**Інструкція до проекту Secure Messenger**

**Мета проєкту**

Метою проєкту є розробка системи захищеного обміну повідомленнями між клієнтами із використанням сучасних криптографічних методів. Проєкт забезпечує конфіденційність, цілісність та автентичність повідомлень за допомогою наскрізного шифрування та механізмів підтвердження доставки.

**Основні функціональні можливості**

1. **Наскрізне шифрування:**
   * Використання алгоритму ChaCha20-Poly1305 для шифрування повідомлень.
   * Обмін ключами за допомогою алгоритму Диффі-Хеллмана на основі кривої X25519.
2. **Перевірка цілісності:**
   * Генерація HMAC для кожного повідомлення для забезпечення цілісності та захисту від модифікацій.
3. **Автоматичне підтвердження доставки:**
   * Реалізація механізму ACK для підтвердження отримання повідомлень.
   * Повторна відправка повідомлень у разі відсутності підтвердження.
4. **Реалізація протоколу Double Ratchet:**
   * Динамічне оновлення ключів для кожного повідомлення.
   * Забезпечення преривного секрету в разі компрометації ключів.

**Короткий опис реалізації**

1. **Сервер:**
   * Сервер працює як посередник для пересилки зашифрованих повідомлень між клієнтами.
   * Здійснюється обмін публічними ключами між клієнтами.
2. **Клієнт:**
   * Кожен клієнт генерує пару ключів (приватний і публічний).
   * Встановлюється спільний секретний ключ через алгоритм Диффі-Хеллмана.
   * Повідомлення шифруються за допомогою ChaCha20-Poly1305 та передаються серверу для пересилки партнеру.
3. **Механізм HMAC:**
   * Для кожного повідомлення додається HMAC для перевірки цілісності та автентичності.
   * У разі невідповідності HMAC повідомлення вважається некоректним.
4. **Повторне надсилання:**
   * Кожне повідомлення має унікальний ідентифікатор.
   * Якщо підтвердження доставки не отримано, повідомлення відправляється повторно.

**Інструкція з використання**

1. **Запуск сервера:**
   * Перейдіть у директорію проєкту.
   * Запустіть сервер командою: ***python server.py***
   * Сервер буде слухати підключення на адресі 127.0.0.1 та порті 5000.
2. **Запуск клієнта:**
   * У новому терміналі запустіть клієнта:

***python client.py <Ваше\_ім’я> <Ім’я\_партнера>***

* + Приклад: ***python client.py Alice Bob***

1. **Відправка повідомлень:**
   * Введіть повідомлення у консоль клієнта та натисніть ***Enter***.
   * Щоб вийти, введіть exit.
2. **Протокол роботи:**
   * Клієнти автоматично обмінюються ключами та встановлюють спільний секрет.
   * Повідомлення шифруються та надсилаються серверу для пересилки партнеру.
   * У разі відсутності підтвердження повідомлення автоматично надсилається повторно.

**Файлова структура проекту**

/secure\_messenger

├── app/

│ ├── \_\_init\_\_.py # Налаштування логування для клієнта та сервера

│ ├── config.py # Реалізація Double Ratchet протоколу

│ ├── utilities.py # Допоміжні функції

│ ├── double\_ratchet.py

│ └── tests/ # Тестові сценарії

│ ├── \_\_init\_\_.py

│ ├── test\_keys.py # Тести для генерації та обміну ключами

│ ├── test\_encryption.py # Тести шифрування/дешифрування

│ └── test\_ratchet.py # Тести Double Ratchet

├── logs/

│ ├── client.log # Логи клієнта

│ ├── server.log # Логи сервера

├── client.py # Код клієнта

├── server.py # Код сервера

└── README.md # Документація проєкту

**Основні файли:**

1. **client.py** – клієнтська частина, включає реалізацію роботи з сервером, шифрування повідомлень, їх надсилання та отримання.
2. **server.py** – серверна частина, відповідає за прийом, пересилання повідомлень між клієнтами, обмін ключами.
3. **app/utilities.py** – допоміжні функції (генерація ключів, шифрування, розрахунок HMAC тощо).
4. **app/double\_ratchet.py** – реалізація протоколу Double Ratchet.
5. **app/config.py** – налаштування для логування.
6. **app/tests/** – тестові сценарії для перевірки ключових компонентів проєкту.
7. **logs/** – файли логів клієнта та сервера.

**Системні вимоги**

1. **Python 3.9 або вище.**
2. **Бібліотеки:**
   * cryptography
   * uuid
   * logging
   * socket
   * threading
3. **Система:**
   * Підтримка ОС Windows (не перевіряв Linux, macOS).

**Висновок**

Проєкт Secure Messenger демонструє використання сучасних криптографічних технологій для захищеного обміну повідомленнями. Його особливості включають наскрізне шифрування, підтвердження доставки, динамічне оновлення ключів та інтеграцію протоколу Double Ratchet. Це забезпечує високий рівень конфіденційності, автентичності та надійності в комунікаціях.