МЕТОДИ РЕАЛІЗАЦІЇ КРИПТОГРАФІЧНИХ МЕХАНІЗМІВ ЛАБОРАТОРНА РОБОТА №4

“Дослідження особливостей реалізації існуючих програмних систем, які використовують криптографічні механізми захисту інформації”.

Недождій Максим, Буржимський Ростислав

ФІ-42мн

1 Мета роботи

Отримання практичних навичок побудови гібридних криптосистем.

2 Постановка задачі

Підгрупа 1А. Реалізація для інтелектуальної картки, токена.

3 Хід роботи

3.1 Сутність об'єктів дослідження

Смарт-картки та токени є ключовими інструментами для реалізації сучасних криптографічних систем, які забезпечують захист даних і управління ключами.

- Смарт-картки – це пристрої з контактним або безконтактним інтерфейсом, що використовуються для зберігання ключової інформації, авторизації користувачів, створення кваліфікованого електронного підпису та інших операцій у системах захисту даних. Вони часто виконані у формі банківської картки і здатні генерувати власні криптографічні ключі, мінімізуючи втручання зовнішніх пристроїв.
- Смарт-токени, зі свого боку, є носіями ключової інформації у формі USB-пристрою, що поєднує компактність із можливістю виконання криптографічних функцій. Як приклад, токени здатні зберігати ключі, виконувати їх ротацію і підтримувати функції криптографічного захисту.

Програмна реалізація криптосистем на базі цих пристроїв є складним процесом, що супроводжується низкою викликів:

- Забезпечення безпеки ключової інформації. Основною задачею є надійний захист ключів від несанкціонованого доступу та витоків. Це включає захист ключів в оперативній пам'яті, безпечне зберігання ключів на дисках або у зовнішніх пристроях, а також захист від атак через побічні канали (наприклад, часові атаки чи атаки за споживанням енергії).
- Забезпечення правильності функціонування програми. Реалізація криптографічних алгоритмів має відповідати міжнародним стандартам (наприклад, FIPS, ISO/IEC). Для цього необхідна формальна верифікація алгоритмів, постійне тестування, а також регулярне оновлення програмного забезпечення з метою усунення можливих вразливостей.
- Оптимізація продуктивності. Оскільки криптографічні операції є обчислювально складними, важливо забезпечити їх ефективність за допомогою апаратних прискорювачів (наприклад, AES-NI для прискорення AES), сучасних бібліотек та оптимізації обчислень.

- Сумісність і масштабованість. Криптосистеми мають бути інтегрованими у вже існуючі інфраструктури та підтримувати роботу на різних платформах, таких як операційні системи та архітектури, через стандартні інтерфейси (наприклад, CryptoAPI, PKCS#11).
- Управління ключами. Ефективне управління включає генерацію, розподіл, ротацію ключів, а також розробку політик доступу до них. Особливо важливо забезпечити захищений обмін ключовою інформацією.
- Захист від атак. Необхідно враховувати сучасні методи атак, включаючи атаки на основі побічних каналів, атаки повторного використання повідомлень і атаки, спрямовані на апаратні та програмні вразливості.

Таким чином, смарт-картки та токени, поєднані з відповідним програмним забезпеченням, можуть забезпечити високий рівень безпеки й ефективності криптографічних систем. Однак для їхньої успішної реалізації необхідно вирішити низку задач, пов'язаних із захистом, продуктивністю, сумісністю, а також управлінням ключовою інформацією. Це дозволить створити системи, які відповідатимуть сучасним вимогам інформаційної безпеки.

3.2 Порівняння інтерфейсів Microsoft Crypto API та PKCS#11

3.2.1 Microsoft Crypto API

Microsoft Crypto API – це програмний інтерфейс прикладних програм (API), що надає розробникам Windows-додатків стандартний набір функцій для роботи з криптографією. Вперше представлений компанією Microsoft, він інтегрований в операційні системи Windows і доступний для використання з простору користувача. Архітектура Crypto API побудована на концепції Cryptographic Service Providers (CSP), які виступають незалежними модулями для забезпечення різних криптографічних операцій.

Основні функції:

- Шифрування та розшифровування даних: забезпечує захист конфіденційної інформації.
- Генерація та перевірка цифрових підписів: використовується для автентифікації даних.

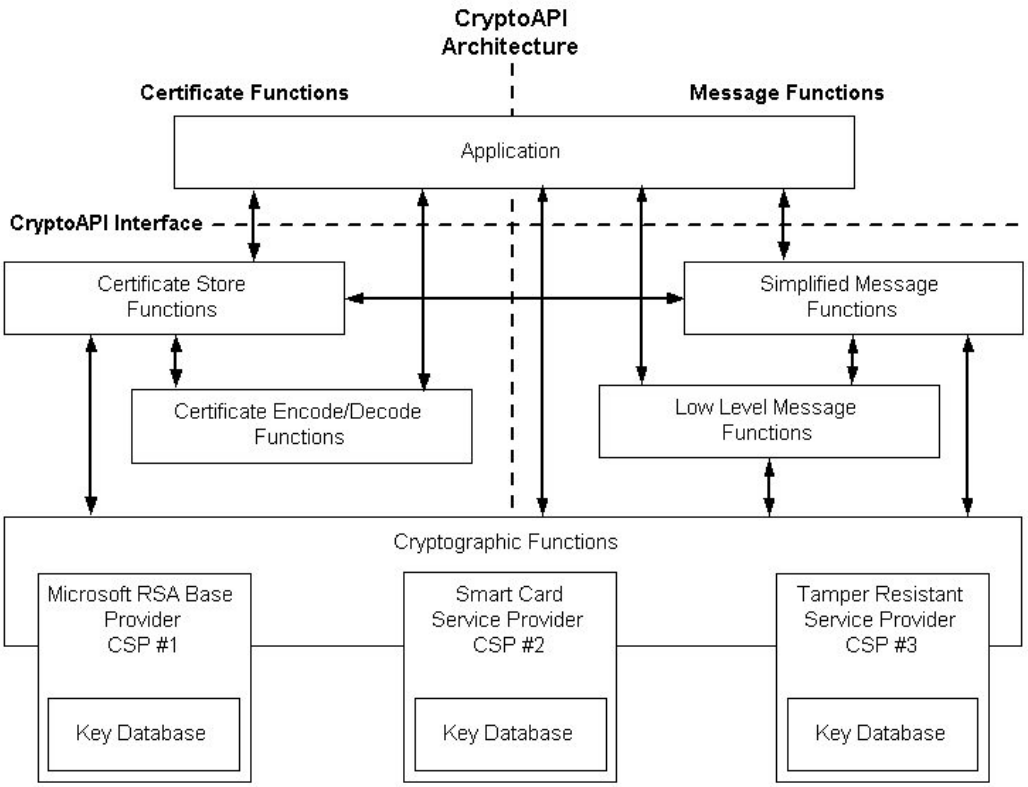

Figure 1: Архітектура CryptoAPI

- Алгоритми хешування: гарантують цілісність і одностороннє збереження даних.
- Захист від атак: реалізовано через вбудовані механізми безпеки.

Особливості:

- Контроль доступу до ключів: Crypto API інтегрується з Windows Data Protection API (DPAPI), що дозволяє обмежити доступ до ключів лише авторизованим користувачам.
- Підтримка інтелектуальних карток і токенів: забезпечується через Microsoft Smart Card API, що дозволяє працювати із зовнішніми пристроями.
- Простота інтеграції: інтерфейс добре сумісний з іншими компонентами Windows, спрощуючи розробку.
- Підтримка Linux: у Linux Crypto API виступає як криптографічна структура ядра для таких задач, як IPsec або dm-crypt. Хоча основною метою є підтримка внутрішніх процесів ядра, він також доступний для користувачів.

3.2.2 PKCS#11 (Cryptoki)

PKCS#11 – це стандарт програмного інтерфейсу для криптографічних токенів, створений компанією RSA Security. Вперше опублікований у 1995 році, він є незалежним від платформи API, що використовується для взаємодії з апаратними модулями безпеки (HSM), смарт-картками та криптографічними токенами. Основні функції:

- Шифрування та розшифровування даних: реалізує захист інформації.
- Генерація та перевірка цифрових підписів: забезпечує автентифікацію та цілісність даних.
- Ідентифікація і аутентифікація криптографічного модуля: підтвердження автентичності пристрою.
- Управління ключами: створення, зберігання, оновлення та видалення ключів.
- Виконання інших криптографічних функцій: наприклад, створення сертифікатів

Особливості:

- Контроль доступу до ключів: PKCS#11 дозволяє працювати з ключами, зберігаючи їх в ізольованому середовищі (наприлад, у HSM чи токенах), що мінімізує ризик їх витоку.
- Правильність функціонування: стандартизовані механізми забезпечують стабільну та безпечну роботу.
- Підтримка апаратних пристроїв: спеціально розроблений для взаємодії з апаратними криптографічними маркерами.
- Кросплатформеність: PKCS#11 підтримується на різних операційних системах, що робить його універсальним.

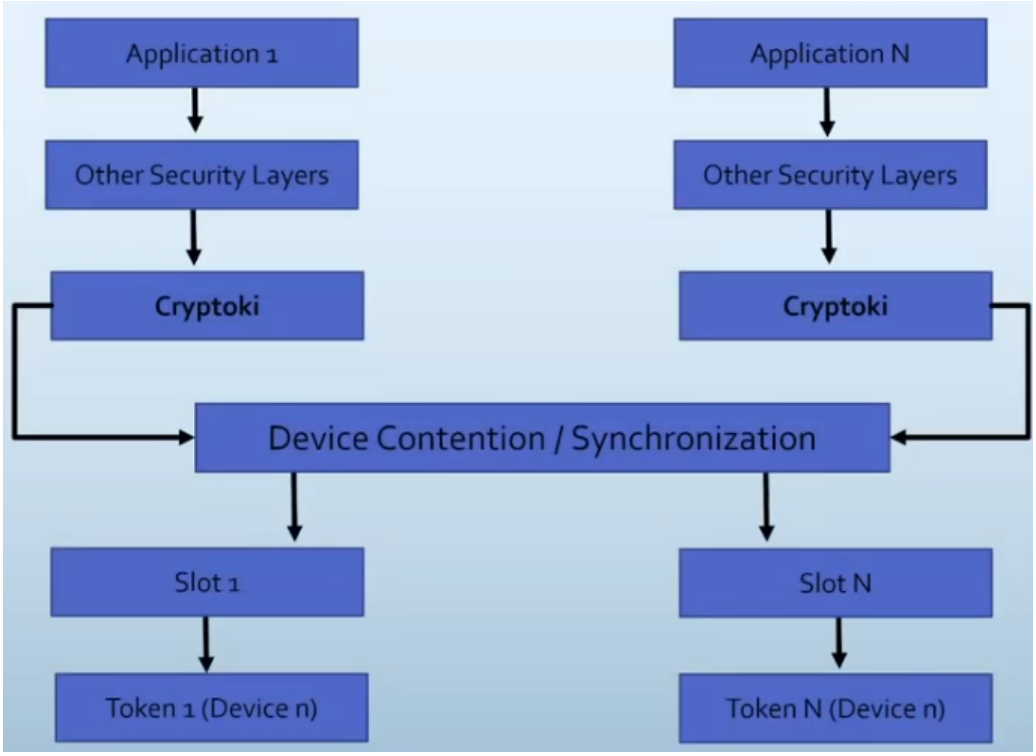

Figure 2: Архітектура PKCS

3.3 Таблиця порівняння деяких параметрів Microsoft Crypto API та PKCS#11

Параметр	**Microsoft Crypto API**	**PKCS#11**
Рік створення	Crypto API – для Windows з 1990-х років, Linux – 2002 рік	1995 рік
Контроль доступу до ключів	Інтеграція з DPAPI, доступ лише авторизованим користувачам	Ізольоване зберігання в апаратних пристроях
Підтримка платформи	Windows, Linux	Кросплатформений стандарт
Підтримка апаратних пристроїв	Через Smart Card API для Windows	Призначений для роботи з HSM, токенами, смарт-картами
Простота інтеграції	Добре інтегрується з екосистемою Windows	Універсальний, але вимагає більше налаштувань
Механізми безпеки	CSP, перевірка цілісності даних	Відповідність стандартам безпеки, наприклад, FIPS
Основне використання	Робота з додатками Windows, мережева безпека в Linux	Управління криптографічними токенами та HSM

4 Висновок

Crypto API є оптимальним вибором для розробників Windows-додатків завдяки глибокій інтеграції в екосистему Microsoft та простоті використання. PKCS#11 підходить для складних криптографічних операцій з акцентом на апаратну безпеку і є універсальним рішенням для роботи на різних платформах.