НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ

Кафедра Інформаційної Безпеки

Лабораторна робота №1

з дисципліни

«МЕТОДИ РЕАЛІЗАЦІЇ КРИПТОГРАФІЧНИХ МЕХАНІЗМІВ»

Виконали:

Студени 6 курсу ФТІ

групи ФБ-41мн

Бондаренко О.Ю., Кригін Д.О.

Перевірила:

асистент

Байденко П.В.

Київ 2025

Лабораторна робота №1
Вибір та реалізація базових фреймворків та бібліотек

Мета: Вибір базових бібліотек/сервісів для подальшої реалізації криптосистеми

Завдання: Другий тип лабораторної роботи. Підгрупа 2B. Порівняння бібліотек OpenSSL, Crypto++, CryptoLib, PyCrypto для розробки гібрідної криптосистеми під Linux платформу.

Хід Роботи

Алгоритми для дослідження:

- Симетричні: AES-256 CBC (шифрування, розшифрування)
- Асиметричні: RSA-2048 (генерація ключів, шифрування,
- Гешування: SHA-256

Конфігурація Операційної Системи для проведення вимірів:

--- CPU Details ---

Architecture:	x86_64
CPU(s):	4
Model name:	AMD Ryzen 7 6800H with Radeon Graphics
Core(s) per socket:	1

--- Memory Details ---

Mem:	16Gi

OpenSSL

Встановлення бібліотеки на систему

$ sudo apt update -y
$ sudo apt install -y libssl-dev

Компіляція бенчмарків

$ cd openssl
$ make clean_build

Генерація тестових даних

$ dd if=/dev/urandom of=testfile_1gb.bin bs=1M count=1024 oflag=sync status=progress

Запуск бенчмарків

$ for i in {1..10}; do echo "[*] Execution ${i}/10"; ./aes-256-cbc-benchmark.elf ./testfile_1gb.bin; done

```
user@debian:~/Labs/MRKM/Lab1/mrkm-2025-main/lab1/openssl$ for i in {1..10}; do echo "[*] Execution ${i}/10"; ./aes-256-cbc-benchmark.elf ./testfile_1gb.bin;
[*] Execution 1/10
[*] Benchmarking AES-256-CBC Encryption
[*] Benchmark report
[*] Execution time: 2.917847 seconds
[*] Memory used: 4912 KB -> 4912 KB (delta 0 KB)

[*] Benchmarking AES-256-CBC Decryption
[*] Benchmark report
[*] Execution time: 2.236130 seconds
[*] Memory used: 5324 KB -> 5324 KB (delta 0 KB)
[*] Execution 2/10
[*] Benchmarking AES-256-CBC Encryption
[*] Benchmark report
[*] Execution time: 2.880053 seconds
[*] Memory used: 4912 KB -> 4912 KB (delta 0 KB)

[*] Benchmarking AES-256-CBC Decryption
[*] Benchmark report
[*] Execution time: 2.116443 seconds
[*] Memory used: 5324 KB -> 5324 KB (delta 0 KB)
[*] Execution 3/10
[*] Benchmarking AES-256-CBC Encryption
[*] Benchmark report
[*] Execution time: 2.912818 seconds
[*] Memory used: 4612 KB -> 4612 KB (delta 0 KB)

[*] Benchmarking AES-256-CBC Decryption
[*] Benchmark report
[*] Execution time: 2.118158 seconds
[*] Memory used: 5324 KB -> 5324 KB (delta 0 KB)
[*] Execution 4/10
[*] Benchmarking AES-256-CBC Encryption
[*] Benchmark report
[*] Execution time: 2.908382 seconds
[*] Memory used: 4912 KB -> 4912 KB (delta 0 KB)

[*] Benchmarking AES-256-CBC Decryption
[*] Benchmark report
[*] Execution time: 2.147648 seconds
[*] Memory used: 5324 KB -> 5324 KB (delta 0 KB)
[*] Execution 5/10
[*] Benchmarking AES-256-CBC Encryption
[*] Benchmark report
[*] Execution time: 2.856443 seconds
[*] Memory used: 4912 KB -> 4912 KB (delta 0 KB)
```

$ for i in {1..10}; do echo "[*] Execution ${i}/10"; ./sha-256-benchmark.elf ./testfile_1gb.bin; done

```
user@debian:~/Labs/MRKM/Lab1/mrkm-2025-main/lab1/openssl$ for i in {1..10}; do echo "[*] Execution ${i}/10"; ./sha-256-benchmark.elf ./testfile_1gb.bin; done
[*] Execution 1/10
[*] Benchmarking SHA-256 hashing
[*] Benchmark report
[*] Execution time: 0.836877 seconds
[*] Memory used: 4844 KB -> 4844 KB (delta 0 KB)
[*] Execution 2/10
[*] Benchmarking SHA-256 hashing
[*] Benchmark report
[*] Execution time: 0.864260 seconds
[*] Memory used: 4832 KB -> 4832 KB (delta 0 KB)
[*] Execution 3/10
[*] Benchmarking SHA-256 hashing
[*] Benchmark report
[*] Execution time: 0.842496 seconds
[*] Memory used: 4844 KB -> 4844 KB (delta 0 KB)
[*] Execution 4/10
[*] Benchmarking SHA-256 hashing
[*] Benchmark report
[*] Execution time: 0.855669 seconds
[*] Memory used: 4844 KB -> 4844 KB (delta 0 KB)
[*] Execution 5/10
[*] Benchmarking SHA-256 hashing
[*] Benchmark report
[*] Execution time: 0.850523 seconds
[*] Memory used: 4840 KB -> 4840 KB (delta 0 KB)
[*] Execution 6/10
[*] Benchmarking SHA-256 hashing
[*] Benchmark report
[*] Execution time: 0.858863 seconds
[*] Memory used: 4844 KB -> 4844 KB (delta 0 KB)
[*] Execution 7/10
[*] Benchmarking SHA-256 hashing
[*] Benchmark report
[*] Execution time: 0.858894 seconds
[*] Memory used: 4840 KB -> 4840 KB (delta 0 KB)
```

$ for i in {1..10}; do echo "[*] Execution ${i}/10"; ./rsa-benchmark.elf; done

```
user@debian:~/Labs/MRKM/Lab1/mrkm-2025-main/lab1/openssl$ for i in {1..10}; do echo "[*] Execution ${i}/10"; ./rsa-benchmark.elf; done
[*] Execution 1/10
[*] Benchmarking RSA-2048 Key Generation
[*] Benchmark report
[*] Execution time: 0.095704 seconds
[*] Memory used: 4612 KB -> 4612 KB (delta 0 KB)

[*] Benchmarking RSA-2048 Encryption
[*] Benchmark report
[*] Execution time: 0.000060 seconds
[*] Memory used: 5488 KB -> 5488 KB (delta 0 KB)

[*] Benchmarking RSA-2048 Decryption
[*] Benchmark report
[*] Execution time: 0.000820 seconds
[*] Memory used: 5488 KB -> 5488 KB (delta 0 KB)
[*] Execution 2/10
[*] Benchmarking RSA-2048 Key Generation
[*] Benchmark report
[*] Execution time: 0.182748 seconds
[*] Memory used: 4900 KB -> 4900 KB (delta 0 KB)

[*] Benchmarking RSA-2048 Encryption
[*] Benchmark report
[*] Execution time: 0.000055 seconds
[*] Memory used: 4900 KB -> 4900 KB (delta 0 KB)

[*] Benchmarking RSA-2048 Decryption
[*] Benchmark report
[*] Execution time: 0.000778 seconds
[*] Memory used: 4900 KB -> 4900 KB (delta 0 KB)
[*] Execution 3/10
[*] Benchmarking RSA-2048 Key Generation
[*] Benchmark report
[*] Execution time: 0.123823 seconds
[*] Memory used: 4900 KB -> 4900 KB (delta 0 KB)

[*] Benchmarking RSA-2048 Encryption
[*] Benchmark report
[*] Execution time: 0.000085 seconds
[*] Memory used: 4900 KB -> 4900 KB (delta 0 KB)

[*] Benchmarking RSA-2048 Decryption
[*] Benchmark report
[*] Execution time: 0.000935 seconds
[*] Memory used: 4900 KB -> 4900 KB (delta 0 KB)
[*] Execution 4/10
[*] Benchmarking RSA-2048 Key Generation
[*] Benchmark report
```

Crypto++

https://www.cryptopp.com/#download

Будуємо бібліотеку

$ sudo apt install build-essential
$ wget https://www.cryptopp.com/cryptopp890.zip
$ unzip cryptopp890.zip
$ cd cryptopp890
$ make
$ sudo make install

Зкомпілюємо наш код

$ gcc -c benchmark.c -o benchmark.o
$ g++ -std=c++17 cryptopp_benchmark.cpp benchmark.o -o cryptopp_benchmark -lcryptopp -lpthread

Запуск бенчмарків

```
(.venv) user@debian:~/Labs/MRKM/Lab1$ ./cryptopp_benchmark test_file_1GB
--- Crypto++ Comprehensive Benchmark ---
Input File: test_file_1GB
File Size: 1024.00 MB
Number of Runs: 10
-------------------------------------------------------------------------------------------------
| Run | AES Enc (s) | AES Dec (s) | RSA Gen (s) | RSA Enc (s) | RSA Dec (s) | SHA-256 (s) | Memory Delta (KB) |
=================================================================================================
|   1 |      4.4223 |      0.5562 |    0.008252 |    0.000047 |    0.000790 |      1.1511 |              2632 |
|   2 |      4.7002 |      0.5757 |    0.004706 |    0.000041 |    0.000823 |      1.1274 |                 0 |
|   3 |      5.1624 |      0.5592 |    0.018957 |    0.000050 |    0.000952 |      1.1421 |                24 |
|   4 |      5.0848 |      0.5570 |    0.023859 |    0.000049 |    0.000838 |      1.1554 |                 0 |
|   5 |      5.0557 |      0.5729 |    0.005119 |    0.000047 |    0.000857 |      1.1665 |                 0 |
|   6 |      5.0652 |      0.5640 |    0.073062 |    0.000054 |    0.000840 |      1.1368 |                 0 |
|   7 |      4.8653 |      0.5606 |    0.014650 |    0.000050 |    0.000846 |      1.1989 |                 0 |
|   8 |      5.1416 |      0.5686 |    0.088061 |    0.000067 |    0.000842 |      1.1630 |                 0 |
|   9 |      5.2241 |      0.5768 |    0.012117 |    0.000048 |    0.000884 |      1.1664 |                 4 |
|  10 |      5.0884 |      0.5580 |    0.055304 |    0.000051 |    0.000850 |      1.1807 |                 0 |
=================================================================================================

============================================================
### FINAL AVERAGE TIME RESULTS (10 Runs) ###
============================================================
**1. Symmetric Encryption (AES-256 CBC) on 1GB**
   Average Encryption Time: **4.9810 seconds**
   Average Decryption Time: **0.5649 seconds**
------------------------------------------------------------
**2. Asymmetric Encryption (RSA-2048) on 32 Bytes (Key Exchange)**
   Average Key Generation Time: **0.030409 seconds**
   Average Encryption Time:   **0.000050 seconds**
   Average Decryption Time:   **0.000852 seconds**
------------------------------------------------------------
**3. Hashing (SHA-256) on 1GB**
   Average Hashing Time: **1.1588 seconds**
============================================================
```

PyCrypto

Налаштуємо середовище

$ python3 -m venv .venv
$ source .venv/bin/activate
$ pip install pycryptodome, memory-profiler

Створимо тестовий файл

$ dd if=/dev/urandom of=testfile_1gb.bin bs=1M count=1024 oflag=sync status=progress

Запуск бенчмарків

```
(.venv) user@debian:~/Labs/MRKM/Lab1$ python3 pycryptodome_benchmark.py test_file_1GB
--- PyCryptodome AES|RSA|SHA256 Benchmark ---
Input File: test_file_1GB
File Size: 1024.00 MB
Number of Runs: 10
------------------------------------------------------------
Run 1/10: AES Enc=3.5553s, Dec=1.9363s
Run 1/10: RSA Gen=0.310679s, Enc=0.000583s, Dec=0.001188s
Run 1/10: SHA-256 Hash=3.1973s
Run 2/10: AES Enc=3.6131s, Dec=1.9667s
Run 2/10: RSA Gen=0.322859s, Enc=0.000463s, Dec=0.001065s
Run 2/10: SHA-256 Hash=3.1559s
Run 3/10: AES Enc=3.8638s, Dec=1.9798s
Run 3/10: RSA Gen=0.388931s, Enc=0.000447s, Dec=0.001136s
Run 3/10: SHA-256 Hash=3.1449s
Run 4/10: AES Enc=3.5814s, Dec=1.9774s
Run 4/10: RSA Gen=0.335931s, Enc=0.000410s, Dec=0.001009s
Run 4/10: SHA-256 Hash=3.1574s
Run 5/10: AES Enc=3.5562s, Dec=1.9517s
Run 5/10: RSA Gen=1.061509s, Enc=0.000433s, Dec=0.001035s
Run 5/10: SHA-256 Hash=3.1501s
Run 6/10: AES Enc=3.7570s, Dec=1.9911s
Run 6/10: RSA Gen=0.300045s, Enc=0.000430s, Dec=0.001048s
Run 6/10: SHA-256 Hash=3.1228s
Run 7/10: AES Enc=3.7293s, Dec=1.9906s
Run 7/10: RSA Gen=0.110820s, Enc=0.000459s, Dec=0.001088s
Run 7/10: SHA-256 Hash=3.1943s
Run 8/10: AES Enc=3.5802s, Dec=2.0083s
```

```
Run 8/10: RSA Gen=0.419396s, Enc=0.000440s, Dec=0.001074s
Run 8/10: SHA-256 Hash=3.3421s
Run 9/10: AES Enc=4.1337s, Dec=2.0618s
Run 9/10: RSA Gen=0.286245s, Enc=0.000437s, Dec=0.001069s
Run 9/10: SHA-256 Hash=3.3286s
Run 10/10: AES Enc=3.8005s, Dec=2.2023s
Run 10/10: RSA Gen=0.794303s, Enc=0.000967s, Dec=0.001380s
Run 10/10: SHA-256 Hash=3.3319s

============================================================
### FINAL AVERAGE TIME RESULTS (10 Runs) ###
============================================================
**1. Symmetric Encryption (AES-256 CBC) on 1GB**
   Average Encryption Time: **3.7171 seconds**
   Average Decryption Time: **2.0066 seconds**
------------------------------------------------------------
**2. Asymmetric Encryption (RSA-2048) on 32 Bytes (Key Exchange)**
   Average Key Generation Time: **0.433072 seconds**
   Average Encryption Time:   **0.000507 seconds**
   Average Decryption Time:   **0.001109 seconds**
------------------------------------------------------------
**3. Hashing (SHA-256) on 1GB**
   Average Hashing Time: **3.2125 seconds**
============================================================
```

Таблиця Результатів

AES і SHA-256 були запущені на файлі розміру 1GB, RSA був запущений на 32-byte даних - умовному сесійному ключу.

	AES Encrypt	AES Decrypt	RSA Keygen	RSA Encrypt	RSA Decrypt	SHA-256
OpenSSL	2.7962	2.056	0.1755	0.000068	0.000824	0.8509
Crypto++	4.9810	0.5649	0.0304	0.0001	0.0009	1.1588
Pycrypto	3.7171	2.0066	0.4331	0.0005	0.0011	3.2125

Висновки

У ході лабораторної роботи було проведено порівняльний аналіз продуктивності трьох криптографічних бібліотек: **OpenSSL**, **Crypto++** та **PyCrypto** (**PyCryptodome**) - для розробки гібридної криптосистеми на платформі Linux. Тестування проводилося на трьох типах операцій: симетричне шифрування (AES-256 CBC), асиметричне шифрування (RSA-2048) та гешування (SHA-256).

Згідно з фінальною таблицею результатів, були отримані наступні дані:

OpenSSL продемонструвала найкращу загальну продуктивність. Вона виявилася найшвидшою у більшості тестів: шифрування AES (2.7962 с), шифрування RSA (0.000068 с), розшифрування RSA (0.000824 с) та гешування SHA-256 (0.8509 с).

Crypto++ показала дуже конкурентні, але змішані результати. Вона була найшвидшою у двох категоріях: розшифрування AES (0.5649 с) та генерація ключів RSA (0.0304 с). Однак, вона показала найгірший час у шифруванні AES (4.9810 с).

PyCrypto (**PyCryptodome**) виявилася найповільнішою бібліотекою майже у всіх категоріях, особливо суттєво поступившись у гешуванні SHA-256 (3.2125 с) та генерації ключів RSA (0.4331 с).

Такі розбіжності у продуктивності пояснюються фундаментальними відмінностями в архітектурі та мовах реалізації цих бібліотек:

OpenSSL (написана на C) та **Crypto++** (написана на C++) є компільованими бібліотеками. Це дає їм змогу виконувати бінарний код безпосередньо на машині, забезпечуючи максимальну швидкість, низькорівневі оптимізації та прямий доступ до апаратних інструкцій (наприклад, AES-NI для шифрування).

PyCrypto (**PyCryptodome**) є бібліотекою для Python. Хоча багато її криптографічних примітивів реалізовані на C для швидкості, самі операції викликаються через інтерпретатор Python. Цей прошарок інтерпретації створює значні накладні витрати, особливо при роботі з великими обсягами даних (як тестовий файл на 1 ГБ). Це безпосередньо пояснює, чому **PyCrypto** настільки повільніша в операціях, що залежать від вводу-виводу (AES, SHA-256).

Crypto++ в деяких тестах показала себе краще за OpenSSL, це можна пояснити наявністю специфічних оптимізацій для конеретних алгоритмів. Але з точки зору вибору криптографічної бібліотеки для промислового продукту, вона підходить менше, адже менш популярна та менш підтримувана (останній реліз 10/01/2023), що може призвести до наявності вразливостей та непередбачуваної поведінки, та відсутності сучасних криптоалгоритмів.

OpenSSL є де-факто промисловим стандартом, який використовується в більшості програм, що використовують криптографічні алгоритми, він є стандартною криптографічною бібліотекою Linux. Її продуктивність є

результатом десятиліть оптимізації на рівні асемблера та C, що забезпечує стабільно високу та збалансовану швидкість у всіх операціях, особливо в гешуванні (SHA-256) та симетричному шифруванні.

Для задачі розробки високопродуктивної гібридної криптосистеми на Linux, **OpenSSL** виглядає як найбільш збалансований та швидкий вибір. Crypto++ є сильною альтернативою. PyCrypto підходить для прототипування, але не для високонавантажених систем через накладні витрати Python.