Міністерство освіти та науки України

Харківський національний університет радіоелектроніки

Кафедра програмної інженерії

Лабораторна робота №2

з дисципліни: «Безпека програм та даних»

на тему: «Програмна реалізація шифрування даних за допомогою

несиметричного криптоалгоритму RSA»

Виконав

ст. гр. ПЗПІ-20-1

Бабанін А.К.

Перевірив

доцент кафедри ПІ

Турута О.О.

2023

**Мета роботи:** Отримати навички використання методики роботи асиметричних алгоритмів шифрування. Реалізувати програмно (на будь-якій мові програмування) роботу алгоритму RSA.

Хід роботи

1. Розробити клієнт-серверний додаток, який використовує незахищений канал зв’язку. З клієнта на сервер захищено відправляється логін та пароль. Для реалізації клієнтського та серверного програмного забезпечення використовувати різні мови програмування. Для алгоритму шифрування RSA дозволяється використовувати бібліотеки.
2. Створити приємний та зрозумілий інтерфейс для перевірки зробленої роботи.
3. Реалізувати функції кодування для web-інтерфейсу, які (1) сформують ключі для відправлення даних на сервер, (2) відправлять дані та отримають відповідь.
4. Сформувати звіт в електронному вигляді.

Для початку сформулюємо принцип роботи додатку з урахуванням принципу роботи алгоритму шифрування RSA.

1. додаток має сформувати пару ключів для шифрування, один публічний інший приватний. Надалі використовуватимемо публічний ключ для шифрування і приватний для дешифрування.
2. обов’язки формування ключів покладаються на серверний додаток, адже він має отримувати зашифровані дані та мати можливість розшифрувати їх, для подальшої обробки.
3. клієнтський додаток повинен мати можливість відправити запит на отримання публічного ключа, та використовуючи його зашифрувати необхідне повідомлення.

Таким чином процес взаємодії можна описати таким чином: клієнт відправляє запит до серва на публічний ключ, сервер повертає публічний ключ зі згенерованної пари, клієнт шифрує повідомлення і надсилає його до серверу, сервер використовуючи приватний ключ дешифрує повідомлення.

Для використання алгоритму RSA застосовується бібліотека jsEncrypt з 2048 бітним ключем. Пароль та логін шифруються, додаються до JSON об’єкту та надсилаються на сервер.

Лістінг коду для шифрування форми і її відправки на сервер

const **handleSumbitButtonClick** = async () => {

    const jsEncrypt = new **JSEncrypt**();

    jsEncrypt.**setPublicKey**(publicKey);

    let kpassword;

    let klogin;

    let local\_login = login;

    let local\_password = password;

    if (useEncryption){

      local\_login = jsEncrypt.**encrypt**(login)

      local\_password = jsEncrypt.**encrypt**(password)

    }

    let request = {

      useEncryption: useEncryption,

      login: local\_login,

      password: local\_password

    };

    let response;

    try {

      response = await **fetch**('http://localhost:8080/login', {

        method: 'POST',

        headers: {

          'Content-Type': 'application/json',

        },

        body: JSON.**stringify**(request),

      });

    } catch (error) {

      console.**error**('Error during POST request:', error.message);

    }

    const result = await response.**json**();

**setDecryptedLogin**(result.login)

**setDecryptedPassword**(result.password)

    console.**log**(result)

  }

На серверній частині використовується бібліотека crypto. В залежності від параметру useEncryption відбувається дешифрування. Результат операції повертається до кліенту.

Лістінг коду для дешифрування:

app.**post**('/login', (req, res) => {

    console.**log**(req.body);

    const useEncryption = req.body.useEncryption;

    const login = req.body.login;

    const password = req.body.password;

    let response;

    if (!useEncryption){

        response = {

            login: login,

            password: password

        }

    }else{

        const decryptedLogin = **crypto**.**privateDecrypt**({ key: privateKey, padding: **crypto**.**constants**.RSA\_PKCS1\_PADDING }, **Buffer**.**from**(login, 'base64')).**toString**('utf-8');

        const decryptedPassword = **crypto**.**privateDecrypt**({ key: privateKey, padding: **crypto**.**constants**.RSA\_PKCS1\_PADDING }, **Buffer**.**from**(password, 'base64')).**toString**('utf-8');

        response = {

            login: decryptedLogin,

            password: decryptedPassword

        }

    }

    res.**json**(response)

});

Результат запиту публічного ключа можна побачити у консолі клієнтського застосунку:

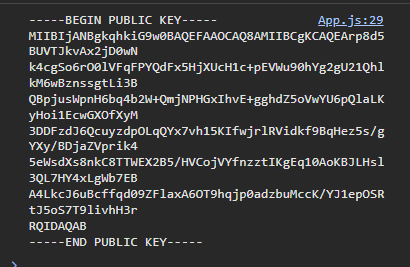


Рисунок 1 – Консольний лог про отримання публічного ключ

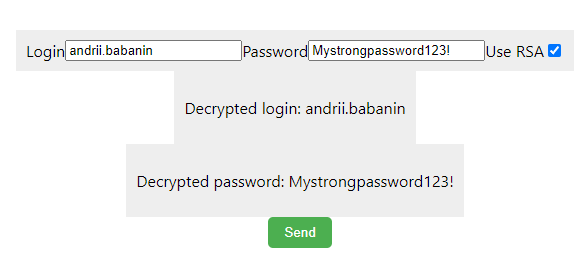


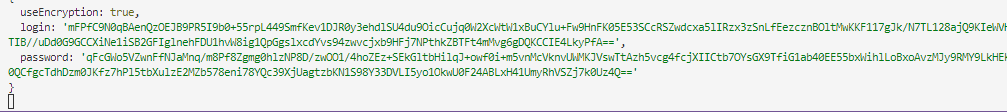
Рисунок 2 – Заповнена форма з паролем та логіном

Рисунок 3 - Серверний лог при отриманні зашифрованних данних

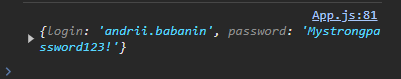


Рисунок 4 – Результат дешифрування

**Висновки:** В ході виконання лабораторної роботи було отримано навички використання методики роботи асиметричних алгоритмів шифрування. Реалізовано програмно роботу алгоритму RSA