



Project Report:

RC2-CBC Encryption/Decryption Web Application

Submitted By:

Name:- Sarnaavho Pal

Roll:- 23

Year & Stream:- 3rd Year CSE (AIML)

Academic Year:- 2024-25

Enrolment number:- 12022002016036

Submitted To:

Prof. Indrajit De & Prof. Pabak Indu

Course: Introduction to Cyber Security

Institute of Engineering & Management, Kolkata

1. Introduction

Encryption is an essential aspect of cybersecurity, ensuring secure data transmission and storage. This project implements a web-based RC2 encryption and decryption system using Cipher Block Chaining (CBC) mode. The project demonstrates the working of RC2 encryption in a user-friendly interface, allowing users to generate keys and initialization vectors (IVs), encrypt plaintext, and decrypt ciphertext.

2. Objective

The objective of this project is to create an interactive web application that:

- Allows users to perform RC2 encryption in CBC mode.
- Enables key and Initialization Vector generation.
- Facilitates decryption using the provided key and Initialization Vector.
- Provides insights into the working of RC2 encryption with a simple demonstration.

3. Working Process of RC2-CBC Algorithm

3.1 RC2 Overview

RC2 is a symmetric-key block cipher that operates on 64-bit blocks of data with a variable key size (from 8 to 128 bits). It was designed by Ron Rivest and is now considered outdated due to vulnerabilities.

3.2 CBC Mode

Cipher Block Chaining (CBC) mode enhances security by XORing each plaintext block with the previous ciphertext block before encryption. The first block is XORed with an Initialization Vector (IV).

3.3 Encryption Process

- The user selects a key size (128-bit, 96-bit, or 64-bit).
- A random key and Initialization Vector are generated using *forge.random.getBytesSync()*.
- The plaintext message is XORed with the Initialization Vector (for the first block) or previous ciphertext block.
- The RC2 encryption function encrypts the resulting value.
- The encrypted output is displayed in hexadecimal format.

3.4 Decryption Process

- The user provides the encryption key, Initialization Vector, and ciphertext.
- The RC2 decryption function is initialized with the key.
- The ciphertext is decrypted in CBC mode, reversing the encryption process.
- The output plaintext is displayed.

4. Implementation Details

3.1 Key Generation (script.js)

```
function generateKey(elementId) {  
    const keySize = document.getElementById('key-size').value;  
    const keyBytes = forge.random.getBytesSync(parseInt(keySize));  
    const keyHex = forge.util.bytesToHex(keyBytes);  
    document.getElementById(elementId).value = keyHex; }  

```

3.2 Encryption (script.js)

```
function simulateEncryption() {  
    const cipher = forge.rc2.createEncryptionCipher(keyBytes);  
    cipher.start(ivBytes);  
    cipher.update(forge.util.createBuffer(plaintext, 'utf8'));  
    cipher.finish();  
    const encryptedHex = forge.util.bytesToHex(cipher.output.getBytes()); }  

```

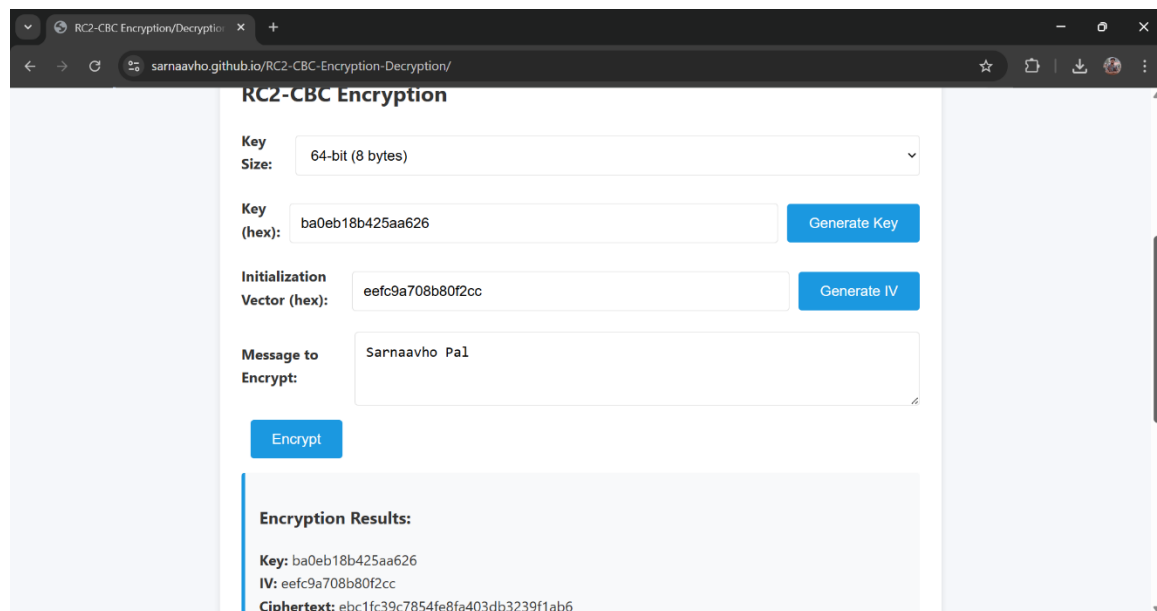
3.3 Decryption (script.js)

```
function simulateDecryption() {  
    const cipher = forge.rc2.createDecryptionCipher(keyBytes);  
    cipher.start(ivBytes);  
    cipher.update(forge.util.createBuffer(forge.util.hexToBytes(ciphertextHex)));  
    cipher.finish();  
    const decryptedText = cipher.output.toString('utf8'); }  

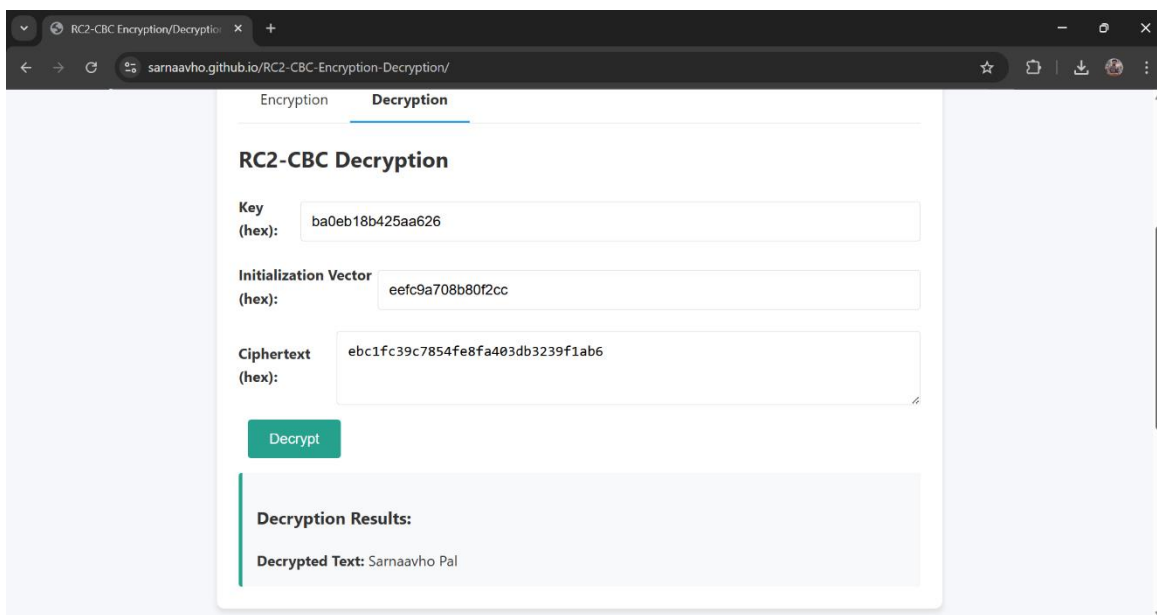
```

5. Results and Observations





The screenshot shows the 'RC2-CBC Encryption' section of a web application. It features a 'Key Size' dropdown set to '64-bit (8 bytes)'. The 'Key (hex)' field contains 'ba0eb18b425aa626' with a 'Generate Key' button. The 'Initialization Vector (hex)' field contains 'eefc9a708b80f2cc' with a 'Generate IV' button. The 'Message to Encrypt' field contains 'Sarnaavho Pal' and an 'Encrypt' button. Below these fields, the 'Encryption Results' are displayed: 'Key: ba0eb18b425aa626', 'IV: eefc9a708b80f2cc', and 'Ciphertext: ebc1fc39c7854fe8fa403db3239f1ab6'.



The screenshot shows the 'RC2-CBC Decryption' section of the same web application. It has tabs for 'Encryption' and 'Decryption', with 'Decryption' being the active tab. The 'Key (hex)' field contains 'ba0eb18b425aa626'. The 'Initialization Vector (hex)' field contains 'eefc9a708b80f2cc'. The 'Ciphertext (hex)' field contains 'ebc1fc39c7854fe8fa403db3239f1ab6' and a 'Decrypt' button. Below these fields, the 'Decryption Results' are displayed: 'Decrypted Text: Sarnaavho Pal'.

GitHub Link:- <https://github.com/SARNAAVHO/RC2-CBC-Encryption-Decryption.git>

Website Link:- <https://sarnaavho.github.io/RC2-CBC-Encryption-Decryption/>

5. Security Considerations

- RC2 is considered weak and should not be used for secure applications.
- The project demonstrates encryption concepts but is not suitable for production security.

7. Conclusion

This project provides an interactive demonstration of how RC2 encryption in CBC mode functions. While it does not perform real encryption, it helps users understand the key concepts of block ciphers, Initialization Vectors, and encryption workflows. The application serves as an educational tool rather than a secure encryption mechanism.

8. References

- <https://en.wikipedia.org/wiki/RC2>
- Cipher Block Chaining (CBC)