МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНИЙ
ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ
ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

Звіт
з виконання завдань практичної роботи №1 на тему
«Вибір та реалізація базових фреймворків та бібліотек»
з кредитного модуля «Методи реалізації криптографічних механізмів»

Виконали:
студенти гр. ФБ-31мн
Журибіда Ю.Б.
Швець М.К.
Шостак А.А.

Київ 2024

Зміст

Вступ

Мета роботи: вибір базових бібліотек/сервісів для подальшої реалізації криптосистеми.

Ми будемо реалізовувати завдання для підгрупи 3А — розробка технічних вимог (із вибором або бібліотеки реалізації арифметичних операцій або бібліотеки реалізації основних криптографічних примітивів) для різних варіантів реалізацій Web-сервісу електронного цифрового підпису.

Теоретичні відомості

Для початку, невелика кількість теоретичних відомостей про сам електронний цифровий підпис. Електронний цифровий підпис (ЕЦП) — це криптографічний механізм, який використовується для забезпечення автентичності та цілісності електронних документів. ЕЦП гарантує, що документ не був змінений після підписання і підтверджує особу підписанта.

Основою для створення і перевірки ЕЦП є асиметричне шифрування, яке використовує пару ключів — приватний і публічний:

- приватний ключ використовується для створення підпису. Зберігається в таємниці і доступний лише власнику;
- публічний ключ, в свою чергу, використовується для перевірки підпису. Він може бути доступний будь-кому.

Розділ I. Загальні вимоги до Web-сервісу електронного цифрового підпису		
	1	2
1	Функціонал	Генерація криптографічних ключів для створення та перевірки цифрових підписів. Створення цифрових підписів для файлів різних форматів (PDF, TXT, XML, JSON). Перевірка валідності цифрового підпису. Експорт і імпорт ключів у загальноприйнятих форматах (PEM, DER). Опціональна підтримка сертифікатів для ідентифікації підписувача.
2	Протоколи та стандарти	Взаємодія з користувачами через захищений API (HTTPS із TLS 1.2 або вище). Використання широко підтримуваних стандартів підписів (PKCS#1, PKCS#7, PKCS#12). Можливість інтеграції з іншими сервісами через стандартні протоколи (REST, gRPC).
3	Безпека	Всі приватні ключі зберігаються у безпечному вигляді на сервері (наприклад, зашифровані за допомогою AES). Перевірка коректності вхідних даних (вхідні дані перевіряються для запобігання SQL-ін'єкціям, XSS та іншим атакам). Логування операцій без збереження конфіденційної інформації.
4	Масштабованість	Підтримка роботи у багатокористувацькому середовищі. Легкість інтеграції в хмарну інфраструктуру (наприклад, AWS, Azure, Google Cloud). Забезпечення продуктивності для виконання 1000+ підписів за хвилину.
Розділ II. Вимоги до криптографічної реалізації		
1	Арифметичні операції	Підтримка швидких і точних операцій з великими числами. Використання ефективних алгоритмів обчислення модульної арифметики (потрібно для RSA та інших алгоритмів). Підтримка апаратного прискорення криптографічних операцій.
2	Криптографічні алгоритми	Генерація ключів: RSA (2048, 3072, 4096 біт) для цифрових підписів. ECC (Elliptic Curve Cryptography) для більшої ефективності (P-256, P-384).

		Хешування: Підтримка SHA-256 як основного алгоритму. Опціональна підтримка SHA-1, SHA-384, SHA-512 для сумісності. Цифровий підпис: RSASSA-PSS як сучасний стандарт підпису. Підтримка RSASSA-PKCS1-v1_5 для сумісності зі старими системами. Шифрування: Використання симетричного шифрування для збереження даних (наприклад, AES-256).
3	Формати ключів	PEM: Основний формат для збереження ключів і сертифікатів. DER: Бінарний формат для сумісності з різними платформами. PKCS#12: Для зберігання ключів із сертифікатами.
Розділ III. Вимоги до обробки даних		
1	Формати файлів	Підтримка файлів довільного розміру, із завантаженням у пам'ять частинами для обробки великих документів. Робота з основними форматами документів (TXT, PDF, XML).
2	Кодування	Використання Base64 для передачі підписів і ключів через JSON. Підтримка Hex-кодування для альтернативної передачі даних.
3	Перевірка підпису	Валідація цифрового підпису із забезпеченням ідентифікації ключа (за допомогою сертифікатів або хешів ключа). Визначення причин недійсності підпису (недійсний ключ, підроблений документ тощо).
Розділ IV. Вимоги до API		
1	Ендпоінти сервісу	/generate_keys: Генерація приватного і публічного ключа. /sign_document: Підписання документа. /verify_signature: Перевірка цифрового підпису. /export_certificate: Експорт сертифікатів у PEM/DER.
2	Формат передачі даних	JSON для текстових даних і метаданих. Multipart для завантаження файлів.
3	Розширення API	Підтримка запитів з передачею ключів у зашифрованому вигляді (захист приватних даних). Функціонал для створення самопідписаних сертифікатів.
Розділ V. Вимоги до продуктивності		
1	Час виконання	Генерація ключів: не більше 200 мс для RSA 2048. Підписання документа розміром 1 МБ: до 100 мс. Перевірка підпису документа розміром 1 МБ: до 50 мс.
2	Оптимізація	Використання багатопотокової обробки. Можливість використання апаратного забезпечення (наприклад, HSM або CPU з AES-NI).

Найпоширенішими та найбільш уживаними бібліотеками, які можуть бути використані для реалізації Web-сервісу ЕЦП, та з якими учасники нашої бригади мали справу в межах своєї професійної діяльності — Libsodium, OpenSSL та Bouncy Castle.

Перш ніж проводити перевірку відповідності цих бібліотек технічним вимогам, розглянемо їх суть.

Libsodium - сучасна криптографічна бібліотека, створена для розробників, яким потрібен простий, ефективний і безпечний інструмент для реалізації криптографічних операцій. Розширення бібліотеки NaCl (Networking and Cryptography Library), створеної криптографом Даніелем Бернштейном, Libsodium спрощує реалізацію криптографічних примітивів з правильними налаштуваннями за замовчуванням і мінімізує ризик помилок під час використання.

Бібліотека підтримує найсучасніші алгоритми, такі як асиметрична криптографія (Ed25519 для цифрових підписів, Curve25519 для обміну ключами), симетрична криптографія (AES-GCM, ChaCha20-Poly1305), хешування (BLAKE2, SHA-256, SHA-512), функції розтягування ключів (Argon2, Scrypt).

Кросплатформенна Libsodium, яка підтримує Windows, Linux, macOS, Android та iOS, пропонує простий інтерфейс API, розроблений для полегшення інтеграції та захисту від побічних атак, таких як механізми атак під час виконання. Бібліотека широко використовується для обміну ключами, підписання даних, шифрування інформації, хешування паролів та генерації токенів автентифікації. Використання ChaCha20-Poly1305 забезпечує високу продуктивність, особливо на пристроях без апаратного прискорення AES. libsodium також надає безпечні налаштування за замовчуванням, які дозволяють використовувати криптографічні функції без глибоких знань у цій галузі.

Основними перевагами є сучасність, простота використання, висока продуктивність і безпечні налаштування за замовчуванням. Однак Libsodium не підтримує застарілі алгоритми, такі як RSA або SHA-1, і не має інструментів для роботи з форматами PKCS#7 або PKCS#12, що може вимагати використання зовнішніх бібліотек. Ця бібліотека ідеально підходить для нових проектів, де важлива безпека, сучасні алгоритми і простота інтеграції. Якщо вам потрібно працювати зі старими криптографічними стандартами або інтегруватися з застарілими системами, вам можуть знадобитися додаткові інструменти.

OpenSSL - одна з найпопулярніших і найпоширеніших криптографічних бібліотек, яка дозволяє реалізовувати криптографічні примітиви, безпечні протоколи і стандарти безпеки даних. Вона використовується для генерації ключів, створення та перевірки цифрових підписів, шифрування даних та забезпечення безпечних з'єднань за допомогою протоколів TLS та SSL.OpenSSL підтримує широкий спектр шифрів, включаючи RSA, DSA, ECC, AES, алгоритми ChaCha20-Poly1305, SHA-256 та SHA-512.

Однією з головних переваг OpenSSL є те, що вона підтримує класичні криптографічні стандарти, такі як PKCS#1, PKCS#7 та PKCS#12. Бібліотека може зберігати ключі у форматах PEM, DER та PKCS#12, а також обробляти цифрові сертифікати. OpenSSL є високопродуктивною і підтримує апаратне прискорення, таке як AES-NI, що робить її ефективною для використання на високонавантажених системах. Вона також підтримує останні версії TLS (1.2 і 1.3), забезпечуючи високу безпеку мережевих з'єднань.

Бібліотека є кросплатформенною і підтримується на більшості операційних систем, включаючи Windows, Linux і macOS. openSSL надає гнучкий і потужний інтерфейс для розробників, але його використання може бути складним у налаштуванні і вимагає хорошого розуміння криптографії.

OpenSSL надає багатофункціональну консольну утиліту для виконання криптографічних операцій, яку багато адміністраторів використовують для генерації ключів і сертифікатів, створення підписів і встановлення безпечних з'єднань.

Основними перевагами OpenSSL є широка підтримка класичних і сучасних алгоритмів, висока продуктивність, сумісність зі стандартами та підтримка апаратного прискорення. Однак низькорівневий API робить його складним у використанні і важким для початківців. OpenSSL ідеально підходить для проектів, які потребують підтримки класичних стандартів, високої продуктивності і можливості інтеграції з існуючими системами, що працюють на основі сертифікатів або застарілих алгоритмів.

Bouncy Castle багатофункціональна криптографічна бібліотека, розроблена для широкого спектру мов програмування, включаючи Java і C#. Вона підтримує інструменти для реалізації криптографічних примітивів, алгоритмів, стандартів і форматів, а також генерації ключів, цифрових підписів, шифрування, управління сертифікатами та інших

криптографічних операцій. Завдяки своїй універсальності та відкритій ліцензії бібліотека широко використовується в системах від серверів до мобільних додатків. Вона підтримує асиметричні криптографічні алгоритми, такі як RSA і ECC, симетричні шифри, такі як AES і Triple DES, SHA-256, SHA-512, BLAKE2 та інші хеш-алгоритми; забезпечує сумісність з криптографічними стандартами, такими як PKCS#1, PKCS#7 і PKCS#12, а також з форматами ключів і сертифікатів PEM і DER. Однією з особливостей цієї бібліотеки є можливість роботи з криптографічними об'єктами на Java, що робить її основним вибором для розробників цієї платформи.

Бібліотека пропонує високий рівень безпеки завдяки гнучкості в налаштуванні криптографічних операцій, але продуктивність Bouncy Castle може бути нижчою, ніж у бібліотек, оптимізованих для конкретних платформ, таких як OpenSSL. Залежно від використання, продуктивність RSA може бути повільною, оскільки вона не оптимізована для апаратного прискорення.Bouncy Castle має широкі можливості обробки цифрових підписів, включаючи алгоритми RSASSA-PSS і RSASSA-PKCS1-v1_5 Він має широку функціональність і підтримує криптографічні протоколи, такі як CMS (Cryptographic Message Syntax). Він є кросплатформенним і може бути легко інтегрований в різні системи, включаючи сервери і мобільні додатки.

Основними перевагами Bouncy Castle є широкий спектр функцій, універсальність, підтримка багатьох стандартів і інтеграція з мовою Java. У той же час, його недоліками є складність завдання для новачків і відносно низька продуктивність у порівнянні з бібліотеками, оптимізованими для апаратного прискорення. Bouncy Castle пропонує гнучку криптографічну реалізацію, сумісність зі стандартами і багатоплатформне середовище (особливо ідеально підходить для проектів, які вимагають роботи на Java-платформах).

Оформлення результатів

Створимо таблицю відповідності бібліотек Libsodium, OpenSSL та Bouncy Castle для реалізації нашого Web-сервісу ЕЦП:

Вимога	Libsodium	OpenSSL	Bouncy Castle
Функціонал			
Генерація ключів RSA та ECC	Часткова підтримка (без RSA)	Повна підтримка	Повна підтримка
Хешування (SHA-256, SHA-512)	Повна підтримка	Повна підтримка	Повна підтримка
Хешування (SHA-256, SHA-512)	Часткова підтримка (тільки Ed25519)	Повна підтримка	Повна підтримка
Робота з форматами ключів (PEM, DER)	Обмежена підтримка	Повна підтримка	Повна підтримка
Підтримка PKCS#7, PKCS#12	Немає	Повна підтримка	Повна підтримка
Протоколи та стандарти			
HTTPS із TLS 1.2/1.3	Можлива через зовнішні бібліотеки	Повна підтримка	Підтримка через сторонні засоби
Інтеграція через REST, gRPC	Потребує додаткової реалізації	Потребує додаткової реалізації	Потребує додаткової реалізації
Безпека			
Захищене зберігання ключів	Повна підтримка	Повна підтримка	Залежить від реалізації
Захист від атак (SQL-ін'єкції, XSS)	Залежить від реалізації	Залежить від реалізації	Залежить від реалізації
Масштабованість			

Робота у багатокористувацькому середовищі	Повна підтримка	Повна підтримка	Повна підтримка
Інтеграція в хмарну інфраструктуру	Повна підтримка	Повна підтримка	Повна підтримка
Продуктивність для 1000+ підписів/хв.	Найвища на сучасних алгоритмах	Висока (з апаратним прискоренням)	Середня
Криптографічні алгоритми			
RSA (2048, 3072, 4096 біт)	Немає	Повна підтримка	Повна підтримка
ECC (P-256, P-384)	Повна підтримка	Повна підтримка	Повна підтримка
RSASSA-PSS	Немає	Повна підтримка	Повна підтримка
AES-256	Повна підтримка	Повна підтримка	Повна підтримка
Формати ключів			
PEM, DER	Обмежена підтримка	Повна підтримка	Повна підтримка
PKCS#12	Немає	Повна підтримка	Повна підтримка
Продуктивність			
Генерація RSA 2048	Немає	~200 мс	~300 мс
Підписання документа (1 МБ)	Найвища (на Ed25519)	Висока	Середня
Перевірка підпису (1 МБ)	Найвища (на Ed25519)	Висока	Середня

Висновки

Виходячи з таблиці відповідності, OpenSSL є найбільш універсальною бібліотекою, що повністю відповідає всім технічним вимогам. Вона забезпечує підтримку класичних криптографічних алгоритмів і форматів, високу продуктивність із можливістю використання апаратного прискорення, а також відповідає стандартам і протоколам, необхідним для реалізації Web-сервісу електронного цифрового підпису.

Libsodium чудово підходить для нових систем, які використовують сучасні алгоритми, такі як Ed25519 і ChaCha20-Poly1305, але вона не підтримує класичні алгоритми, як RSA, і не працює з форматами PKCS#7 чи PKCS#12, що робить її непридатною для описаних вимог.

Bouncy Castle добре підходить для проєктів на Java чи C#, але поступається OpenSSL у продуктивності та простоті використання для реалізації сучасних API.

Тому ми вирішили використовувати OpenSSL для подальшої реалізації Web-сервісу електронного цифрового підпису.