Лабораторна робота виконана на основі клієнт-серверної програми для лабораторної роботи password_storage.

Тепер ми зберігаємо нову інформацію про користувача – номер телефону.

Номер телефону буде зберігатися в базі даних разом з іншою інформацією про користувача. Ця інформація є відкритою для інших користувачів, проте її необхідно зберігати в зашифрованому вигляді в базі даних, щоб вберегтися від витоку одразу всіх номерів телефону усіх користувачів.

Ми використали метод envelope encryption.

- DEK генерується локально.
- DEK зберігається зашифрованим.
- Кожен DEK використовується тільки для одного користувача.
- Використовується надійний алгоритм шифрування (AES) в режиме счетчика Галуа (GCM).

Для шифрування даних кожного користувача ми створюємо окремий ключ шифрування даних(DEK).

Для шифрування і створення ключів ми використовуємо бібліотеку ***cryptography***, тому що вона має відкритий код, зарекомендувала себе як надійний інструмент, вона підтримується та активно розвивається. А конкретніше ми використовуємо AESGSM – це симетричний блочний шифр в режимі GCM (лічильник з автентифікацією Галуа). Даний режим автентифікованого шифрування дозволяє одержувачу легко виявити будь-які зміни, перш ніж почати його розшифровку, що значно покращує захист від спотворень.

Також ми використовуємо модуль ***secrets*** для генерації криптографічно стійких псевдовипадкових чисел.

DEK створюється за допомогою метода AESGCM.generate_key().

Сіль створюється за допомогою методу secrets.token_bytes()

DEK-и для користувачів ми зберігаємо зашифровано за допомогою KEK(ключ для шифрування ключів) в бінарному файлі поряд зі скриптом програми.

Сам KEK зберігається як змінна середовища.

Для того щоб вкрасти інформацію необхідно:

- мати доступ до бази даних
- мати доступ до бінарного фалу з DEK-ами
- мати KEK

Виконати всі ці пункти достатньо складно, тому дані є достатньо захищеними