

Project Report: RC-4 Encryption and Decryption Tool

Introduction

The RC-4 Encryption and Decryption Tool is a web application developed using the Flask framework in Python. The tool leverages the RC4 (Rivest Cipher 4) algorithm, a symmetric stream cipher widely used in communication protocols and systems. The purpose of this project is to provide a user-friendly interface for encrypting and decrypting messages using the RC-4 algorithm.

RC4 Algorithm Overview

The RC4 algorithm is based on generating a continuous keystream to encrypt and decrypt data byte by byte. The key scheduling and keystream generation are crucial steps in the algorithm:

1. **Key Scheduling** (key_schedule function): The key provided by the user is scheduled to create an initial permutation of 256 bytes (S). The algorithm then generates a temporary array (T) based on the key.
2. **Keystream Generation** (generate_keystream function): The keystream is generated by repeatedly swapping elements in the permutation array (S). The algorithm uses the generated keystream to perform XOR operations with the plaintext or ciphertext, resulting in the encrypted or decrypted message.

Encryption and Decryption Functions

Encryption (encrypt function): This function takes a plaintext message and a key as input. It converts both the plaintext and the generated keystream into binary strings, performs XOR operations, and returns the result in hexadecimal form.

Decryption (decrypt function): This function takes a ciphertext and a key as input. It converts the ciphertext from hexadecimal to binary, generates the keystream, performs XOR operations, and returns the decrypted message in ASCII format.

Web Application

The Flask framework is used to create a simple web interface for users to interact with the RC-4 encryption and decryption functionalities. The application consists of an HTML form for inputting plaintext/ciphertext and the encryption/decryption key. The user can select the desired operation (Encrypt or Decrypt) and submit the form to view the result.

Challenges and Errors Faced

Problem-1: The main challenge is arising in the encryption process. In encryption, the plaintext is encrypted using the keystream and the encrypted text could be of non-printable characters (like TAB, NULL, SPACE, etc) which could not be displayed.

Solution: To overcome this problem, we produced an idea of displaying the ciphertext in its hexadecimal form. So that the non-printable characters cannot be missed at the time of decryption.

Problem-2: The next challenge is raised in the decryption process. In decryption, if the ciphertext is not in a valid hexadecimal format, then a value error is raised.

Solution: To overcome this problem, an exception case is introduced, so that only if the ciphertext is in valid hexadecimal format the decryption process starts. If not, an error message is displayed, and decryption process never executes.

Contributions

B Sai Sathwik (CB.EN.U4CYS22013) - Key Scheduling algorithm understanding and implementation in the code.

B Prem Kumar (CB.EN.U4CYS22014) - Decryption process understanding, error handling in decryption and implementation in the code.

B Rushyendra Reddy (CB.EN.U4CYS22015) - Encryption process understanding, error handling in encryption, introduction to usage of **FLASK SERVER**, and implementation in the code.

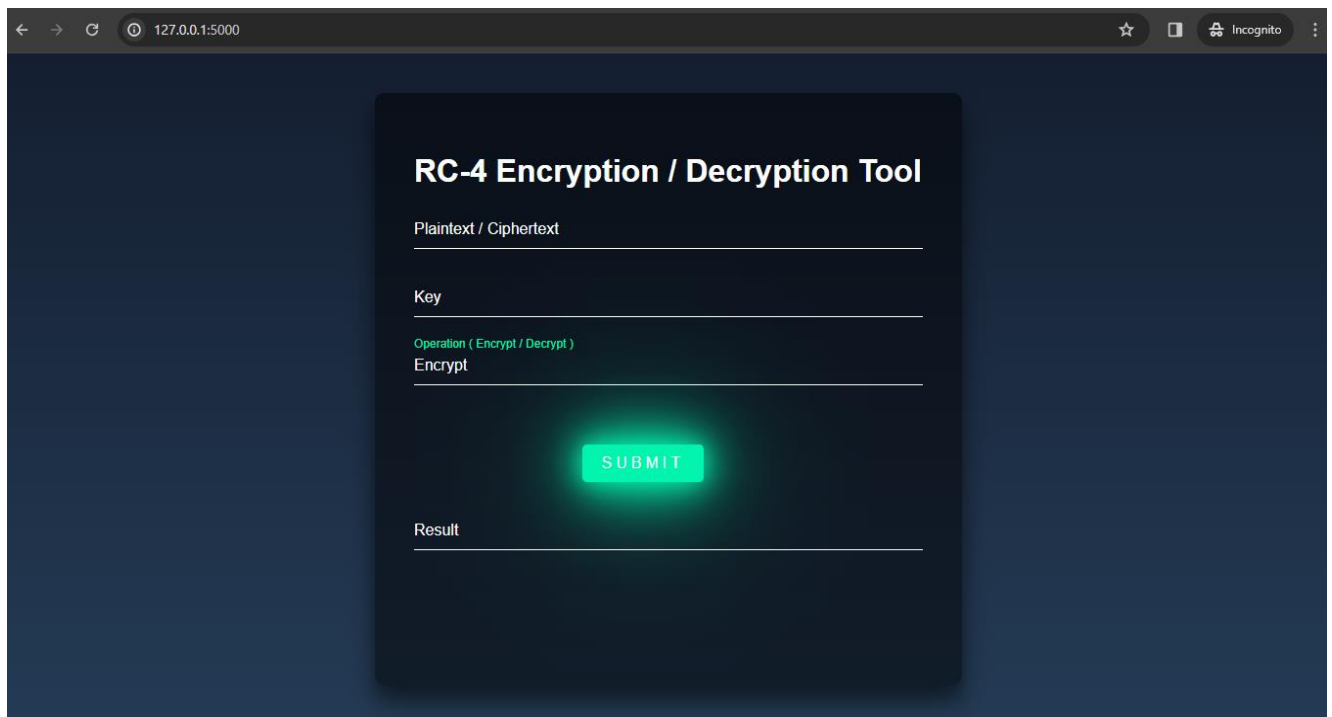
C S Amritha (CB.EN.U4CYS22016) - Theory (History of RC4, Variants in RC4), Key generation algorithm understanding and implementation in the code.

USER MANUAL OF RC4:

Step – 1: First, run the provided python code. Make sure that the HTML files are in the ‘templates’ folder.

Step – 2: After the start of the execution, open any browser (Google Chrome, Brave, Bing, etc) and connect to the IP 127.0.0.1:5000 . Make sure that the python program is running in the background.

Step – 3: After connecting to the IP server, the user interface looks as below

The image shows a web browser window with the address bar displaying '127.0.0.1:5000'. The browser is in Incognito mode. The main content area features a dark-themed interface titled 'RC-4 Encryption / Decryption Tool'. It contains several input fields: 'Plaintext / Ciphertext', 'Key', and 'Operation (Encrypt / Decrypt)'. The 'Operation' field currently has 'Encrypt' selected. Below these fields is a prominent red 'SUBMIT' button. At the bottom, there is a 'Result' label followed by an empty output field.

Step – 4: Enter the desired plaintext / ciphertext (ciphertext must be in hexadecimal form) in the first box.

RC-4 Encryption / Decryption Tool

Plaintext / Ciphertext
cybersecurity

Key

Operation (Encrypt / Decrypt)

SUBMIT

Result

Step – 5: Enter the desired key to encrypt the plaintext in the second box. In case of decryption, the key should be the same which is used for encryption. Otherwise, wrong results would be produced.

RC-4 Encryption / Decryption Tool

Plaintext / Ciphertext
cybersecurity

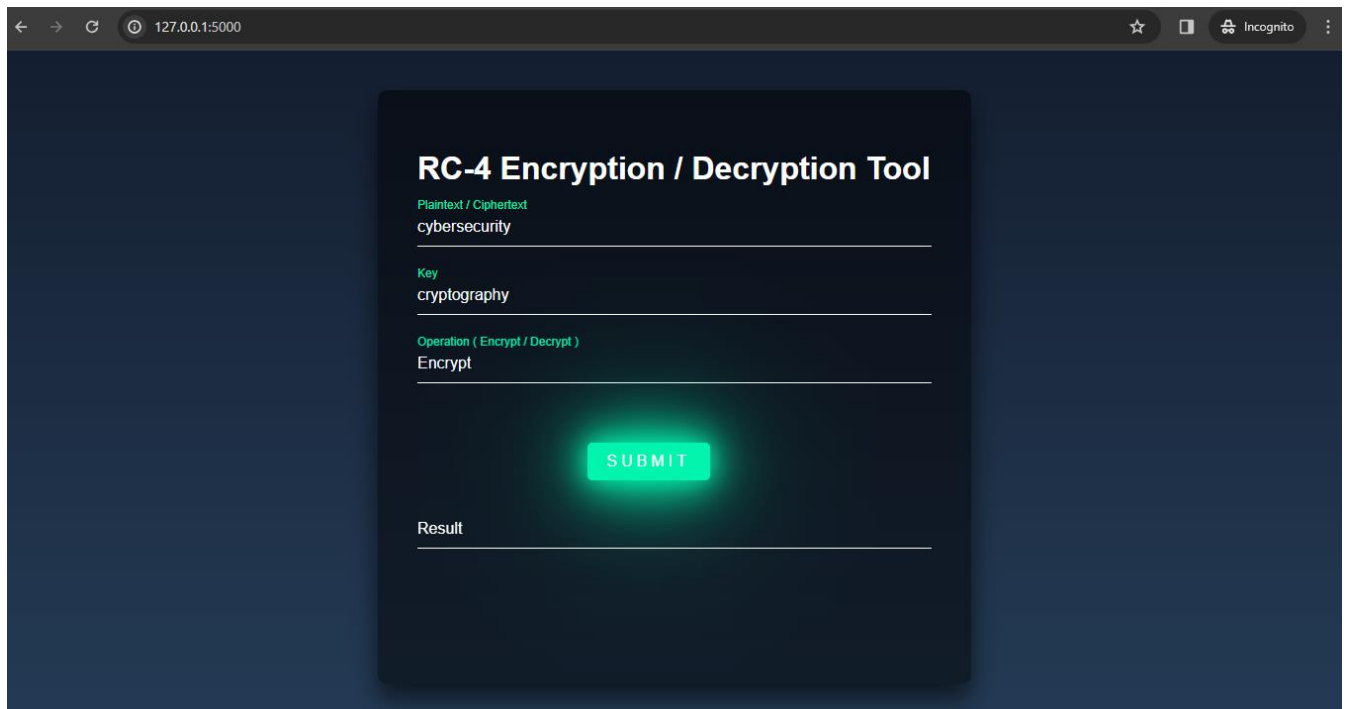
Key
cryptography

Operation (Encrypt / Decrypt)

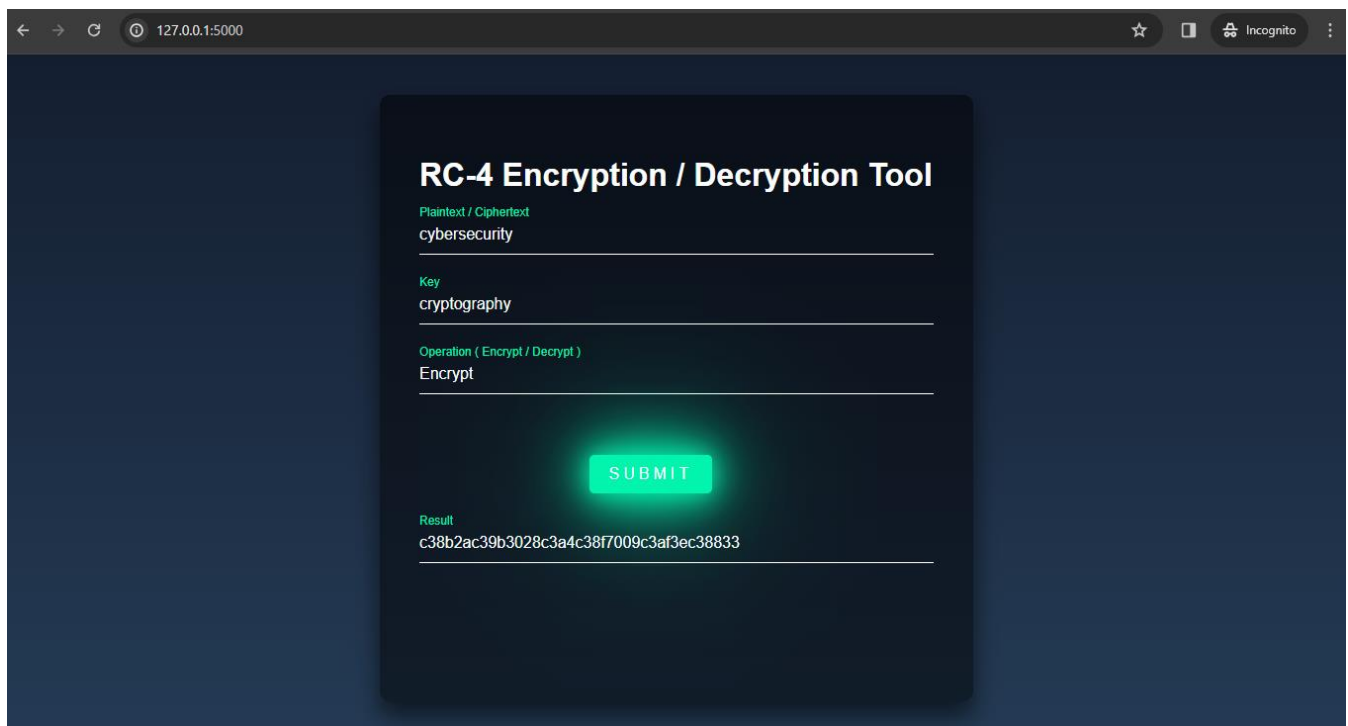
SUBMIT

Result

Step – 6: Enter the operation to be performed. Type “Encrypt” for encryption and type “Decrypt” for decryption. If any other text is entered, then error message “Invalid Operation” will be displayed.



Step – 7: After entering in the required fields, click on the Submit button to perform the operation.



Step – 8: Error messages are displayed under the result section if any errors are occurred.

Conclusion

The RC-4 Encryption and Decryption Tool successfully implements the RC4 algorithm for secure message encryption and decryption. The web application provides an accessible platform for users to utilize the RC-4 algorithm without requiring extensive technical knowledge. The project has faced and overcome challenges related to Encryption/Decryption processes. Future improvements may involve enhancing the application's security features and expanding its functionality.