CRYPTOGRAPHY TOOL

A Course Based Project Submitted in Partial Fulfilment of the Requirement for the Award of the degree of

BACHELOR OF TECHNOLOGY

COMPUTER SCIENCE AND ENGINEERING

Submitted by

21071A6204 - ALURU LAVANYA
21071A6217 - DUVVA HRISHIKESH
21071A6243 - PONNAM SIDHARDHA

DEPARTMENT OF CSE-CYS, DS & (AI &DS)

VALLURUPALLI NAGESWARARAO VIGNANA JYOTHI

INSTITUTE OF ENGINEERING AND TECHNOLOGY

(An Autonomous Institute, NAAC Accredited With 'A++' Grade, NBA Accredited, Approved by AICTE, New Delhi, Affiliated to JNTUH)

VALLURUPALLI NAGESWARARAO VIGNANA JYOTHI

INSTITUTE OF ENGINEERING AND TECHNOLOGY

(An Autonomous Institute)

CERTIFICATE

This is to Certify that A.LAVANYA (21071A6204), D.HRISHIKESH(21071A6217), P.SIDHARDHA(21071A6243) has successfully completed their project work at CSE CYS, DS & (AI & DS) Department of VNRVJIET, Hyderabad entitled "CRYPTOGRAPHY TOOL" in partial fulfilment of the requirements for the award of the Bachelor of Technology degree during the Academic year 2022-2023

Project Guide

Mrs.E. Lalitha
Assistant Prof. and Internal Guide
Dept. of CSE-CYS, DS and AI&DS
VNRVJIET

Head of Department

Dr.Mr. Rajasekhar
Prof. and Head
Dept. of CSE-CYS, DS and AI&DS
VNRVJIET

DECLARATION

This is to certify that the project work entitled "CRYPTOGRAPHY TOOL" submitted in VNR Vignana Jyothi Institute of Engineering & Technology in partial fulfilment of requirement for the award of Bachelor of Technology in Computer Science and Engineering. It is a Bonafide report of the work carried out by us under the guidance and supervision of Mrs.E.Lalitha (Assistant Professor), Department of CSE-CYS,DS,AI&DS, VNRVJIET. To the best of our knowledge, this report has not been submitted in any form to any university or institution for the award of any degree or diploma.

A.LAVANYA
21071A6204
CSE-CYS

D.HRISHIKESH
21071A6217
CSE-CYS

P.SIDHARDHA
21071A6243
CSE-CYS

ACKNOWLEDGEMENT

Behind every achievement lies the heartfelt gratitude to those who activated in completing the project. To them we lay the words of gratitude within us.

We are indebted to our venerable principal Dr. C.D. NAIDU for this inflicting devotion, which led us to complete this project. The support, encouragement given by him and his motivation led us to complete the project.

We express our sincere thanks to internal guide Mrs.E.Lalitha and also Head of the Department Dr. M. RAJA SHEKHAR for having provided us a lot of facilities to undertake the project work and guide us to complete the project.

We take the opportunity to express thanks to our faculty of the Dept. of COMPUTER SCIENCE AND ENGINEERING-CYBER SECURITY and remaining members of our college VNR VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY who extended their valuable support in helping us to complete the project in time.

A.LAVANYA (21071A6204)
D.HRISHIKESH (21071A6217)
P.SIDHARDHA (21071A6243)

ABSTRACT

Cryptography tools are essential in today's digital age where sensitive information needs to be protected from unauthorized access. A cryptography tool is a software application that provides various encryption and decryption techniques to secure data. This tool uses different cryptographic algorithms and protocols such as AES, RSA, Blowfish, and DES to encrypt and decrypt data. The user can choose from a variety of techniques such as symmetric key encryption, asymmetric key encryption, and hashing algorithms to secure their data. The tool offers a user-friendly interface for the user to select a specific algorithm, enter the input data, and view the encrypted output. It also provides error handling mechanisms to prevent data loss and to ensure data integrity. This abstract highlights the importance of cryptography tools in securing digital information and the variety of techniques available for the user to choose from.

CONTENTS

INTRODUCTION

The cryptography tool we have generated is a powerful software application that provides a range of encryption and decryption techniques to secure digital information. Cryptography is a science that ensures secure communication by transforming data into an unreadable format. The tool offers symmetric key encryption, asymmetric key encryption, and hashing algorithms to provide maximum security for the user's data.

The tool's usage is essential for organizations and individuals that deal with sensitive information and require data protection. The cryptography tool benefits the user by ensuring the confidentiality, integrity, and authenticity of the data. The tool's ease of use and user-friendly interface make it suitable for both novice and advanced users. The tool can be used in various real-life applications, including secure file sharing, online banking, e-commerce transactions, email communication, and password protection.

The cryptography tool plays a vital role in safeguarding sensitive information from unauthorized access and potential cyber-attacks. This paper provides a comprehensive analysis of the cryptography tool's usage, benefits, and real-life applications, emphasizing its importance in securing digital information.

The increasing dependence on digital communication and the rise of cyber-attacks have made the need for cryptography tools more critical than ever. Cryptography tools ensure secure communication and protect sensitive data from cyber threats.

The tool's advanced encryption algorithms provide strong security measures that guarantee confidentiality, integrity, and authenticity of data. With the cryptography tool, users can encrypt their data, files, and messages, preventing unauthorized access and ensuring that only authorized parties can view and modify the data.

The tool's ability to decrypt encrypted data also provides a crucial feature, allowing authorized users to access and read the protected data. Overall, the cryptography tool is an indispensable tool for individuals and organizations that require data security and protection in today's digital age.

LIBRARIES

hashlib:
A library that provides various secure hash algorithms.
In this tool, it is used to generate a hash of the user's password for secure storage.

getpass:
A library that allows a user to input a password securely without displaying it on the screen.
In this tool, it is used to prompt the user to enter their password without displaying it on the screen.

Crypto.Cipher.AES:
A library that provides advanced encryption standard (AES) encryption.
AES is a widely used encryption standard that is considered very secure.
In this tool, it is used to encrypt and decrypt the user's data.

Crypto.Util.Padding:
A library that provides padding functions for encryption algorithms.
In this tool, it is used to ensure that the data being encrypted is of the correct length for AES encryption.

base64:
A library that provides encoding and decoding of base64 data.
In this tool, it is used to encode and decode the encrypted data for secure transmission.

os:
A library that provides a way to interact with the operating system.
In this tool, it is used to check if a file exists before writing to it.

sys:
A library that provides a way to interact with the Python interpreter.
In this tool, it is used to exit the program if the user enters an invalid input.

Functions:

Here are the functions of CRYPTOGRAPHY TOOL:

caesar_cipher_encrypt(plaintext, shift) -

This function takes in a plaintext string and a shift value and applies the Caesar cipher encryption technique to the plaintext. The shift value specifies the number of positions to shift each letter in the plaintext alphabet.

caesar_cipher_decrypt(ciphertext, shift) -

This function takes in a ciphertext string and a shift value and applies the Caesar cipher decryption technique to the ciphertext. The shift value must be the same value used to encrypt the plaintext in order to obtain the original plaintext.

vigenere_cipher_encrypt(plaintext, key) -

This function takes in a plaintext string and a key string and applies the Vigenere cipher encryption technique to the plaintext. The key is a repeating keyword that is used to encrypt the plaintext.

vigenere_cipher_decrypt(ciphertext, key) -

This function takes in a ciphertext string and a key string and applies the Vigenere cipher decryption technique to the ciphertext. The key must be the same key used to encrypt the plaintext in order to obtain the original plaintext.

encrypt_aes(plaintext) -

This function takes in a plaintext string and applies the AES encryption technique using a secret key. The encrypted output is returned as a Base64 encoded string.

decrypt_aes(ciphertext) -

This function takes in a ciphertext string that was encrypted using the AES encryption technique and a secret key. The decrypted output is returned as a plaintext string.

CODE FOR QR GENERATOR

```python
from Crypto.Cipher import AES
from Crypto.Util.Padding import pad, unpad
import base64
import hashlib
class EncryptionTool:
    def __init__(self):
        self.alphabets = {
            'ENGLISH': 'ABCDEFGHIJKLMNOPQRSTUVWXYZ',
            'SPANISH': 'ABCDEFGHIJKLMNÑOPQRSTUVWXYZ'
        }

    def caesar_cipher_encrypt(self, plaintext, shift):
        ciphertext = ''
        for char in plaintext:
            if char.isalpha():
                char_index = self.alphabets['ENGLISH'].index(char.upper())
                shifted_index = (char_index + shift) % len(self.alphabets['ENGLISH'])
                ciphertext += self.alphabets['ENGLISH'][shifted_index]
            else:
                ciphertext += char
        return ciphertext

    def caesar_cipher_decrypt(self, ciphertext, shift):
        plaintext = ''
        for char in ciphertext:
            if char.isalpha():
                char_index = self.alphabets['ENGLISH'].index(char.upper())
                shifted_index = (char_index - shift) % len(self.alphabets['ENGLISH'])
                plaintext += self.alphabets['ENGLISH'][shifted_index]
            else:
                plaintext += char
        return plaintext

    def vigenere_cipher_encrypt(self, plaintext, key):
        ciphertext = ''
        key_index = 0
        for char in plaintext:
            if char.isalpha():
                key_char = key[key_index % len(key)].upper()
                alphabet = self.alphabets[char.upper()]
                key_index += 1
                char_index = alphabet.index(char.upper())
                shifted_index = (char_index + self.alphabets['ENGLISH'].index(key_char)) % len(alphabet)
                ciphertext += alphabet[shifted_index]
            else:
                ciphertext += char
        return ciphertext

    def vigenere_cipher_decrypt(self, ciphertext, key):
        plaintext = ''
        key_index = 0
        for char in ciphertext:
            if char.isalpha():
                key_char = key[key_index % len(key)].upper()
                alphabet = self.alphabets[char.upper()]
                key_index += 1
                char_index = alphabet.index(char.upper())
                shifted_index = (char_index - self.alphabets['ENGLISH'].index(key_char)) % len(alphabet)
                plaintext += alphabet[shifted_index]
```

```
                char_index = alphabet.index(char.upper())
                shifted_index = (char_index - self.alphabets['ENGLISH'].index(key_char)) % len(alphabet)
                plaintext += alphabet[shifted_index]
            else:
                plaintext += char
        return plaintext

    def encrypt_aes(self, plaintext, key):
        iv = b'0000000000000000'
        backend = AES.new(key.encode(), AES.MODE_CBC, iv)
        padded_plaintext = pad(plaintext.encode(), AES.block_size)
        ciphertext = backend.encrypt(padded_plaintext)
        return base64.b64encode(ciphertext).decode()

    def decrypt_aes(self, ciphertext, key):
        iv = b'0000000000000000'
        backend = AES.new(key.encode(), AES.MODE_CBC, iv)
        decrypted_text = backend.decrypt(base64.b64decode(ciphertext))
        unpadded_text = unpad(decrypted_text, AES.block_size).decode()
        return unpadded_text

if __name__ == '__main__':
    tool = EncryptionTool()
    plaintext = input('Enter plaintext: ')
    key = input('Enter key: ')
    print('Choose encryption/decryption technique:')
    print('1. Caesar Cipher')
    print('2. Vigenere Cipher')
    print('3. AES')
    choice = input('Enter your choice (1-4): ')

    if choice == '1':
        shift = int(input('Enter the shift value: '))
        plaintext = input('Enter the plaintext: ')
        tool = EncryptionTool()
        ciphertext = tool.caesar_cipher_encrypt(plaintext, shift)
        print(f'Ciphertext: {ciphertext}')
        decrypted_text = tool.caesar_cipher_decrypt(ciphertext, shift)
        print(f'Decrypted text: {decrypted_text}')

    elif choice == '2':
        key = input('Enter the key: ')
        plaintext = input('Enter the plaintext: ')
        tool = EncryptionTool()
        ciphertext = tool.vigenere_cipher_encrypt(plaintext, key)
        print(f'Ciphertext: {ciphertext}')
        decrypted_text = tool.vigenere_cipher_decrypt(ciphertext, key)
        print(f'Decrypted text: {decrypted_text}')

    elif choice == '3':
        key = input('Enter the key: ')
        plaintext = input('Enter the plaintext: ')
        tool = EncryptionTool(key)
        ciphertext = tool.encrypt_aes(plaintext)
        print(f'Ciphertext: {ciphertext}')
        decrypted_text = tool.decrypt_aes(ciphertext)
        print(f'Decrypted text: {decrypted_text}')

    elif choice == '4':
        print('Exiting program...')
    else:
        print('Invalid choice. Please try again.')
```

INPUT

OUTPUT

```
Enter plaintext: Hello, world!
Enter key: mykey
Choose encryption/decryption technique:
1. Caesar Cipher
2. Vigenere Cipher
3. AES
Enter your choice (1-4): 1
Enter the shift value: 3
```

```
Ciphertext: KHOOR, ZRUOG!
Decrypted text: HELLO, WORLD!
```

```
Enter plaintext: Hello, world!
Enter key: mykey
Choose encryption/decryption technique:
1. Caesar Cipher
2. Vigenere Cipher
3. AES
Enter your choice (1-4): 2
Enter the key: secret
Enter the plaintext: Hello, world!
```

```
Ciphertext: IBBMU, XPSME!
Decrypted text: HELLO, WORLD!
```

```
Enter plaintext: Hello, world!
Enter key: mykey
Choose encryption/decryption technique:
1. Caesar Cipher
2. Vigenere Cipher
3. AES
Enter your choice (1-4): 3
Enter the key: mysecretkey
Enter the plaintext: Hello, world!
```

```
Ciphertext: H4mZTwavx91nC00iS2QZKQ==
Decrypted text: Hello, world!
```

```
Enter plaintext: Hello, world!
Enter key: mykey
Choose encryption/decryption technique:
1. Caesar Cipher
2. Vigenere Cipher
3. AES
Enter your choice (1-4): 4
```

```
Exiting program...
```

CONCLUSION

In conclusion, the cryptography tool developed using Python provides a range of encryption and decryption techniques, including Caesar Cipher, Vigenere Cipher, and AES. These techniques can be used to secure data transmission and storage, preventing unauthorized access to sensitive information.

The tool's ease of use and versatility make it an ideal choice for individuals and organizations looking to secure their data. Additionally, the tool's open-source nature means that it can be customized and extended to meet specific needs.

Overall, the cryptography tool offers a powerful and flexible solution for securing data, making it an essential tool for modern-day data security.