

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 прийнятний та зрозумілий інтерфейс для перевірки зробленої роботи.
3. Реалізувати функції кодування для web-інтерфейсу, які (1) сформулюють 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 згенерованої 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.

Лістинг коду для шифрування форми і її відправки на сервер

```
const handleSubmitButtonClick = 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 клієнту.

Лістинг 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:

```
-----BEGIN PUBLIC KEY-----                               App.js:29
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEARp8d5
BUVTJkvAx2jD0wN
k4cgSo6rO0lVFqFPYQdFx5HjXUcH1c+pEVWu90hYg2gU21Qh1
kM6wBznssgtLi3B
QBpjusWpnH6bq4b2W+QmjNPHGxIhvE+gghdZ5oVwYU6pQ1aLK
yHoi1EcwGXOfXyM
3DDFzdJ6QcuyzdpOLqQYx7vh15KIfwjr1RVidkf9BqHez5s/g
YXy/BDjaZVprik4
5eWsdXs8nkC8TTWEX2B5/HVCojVYfnzztIKgEq10AoKB3LHs1
3QL7HY4xLgwb7EB
A4LkcJ6uBcfffqd09ZFlaxA60T9hqjp0adzbuMccK/YJ1ep0SR
tJ5o57T9livhH3r
RQIDAQAB
-----END PUBLIC KEY-----
```

Рисунок 1 – Консольний лог про отримання публічного ключ

Login

andrii.babanin

Password

Mystrongpassword123!

Use RSA ☒

Decrypted login: andrii.babanin

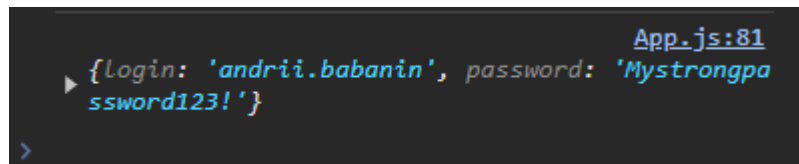
Decrypted password: Mystrongpassword123!

Send

Рисунок 2 – Заповнена форма з паролем та логіном

```
{
  "useEncryption": true,
  "login": "mFPfC9N0qBAenQzOEJ89PR5I9b0+55rpL449SmfKev1DJR0y3ehd1SU4du90icCujq0W2XciWl1xBuCY1u+Fw9HnFK05E53SCcRSZwdcxa51IRzx3zSnLfEezcznB01tMwKKF117gJk/N7TL128ajQ9KiekMTI8//uDd0G9GCCXiNe1i5B2GFIg1nehFDU1hWl8ig1QpGgs1xcdYvs94zwcjxb9HFj7NPthkZBTft4mVg6gDQKCCIE4LkyPFA==",
  "password": "qFcGlo5VZwnFNJaMnq/m8Pf8Zgm0h1zNP8D/zw001/4hoZEz+SEKGI1tbH1qJ+owf0i+m5vnMcVkmvUuMKJVswTtAzh5vcg4fcjXIICtb70YsGX9TfiG1ab40EE55bXWih1LoBxoAvzMJy9RMW9LkHEI0QCfGcTdhDzm0JKfz7hP15tbXu1zE2MzB578eni78YQc39XjUagtzbkN1S98Y33DVL15yo10kwU0F24ABLxH41UmyRhV5Zj7k0Uz4Q=="
}
```

Рисунок 3 - Серверний лог при отриманні зашифрованих даних

A screenshot of a code editor with a dark background. It shows a JavaScript object literal: `{login: 'andrii.babanin', password: 'Mystrongpassword123!'}`. The text is color-coded: `login` is blue, `'andrii.babanin'` is green, `password` is blue, and `'Mystrongpassword123!'` is green. A small blue triangle cursor is positioned to the left of the opening curly brace. In the top right corner, the text `App.js:81` is visible. A small blue greater-than symbol (`>`) is at the bottom left of the editor area.

```
App.js:81
▶ {login: 'andrii.babanin', password: 'Mystrongpa
  ssword123!'}
```

Рисунок 4 – Результат дешифрування

Висновки: В ході виконання лабораторної роботи було отримано навички використання методики роботи асиметричних алгоритмів шифрування.

Реалізовано програмно роботу алгоритму RSA