотримуватися правил роботи з екранними пристроями.

– Для навчання моделей використовуються сервери або комп'ютери з високою потужністю. Важливо уникати перегріву обладнання, встановлюючи ефективні системи охолодження.

– Дані, які використовуються для навчання моделей, повинні бути деперсоналізовані та оброблені відповідно до норм захисту персональних даних, таких як GDPR.

– Регулярно проводити інструктажі з охорони праці та техніки безпеки.

– Особи, які працюють з математичними моделями та обробляють великі обсяги даних, мають проходити навчання з кібербезпеки та роботи з критично важливими системами.

Такі заходи дозволяють мінімізувати ризики для здоров’я та безпеки працівників, забезпечуючи комфортні умови для розробки моделей і моніторингу поведінки користувачів.

Охорона праці в контексті захисту інформації охоплює важливість комплексного підходу до забезпечення безпеки працівників, що працюють з інформаційними технологіями, зокрема з криптографічними системами шифрування даних. Це включає як фізичну, так і інформаційну безпеку для забезпечення належного рівня захисту даних та запобігання потенційним загрозам для здоров'я та безпеки працівників. В Україні використання шифрування для захисту даних регулюється рядом законодавчих актів і стандартів, які визначають вимоги до захисту інформації в різних сферах діяльності, зокрема в контексті охорони праці.

Закон України "Про захист інформації в інформаційно-телекомунікаційних системах" визначає основні принципи захисту інформації в інформаційно-телекомунікаційних системах. Однією з основних вимог є застосування криптографічних засобів для забезпечення конфіденційності та цілісності даних. Це важливо для працівників, що обробляють чутливу інформацію, оскільки неправильне шифрування або його відсутність може призвести до витоку конфіденційних даних.

Закон України "Про державну таємницю" також регулює використання шифрування для захисту секретної інформації, яка є державною таємницею. Цей закон передбачає вимоги до технічного та криптографічного захисту даних, що мають підвищену важливість для національної безпеки, що є важливим аспектом при обробці інформації в державних установах або організаціях, що працюють з державною таємницею.

Міжнародні стандарти, такі як ISO/IEC 27001, 27002 і 27005, стосуються управління інформаційною безпекою і включають рекомендації з впровадження криптографії для захисту даних у підприємствах. Ці стандарти допомагають створити політики і практики безпеки, які враховують використання криптографічних засобів для захисту конфіденційної інформації та зменшення ризиків, пов'язаних з її обробкою.

Робота з інформаційними системами, що включають використання шифрування для захисту даних, не тільки захищає інформацію, але й враховує фізичні та психологічні аспекти праці. Зокрема, з постійним використанням комп'ютерів і онлайн-систем зростає ризик виникнення захворювань, пов'язаних з тривалою роботою за екраном, таких як перенапруження очей, проблеми з поставою і інші. Забезпечення безпеки праці в таких умовах вимагає створення комфортних робочих умов, використання ергономічних меблів і регулярних перерв для профілактики фізичних навантажень.

Для забезпечення безпеки працівників, які працюють з чутливою інформацією, важливо впроваджувати не тільки фізичні, а й технічні заходи. Вони включають використання сучасних криптографічних протоколів, що гарантують захист даних від несанкціонованого доступу. Використання таких

протоколів, як SECURE SOCKETS LAYER/TRANSPORT LAYER SECURITY для захисту з'єднань між користувачами та вебсайтами, або алгоритмів симетричного та асиметричного шифрування, допомагає запобігти витоку або крадіжці інформації, що є важливим для захисту персональних даних і конфіденційної інформації.

Необхідно регулярно проводити аудит та тестування криптографічних систем для виявлення вразливостей та їх усунення. Також важливим є навчання працівників, які працюють з інформаційними системами, щодо правильного застосування шифрування і безпечної обробки даних. Оскільки шифрування є лише частиною загальної стратегії захисту інформації, необхідно розробляти комплексні програми безпеки, що враховують не тільки технічні, а й організаційні аспекти.

Таким чином, в контексті охорони праці важливо інтегрувати ефективне шифрування як частину стратегії забезпечення безпеки працівників і конфіденційної інформації. Правова база та міжнародні стандарти, що регулюють використання криптографії, дозволяють створювати надійні умови для захисту даних і підтримки здоров'я та безпеки працівників, що працюють з інформаційними системами.

4.2. Безпека в надзвичайних ситуаціях: Основні принципи і способи забезпечення життєдіяльності.

Безпека в надзвичайних ситуаціях є однією з найбільш важливих складових життєдіяльності людини, яка має на меті забезпечити здоров'я і життя у разі виникнення аварій, катастроф або стихійних лих. Надзвичайні ситуації можуть бути природними (землетруси, повені, урагани) або техногенними (аварії на виробництві, техногенні катастрофи), і кожна з них вимагає комплексних заходів для забезпечення безпеки і стабільності.

– Основні принципи забезпечення безпеки в надзвичайних ситуаціях

Забезпечення безпеки в надзвичайних ситуаціях ґрунтується на кількох основних принципах:

Принцип превентивних заходів — передбачає розробку та впровадження заходів, спрямованих на попередження виникнення надзвичайних ситуацій, таких як організація моніторингу навколишнього середовища, підготовка до кризових ситуацій та навчання персоналу.

Принцип ефективного реагування — включає в себе планування та координацію дій у разі виникнення надзвичайної ситуації для мінімізації шкоди, наприклад, створення екстрених груп швидкого реагування, організація евакуації.

Принцип захисту життєво важливих функцій — забезпечення стабільності і безпеки найважливіших інфраструктур, таких як водопостачання, електрична енергія, медична допомога та інформаційні системи.

Принцип післякатастрофного відновлення — після ліквідації наслідків надзвичайних ситуацій важливо здійснювати заходи щодо відновлення соціальної інфраструктури та підтримки нормальних умов життєдіяльності.

Основні способи забезпечення життєдіяльності в умовах надзвичайних ситуацій.

У разі виникнення надзвичайної ситуації необхідно використовувати комплексні способи для забезпечення життєдіяльності та безпеки:

- Евакуація — організація своєчасного і безпечного виведення людей з небезпечних зон. Це включає в себе знання евакуаційних шляхів, використання транспорту для переміщення людей і майна.
- Охорона і захист здоров'я — у разі хімічних, радіаційних або біологічних загроз необхідно використовувати засоби індивідуального захисту (Респіратори, захисні костюми) і організовувати надання медичної допомоги.
- Енергетичне забезпечення — у разі виходу з ладу основних джерел енергії слід забезпечити альтернативні джерела енергопостачання, наприклад, генератори або автономні електростанції.
- Інформаційна безпека — підтримка і захист систем комунікації та обміну інформацією між органами управління та громадськістю під час кризи є надзвичайно важливою для координації дій і забезпечення своєчасної інформації.

Підготовка до надзвичайних ситуацій:

Правильна підготовка до надзвичайних ситуацій є запорукою ефективного реагування і мінімізації їхніх наслідків. Основними заходами є:

Планування — розробка планів евакуації, аварійно-рятувальних робіт, медичних заходів.

Навчання персоналу та населення — проведення тренінгів і навчань для громадян та співробітників на випадок надзвичайних ситуацій.

Підготовка технічних засобів — забезпечення аварійних резервів ресурсів, таких як водопостачання, їжа, медикаменти, засоби зв'язку та транспорт.

Правова база і стандарти безпеки в надзвичайних ситуаціях

В Україні питання безпеки в надзвичайних ситуаціях регулюються такими нормативно-правовими актами:

Закон України "Про цивільний захист" — встановлює правові засади цивільного захисту населення в разі надзвичайних ситуацій, визначає обов'язки органів влади та громадян у забезпеченні безпеки.

Закон України "Про охорону праці" — вимагає від підприємств і організацій створення безпечних умов праці, що включає планування і підготовку до надзвичайних ситуацій, а також належне навчання працівників щодо захисту в таких ситуаціях.

ДСТУ 4506-88 "Системи управління безпекою праці та техногенними катастрофами" — стандарт, що встановлює вимоги щодо організації систем управління безпекою в умовах надзвичайних ситуацій.

ВИСНОВКИ

У процесі виконання магістерської кваліфікаційної роботи на тему "Вплив шифрування на продуктивність вебсайтів: порівняльний аналіз сучасних алгоритмів" було досягнуто основної мети – проаналізовано сучасні криптографічні алгоритми та їхній вплив на швидкодію вебресурсів, а також сформульовано рекомендації щодо вибору оптимального шифрування для малого та середнього бізнесу.

Основні результати роботи:

Досліджено сучасні підходи до шифрування у вебсередовищі Розглянуто основні типи алгоритмів (симетричні, асиметричні та гібридні), їхні особливості, переваги та недоліки. Виокремлено ключові аспекти використання шифрування у вебтехнологіях.

Проведено порівняльний аналіз популярних алгоритмів Виконано детальне дослідження алгоритмів AES, RSA, ECC, ChaCha20. Порівняння за критеріями швидкості, рівня безпеки, сумісності та навантаження на сервер дозволило визначити оптимальні сценарії їхнього застосування залежно від вимог вебсайту.

Оцінено вплив шифрування на продуктивність вебсайтів Встановлено, що використання асиметричних алгоритмів суттєво збільшує час встановлення SSL/TLS -з’єднань, тоді як симетричні алгоритми, зокрема AES та ChaCha20, демонструють найвищу швидкість обробки даних.

Розроблено рекомендації для малого та середнього бізнесу Запропоновано практичні рішення для вибору алгоритмів залежно від потреб підприємства, технічних можливостей та обмежень бюджету.

Практичне значення роботи:

Отримані результати можуть бути впроваджені у практичну діяльність компаній, які прагнуть забезпечити захист інформації без втрати швидкодії вебресурсів. Особливу увагу приділено сценаріям впровадження для малого та середнього бізнесу, що дозволяє оптимізувати витрати та гарантувати високий рівень безпеки.

Перспективи подальших досліджень:

– Дослідження впливу шифрування на продуктивність у специфічних умовах, таких як робота з IoT-пристроями або хмарними сервісами.

– Розробка адаптивних алгоритмів шифрування, що враховують поточне навантаження на сервери.

– Поглиблений аналіз енергоспоживання криптографічних алгоритмів, особливо в умовах обмежених ресурсів.

Таким чином, виконана робота зробила значний внесок у розуміння сучасних підходів до шифрування вебсайтів, їхніх можливостей та обмежень. Отримані результати є основою для подальшого вдосконалення систем захисту інформації у цифровому середовищі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Kulchytskyi, T., Rezvorovych, K., Povalena, M., Dutchak, S., & Kramar, R. (2024). LEGAL REGULATION OF CYBERSECURITY IN THE CONTEXT OF THE DIGITAL TRANSFORMATION OF UKRAINIAN SOCIETY. Lex Humana (ISSN 2175-0947), 16(1), 443-460.

2. Yesin, V., Karpinski, M., Yesina, M., Vilihura, V., Kozak, R., Shevchuk, R. (2023). Technique for Searching Data in a Cryptographically Protected SQL Database. Applied Sciences, 13(20), art. no. 11525, 1-21. doi: 10.3390/app132011525.

3. Kuznetsov, O., Poluyanenko, N., Frontoni, E., Kandiy, S., Karpinski, M., & Shevchuk, R. (2024). Enhancing Cryptographic Primitives through Dynamic Cost Function Optimization in Heuristic Search. Electronics, 13(10), 1-52. doi: 10.3390/electronics13101825.

4. Kuznetsov, A., Karpinski, M., Ziubina, R., Kandiy, S., Frontoni E., Veselska, O., & Kozak, R. (2023). Generation of nonlinear substitutions by simulated annealing algorithm. Information, 14(5), art. no. 259, 1-16. doi: 10.3390/info14050259.

5. Karpinski, M., Kovalchuk, L., Kochan, R., Oliynykov, R., Rodinko, M., & Wiecław, L. (2021). Blockchain technologies : probability of double-spend attack on a proof-of-stake consensus. Sensors, 21(19), 1-14. doi: 10.3390/s21196408.

6. Городецький, В. П., & Пономарьов, О. В. (2019). Основи криптографії. Київ: Наукова думка.

7. Lypa, B., Horyn, I., Zagorodna, N., Tymoshchuk, D., Lechachenko T., (2024). Comparison of feature extraction tools for network traffic data. CEUR Workshop Proceedings, 3896, pp. 1-11.

8. Петренко, М. С. (2021). Сучасні методи шифрування даних у мережах Інтернет. Журнал інформаційної безпеки, (4), 12–20.

9. Українська асоціація кібербезпеки. (2022). Рекомендації з впровадження SSL/TLS для малого та середнього бізнесу. Київ. Retrieved from https://uaciber.org.ua

10. Tymoshchuk, D., Yasniy, O., Mytnyk, M., Zagorodna, N., Tymoshchuk, V., (2024). Detection and classification of DDoS flooding attacks by machine learning methods. CEUR Workshop Proceedings, 3842, pp. 184 - 195.

11. Rivest, R., Shamir, A., & Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM, 21(2), 120–126.

12. ТИМОЩУК, Д., ЯЦКІВ, В., ТИМОЩУК, В., & ЯЦКІВ, Н. (2024). INTERACTIVE CYBERSECURITY TRAINING SYSTEM BASED ON SIMULATION ENVIRONMENTS. MEASURING AND COMPUTING DEVICES IN TECHNOLOGICAL PROCESSES, (4), 215-220.

13. Rescorla, E. (2000). HTTP over TLS (RFC 2818). Internet Engineering Task Force (IETF). Retrieved from https://www.rfc-editor.org/rfc/rfc2818.html

14. Bernstein, D. J. (n.d.). ChaCha, a variant of Salsa20. Retrieved from https://cr.yp.to/chacha.html

15. IEEE Standards Association. (2020). Elliptic Curve Cryptography Standards for Efficient Security.

16. NIST. (n.d.). AES Encryption Algorithm. National Institute of Standards and Technology. Retrieved from https://csrc.nist.gov/projects/aes

17. ТИМОЩУК, Д., & ЯЦКІВ, В. (2024). USING HYPERVISORS TO CREATE A CYBER POLYGON. MEASURING AND COMPUTING DEVICES IN TECHNOLOGICAL PROCESSES, (3), 52-56.

18. Microsoft. (n.d.). Рекомендації з налаштування SSL/TLS у Windows Server. Retrieved from https://docs.microsoft.com/uk-ua

19. Cloudflare. (n.d.). Порівняння сучасних алгоритмів шифрування: ECC, RSA та AES. Retrieved from https://www.cloudflare.com

20. Tymoshchuk, D., & Yatskiv, V. (2024). Slowloris ddos detection and prevention in real-time. Collection of scientific papers «ΛΌГОΣ», (August 16, 2024; Oxford, UK), 171-176.

21. GTmetrix. (n.d.). GTmetrix tools. Retrieved from https://gtmetrix.com

22. Apache Software Foundation. (n.d.). Apache JMeter user guide. Retrieved from https://jmeter.apache.org/usermanual/index.html

23. Google. (n.d.). Performance metrics. Retrieved from https://web.dev/metrics/

24. OpenSSL Project. (n.d.). OpenSSL documentation. Retrieved from https://www.openssl.org/docs/

25. HTTP/2 Working Group. (2015). Hypertext transfer protocol version 2 (HTTP/2). Retrieved from https://http2.github.io/

26. Bernstein, D. J., Lange, T., & Schwabe, P. (2012). ECC encryption guide. Retrieved from https://safecurves.cr.yp.to/

Додаток А Публікація

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
імені ІВАНА ПУЛЮЯ
НАУКОВЕ ТОВАРИСТВО ім. ШЕВЧЕНКА

М А Т Е Р І А Л И

XII НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ»

18-19 грудня 2024 року

ТЕРНОПІЛЬ
2024

УДК 001
М34

ПРОГРАМНИЙ КОМІТЕТ

Голова: Приймак Микола – професор кафедри комп'ютерних систем та мереж, д.т.н., професо**р.**
Співголови: Марущак Павло – проректор з наукової роботи, докт. техн. наук, професор.
Баран Ігор – канд. техн. наук, доцент, декан факультету ФІС.
Науковий секретар: Надія Крива– старший викладач.

Члени: Василь Кривень - завідувач кафедри математичних методів в інженерії д.ф.-м.н., професор; Галина Осухівська – завідувач кафедри комп'ютерних систем та мереж, к.т.н., доцент; Микола Карпінський - професор кафедри кібербезпеки, д.т.н., професор; Жанна Баб'як - завідувач кафедри української та іноземних мов, к.пед. н., доцент; Ярослав Литвиненко – професор кафедри комп'ютерних наук, д.т.н., професор; Михайло Петрик - завідувач кафедри програмної іженерії, д.ф.-м.н., професор; Наталія Загородна – завідувач кафедри кібербезпеки, к.т.н., доцент.

ОРГАНІЗАЦІЙНИЙ КОМІТЕТ

Голова: Скоренький Юрій Любомирович – канд. техн. наук, доцент кафедри фізики.
Члени: доцент кафедри комп'ютерних наук, к.т.н. В. Никитюк; доцент кафедри програмної інженерії, к.т.н. Д. Михалик; доцент кафедри кібербезпеки, к.т.н. М. Стадник; доцент кафедри комп'ютерних систем та мереж, к.т.н. Є. Тиш; ст. викладач Л. Джиджора.

Матеріали XII науково-технічної конфції «Інформаційні моделі, системи
М34 та технології» Тернопільського національного технічного університету імені Івана Пулюя, (Тернопіль, 18–19 грудня 2024 р.). – Тернопіль : Тернопільський національний технічний університет імені Івана Пулюя, 2024.

Адреса оргкомітету: ТНТУ ім. І. Пулюя, м. Тернопіль, вул. Руська, 56, 46001,
тел. (0352) 52-41-33, факс (0352) 25-49-83.
E-mail: conffis2024@gmail.com
Редагування, оформлення та верстка: Надія Крива

СЕКЦІЇ КОНФЕРЕНЦІЇ, ЯКІ ПРЕДСТВЛЕНІ В ЗБІРНИКУ

- Математичне моделювання
- Інформаційні системи та технології, кібербезпека
- Комп'ютерні системи та мережі
- Програмна інженерія та моделювання складних розподілених систем
- Новітні фізико-технічні та освітні технології

В збірнику надруковано тези доповідей XII науково-технічної конференції «Інформаційні моделі, системи та технології» (Тернопіль, 18–19 грудня 2024 р.) за такими науковими напрямками: математичне моделювання; інформаційні системи та технології, кібербезпека; комп'ютерні системи та мережі; програмна інженерія та моделювання складних розподілених систем; новітні фізико-технічні та освітні технології.

Розрахований на науковців, викладачів та студентів вузів.

За зміст тез та дотримання норм академічної доброчесності відповідальність несе автор.

УДК 004.4

В. Б. Гурський, Т.Р. Кульчицький

(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

Вплив шифрування на продуктивність вебсайтів: порівняльний аналіз сучасних алгоритмів

V.B. Hurskiy, T.R. Kulchytskyi

The impact of encryption on website performance: a comparative analysis of modern algorithms

1. Значення шифрування для сучасних вебсайтів

– **Забезпечення конфіденційності та безпеки даних**

Шифрування є основою для захисту особистої інформації користувачів, особливо при передаванні даних через Інтернет (наприклад, HTTPS). Воно мінімізує ризик перехоплення та злому даних.

– **Обов'язковість у сучасних стандартах**

Усі сучасні вебсайти мають використовувати HTTPS, що базується на TLS (Transport Layer Security), для відповідності вимогам пошукових систем і підвищення довіри користувачів.

– **Вплив на SEO**

Пошукові системи (наприклад, Google) надають перевагу сайтам із шифруванням, підвищуючи їх у рейтингу.

2. Типи алгоритмів шифрування, що застосовуються у вебсайтах

– **RSA (Rivest–Shamir–Adleman)**

– **Особливості:** Широко використовуваний алгоритм асиметричного шифрування для встановлення з'єднань.

– **Недоліки:** Високе навантаження на процесор через складність обчислень, особливо з великими ключами (2048/4096 біт).

– **Переваги:** Високий рівень захисту.

– **ECDSA (Elliptic Curve Digital Signature Algorithm)**

– **Особливості:** Менші ключі зберігають такий самий рівень безпеки, як і RSA, але працюють швидше.

– **Вплив на продуктивність:** Менше використання обчислювальних ресурсів.

– **AES (Advanced Encryption Standard)**

– **Особливості:** Симетричний алгоритм, що забезпечує швидке шифрування. Зазвичай використовується у HTTPS після встановлення з'єднання.

– **Переваги:** Енергоефективний, підходить для великих обсягів даних.

– **ChaCha20 + Poly1305**

– **Особливості:** Більш ефективний для пристроїв із низькою продуктивністю, таких як мобільні телефони.

– **Переваги:** Швидкий і сучасний, особливо для нестандартних умов з'єднання.

3. Як шифрування впливає на продуктивність вебсайтів

– **Затримки при встановленні з’єднання**
Алгоритми асиметричного шифрування (RSA, ECDSA) впливають на швидкість першого встановлення TLS-з’єднання. Чим складніший алгоритм і більший ключ, тим більше часу потрібно.

– **Ресурси сервера**
Використання складних алгоритмів шифрування збільшує навантаження на сервери. Це може бути критичним для сайтів із великим трафіком.

– **Затримки під час передачі даних**
Симетричні алгоритми, такі як AES, швидше шифрують дані під час передачі, але все одно додають невеликі накладні витрати.

–Оптимізація з'єднань

Використання HTTP/2 і TLS 1.3 значно зменшують накладні витрати шифрування, зменшуючи кількість "рукостискань" і оптимізуючи процес передачі.

4. Порівняльний аналіз продуктивності алгоритмів

–RSA vs ECDSA

RSA забезпечує високу безпеку, але його повільніша продуктивність робить його менш ефективним порівняно з ECDSA, особливо в мобільних мережах.

–AES vs ChaCha20

AES працює ефективніше на сучасних процесорах із підтримкою апаратного шифрування (AES-NI), тоді як ChaCha20 кращий для мобільних пристроїв і старіших процесорів.

–TLS 1.2 vs TLS 1.3

TLS 1.3 швидший за TLS 1.2 завдяки меншій кількості рукостискань і оптимізованому алгоритму шифрування.

5. Шляхи оптимізації використання шифрування

–Вибір правильного алгоритму для аудиторії сайту

Для мобільних користувачів варто використовувати ChaCha20, тоді як для стаціонарних серверів оптимальним є AES із апаратною підтримкою.

–Використання TLS 1.3

Це новий стандарт, який дозволяє значно зменшити затримки й підвищити продуктивність.

–Кешування та стиснення даних

Використання стиснення (наприклад, Brotli) зменшує обсяг переданих даних, а кешування TLS-з'єднань (Session Resumption) скорочує час повторного підключення.

6. Висновки

Шифрування є критично важливим компонентом сучасних вебсайтів, але воно має певний вплив на продуктивність. Правильний вибір алгоритму та технологій шифрування дозволяє забезпечити баланс між безпекою і швидкістю роботи вебсайту. У майбутньому, із розвитком обчислювальних технологій, продуктивність шифрування буде ще кращою.