Звіт з виконання лабораторної роботи 1
Варіант 2A (`OpenSSL` vs `PyCryptodome`)

ФІ-42мн Бондар Петро
ФІ-42мн Кістаєв Матвій

December 13, 2024

Огляд

Підгрупа: 2A.

Завдання (модифіковане): Порівняння бібліотек `OpenSSL` та `PyCryptodome` для розробки гібридної криптосистеми під Windows платформу.

Мета: Дослідження ефективності та доцільності використання бібліотек з точки зору контексту використання та вимог до ресурсів основних криптографічних алгоритмів.

Для порівняння було обрано невеликі набори основних алгоритмів для кожної специфічної задачі криптографії.

1. **Симетричні криптосистеми** (потокові та блокові шифри, довжина текстів: 1МБ):
   - AES: в режимі CBC, ключі розміру 128, 196 та 256;
   - ChaCha20 (Keccak): за стандартом, довжина ключа 256.
2. **Геш-функції** (довжина текстів: 1МБ):
   - SHA-2: МД-схема, варіант з 256-бітним гешем (SHA-256);
   - SHA-3: схема губки, варіант з 512-бітним гешем (SHA3-512).
3. **Генерація ключів для асиметричних криптосистем**:
   - RSA: публічні ключі розміру 1024, 2048 та 4096;
   - DSA: варіанти за FIPS 186-4, тобто (1024, 160), (2048, 256), (3072, 256), а також окремо порахований час для (2048, 224).
4. **Алгоритми цифрового підпису**:
   - RSA: підпис та перевірка підпису, ключі розміру 1024, 2048 та 4096;
   - DSA: підпис та перевірка підпису, ключі виду (1024, 160), (2048, 256), (3072, 256).

Виконання алгоритмів проводилось в приблизно однакових умовах, з мінімізацією впливу на заміри інших, допоміжних, функцій.

Імплементація тестового середовища представлена у відповідних папках репозиторію:

- `lab1\Bondar_Kistaiev_FI-03\OpenSSL` – реалізація на `OpenSSL`;
- `lab1\Bondar_Kistaiev_FI-03\PyCryptodome` – реалізація на `PyCryptodome`.

Для `OpenSSL` також доступні заміри часу для DSA-(2048, 224), генерації простих чисел і розшифрування за AES та ChaCha20. Їх можна знайти в папці `lab1\Bondar_Kistaiev_FI-03\OpenSSL\results`.

Результати порівняння

Порівняння було проведено з точки зору двох параметрів: використання ресурсів та загальна зручність та доречність використання.

Використання ресурсів обчислювального пристрою

Представлені нижче таблиці порівняння можна знайти у файлі `lab1\Bondar_Kistaiev_FI-03\labaaa.xlsx`. Час виражений у **секундах**, заміри зроблені із точністю, дозволеною вбудованими алгоритмами вимірювання часу.

Симетричні криптосистеми

	AES_Enc					
Key Length	*OpenSSL*			*PyCryptodome*		
	mean	median	std	mean	median	std
128	0.0055	0.0054	0.00057	0.017	0.017	0.0012
196	0.0076	0.0075	0.00053	0.019	0.019	0.0023
256	0.0073	0.0072	0.00044	0.02	0.02	0.0014

Figure 1: Порівняння часу шифрування тексту розміру 1МБ за допомогою AES (CBC).

	ChaCha20_Enc					
Key Length	*OpenSSL*			*PyCryptodome*		
	mean	median	std	mean	median	std
256	0.0019	0.0017	0.00064	0.024	0.024	0.0013

Figure 2: Порівняння часу шифрування тексту розміру 1МБ за допомогою AES ChaCha20.

Геш-функції

SHA256					
OpenSSL			*PyCryptodome*		
mean	median	std	mean	median	std
0.0042	0.0041	0.0005	0.029	0.029	0.0013

Figure 3: Порівняння часу гешування тексту розміру 1МБ за допомогою SHA256.

	ChaCha20_Enc					
Key Length	*OpenSSL*			*PyCryptodome*		
	mean	median	std	mean	median	std
256	0.0019	0.0017	0.00064	0.024	0.024	0.0013

Figure 4: Порівняння часу гешування тексту розміру 1МБ за допомогою SHA3-512.

Генерація ключів

	RSA_KeyGen					
Key Length	*OpenSSL*			*PyCryptodome*		
	mean	median	std	mean	median	std
1024	0.0058	0.0053	0.0017	0.081	0.069	0.052
2048	0.032	0.028	0.019	0.76	0.61	0.53
4096	0.31	0.27	0.19	7.14	5.47	5.37

Figure 5: Генерація ключів різних розмірів для RSA.

	DSA_KeyGen					
Key Length	*OpenSSL*			*PyCryptodome*		
	mean	median	std	mean	median	std
(1024, 160)	9.0E-05	8.5E-05	1.8E-05	0.38	0.25	0.39
(2048, 256)	3.9E-04	3.7E-04	5.4E-05	3.32	2.43	2.98
(3072, 256)	8.0E-04	7.8E-04	6.5E-05	23.09	16.68	25.55

Figure 6: Генерація ключів різних розмірів для DSA.

Алгоритми цифрового підпису

	RSA_Sign					
Key Length	*OpenSSL*			*PyCryptodome*		
	mean	median	std	mean	median	std
1024	0.00017	0.00016	3.10E-05	0.00073	0.00072	5.50E-05
2048	0.00043	0.00041	4.30E-05	0.0029	0.0029	5.50E-05
4096	0.0018	0.0017	0.00014	0.016	0.016	0.00044

Figure 7: Підписання повідомлення (малого розміру) за допомогою RSA.

	RSA_Sign_Verify					
Key Length	*OpenSSL*			*PyCryptodome*		
	mean	median	std	mean	median	std
1024	4.40E-06	4.00E-06	8.10E-07	0.00016	0.00016	1.80E-05
2048	1.20E-05	1.20E-05	1.00E-06	0.0004	0.0004	9.80E-06
4096	4.30E-05	4.30E-05	3.20E-06	0.0013	0.0013	6.90E-05

Figure 8: Верифікація підпису RSA для повідомлення (малого розміру).

	DSA_Sign					
Key Length	*OpenSSL*			*PyCryptodome*		
	mean	median	std	mean	median	std
(1024, 160)	7.4E-05	6.9E-05	1.5E-05	0.00035	0.00036	4.40E-05
(2048, 256)	2.7E-04	2.6E-04	2.5E-05	0.0012	0.0012	9.70E-05
(3072, 256)	6.4E-04	6.1E-04	2.1E-04	0.0026	0.0025	0.00014

Figure 9: Підписання повідомлення (малого розміру) за допомогою DSA.

	DSA_Sign_Verify					
Key Length	*OpenSSL*			*PyCryptodome*		
	mean	median	std	mean	median	std
(1024, 160)	0.0002	0.00018	0.0001	0.0005	0.00048	5.50E-05
(2048, 256)	0.00021	0.00018	0.00011	0.002	0.0019	0.00014
(3072, 256)	0.00034	0.0002	0.00036	0.0047	0.0046	0.00035

Figure 10: Верифікація підпису DSA для повідомлення (малого розміру).

Інтерпретація результатів

З огляду на вищенаведені таблиці, стає зрозумілою перевага швидшого `OpenSSL` над дещо (а іноді і набагато) повільнішим за нього `PyCryptodome`. Це спричинено підходом до реалізації самої бібліотеки (низькорівнева бібліотека на C++ проти високорівневого модуля на Python).

Безпека та ефективність використання пам'яті

Очевидним у ході виконання лабораторної роботи стало те, що `OpenSSL` дещо ефективніше розпоряджається ресурсами пам'яті, але це і не дивно з огляду на те, що бібліотека була розроблена та використана на низькорівневій мові програмування. З іншого боку, у цієї особливості є і недолік: контроль над вивільненням пам'яті ВСІХ структур, що використовуються для побудови криптосистеми, а також буферів пам'яті, покладається цілком і повністю на користувача. Це несе неабиякі ризики витоків пам'яті, що не є такою загрозою у `PyCryprodome`, так як там користувач не несе ніякої відповідальності за те, що відбувається у пам'яті.

Зручність та доречність використання

Роблячи загальне порівняння цих двох бібліотек можна виділити основні відмінності з точки зору користувацького досвіду: зручність та гнучкість використання бібліотек.

- **Зручність використання.**

  Бібліотека `PyCryprodome`, суб'єктивно, перемагає у своїй дружності до користувача. Все по канонам мови Python: модуль, а в ньому функція, функцію – викликати, результат – отримати. Мрія будь-якого програміста з ім'ям Матвій.

  В свою чергу `OpenSSL` неймовірно недружній до будь-кого, хто вперше користується цим інструментом. Документація присутня, але більшість функціоналу з детальним описом – застарілі, а що не застаріло має мало опису, прикладів, крайніх випадків та роз'яснень. Не кажучи вже про саму концепцію та ідеологію бібліотеки з використанням контекстів та контролем за кожним кроком виконання.

- **Гнучкість використання.**

  Тут, як не дивно, переможцем вже є `OpenSSL`. Його вищезгадана ідеологія дозволяє досвідченим користувачам гнучко налаштовувати виконання роботи певної криптосистеми, а також створювати свою та додавати її до загальної бібліотеки. Коротше, з великою силою приходить не тільки велика відповідальність, про яку казав дядько Бен, а ще й великі можливості.

Висновки

Якщо коротко описати наведений вище аналіз, то `OpenSSL` можна розглядати як ультимативний інструмент для багатьох задач, який пропонує гнучкість, швидкість, контроль над використанням пам'яті та багато чого іншого, але платою за це є складність використання самого інструменту. Це не просто бібліотека із доступом до вже існуючих криптоалгоритмів, а ще й справжня пісочниця, у якій можна імплементувати власні алгоритми.

Своєю чергою, `PyCryptodome` дає користувачу простоту використання, зрозумілу та чітку документацію, а також приклади для більшої кількості випадків.

Вердикт: `OpenSSL` створена для великого та серйозного промислового використання, де вирішує швидкість, ефективність використання ресурсів пам'яті, а також гнучність налаштування системи під себе, а `PyCryptodome` – чудовий представник бібліотеки, створеної для практики, навчання та створення додатків, невимогливих до ресурсів системи.