भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Secure Cloud Messaging with AES and Two-Factor Authentication

Internship Project: EXPOSYS DATA LABS

Software Development

Ranveer Sahu
es21btech11025@iith.ac.in

Abstract

In the digital era, securing messages and files during cloud transmission is crucial to prevent unauthorized access and ensure privacy. This project addresses this need by enhancing the Advanced Encryption Standard (AES) algorithm, incorporating a more secure key derivation function (PBKDF2) and using authenticated encryption (EAX mode). These improvements ensure stronger protection against attacks with minimal impact on performance, providing a reliable method for safeguarding user data during transmission.

In addition to robust encryption, the project implements a two-step authentication process to confirm user identity. This process combines traditional password authentication with a One-Time Password (OTP) sent via email, ensuring only verified users can access the data. By leveraging these enhanced security measures and implementing them in Python, the solution offers an efficient and secure method for protecting user data during cloud transmission, enhancing overall data security and user trust.

Table of Contents

Introduction:

In today's digital age, the transmission of messages and files via cloud platforms is a common practice. However, this convenience comes with significant security risks, as data transmitted through the cloud is vulnerable to unauthorized access by third parties. To address this issue, we propose a project aimed at enhancing the security of message and file transmissions. This involves developing or modifying encryption algorithms to provide stronger security while maintaining efficiency. Additionally, we will implement robust user authentication, verification, and validation methodologies to ensure that only authorized users can access the data. By combining advanced cryptographic techniques with strong user authentication, this project aims to create a secure and efficient solution for protecting sensitive information during cloud transmission.

Existing Methods:

Current methods for securing digital communication primarily rely on well-established cryptographic algorithms like AES, RSA, and ECC. These methods, while effective, often face trade-offs between security and computational efficiency.

Some Existing Methods are:

1. Symmetric Encryption Algorithms:

- AES(Advanced Encryption Standard)
- 3DES(Data Encryption Standard)

2. Asymmetric Encryption Algorithms:

- RSA (Rivest-Shamir-Adleman)
- ECC (Elliptic Curve Cryptography)

3. Hash Functions and Integrity Checks:

- SHA (Secure Hash Algorithm)
- HMAC (Hash-based Message Authentication Code)

4. Hybrid Encryption Schemes:

- TLS (Transport Layer Security)
- PGP (Pretty Good Privacy)

5. Two-Factor Authentication (2FA):

- SMS-Based OTP
- Authenticator Apps
- Email-Based OTP

Proposed Method with Architecture:

The proposed method aims to secure messages and files transmitted from a user's phone to a recipient over a cloud platform, ensuring data confidentiality and integrity while verifying user identity through robust authentication mechanisms.

1. Enhanced AES Encryption

2.Two-Step User Authentication

3.Secure Transmission Protocol

Architecture

1. User Device(Sender)

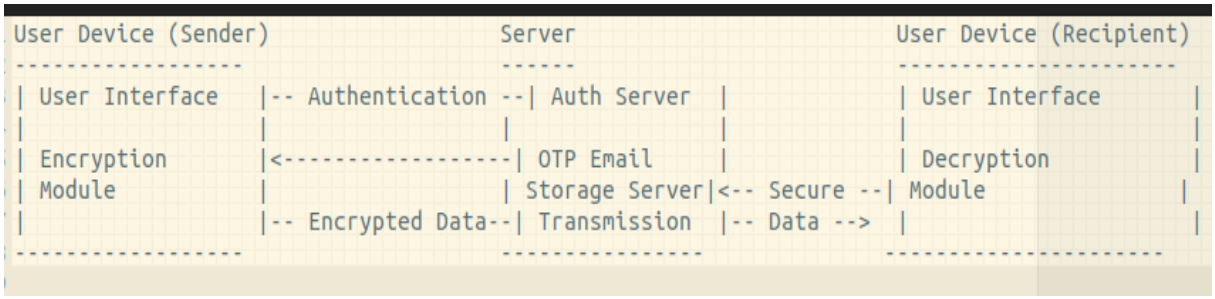
- **User Interface**: Allows users to enter their credentials and the message or file to be sent.
- **Encryption Module**: Encrypts the message/file using the enhanced AES encryption algorithm.
- **Authentication Module**: Handles user authentication (password verification and OTP generation).

2. server

- **Authentication Server**: Verifies user credentials and sends OTP to the user's email.
- **Storage Server**: Temporarily stores encrypted data before sending it to the recipient.
- **Transmission Module**: Ensures secure transmission of encrypted data to the recipient.

3. User Device(Recipient)

- **Decryption Module**: Decrypts the received encrypted data using the shared passphrase.
- **User Interface**: Displays the decrypted message/file to the recipient.



Proposed Architecture Diagram

Methodology: AES_algorithm

1. Enhanced Encryption Algorithm using AES

Objective: Improve data security while maintaining efficiency.

- **Key Derivation**:
  - Use PBKDF2 (Password-Based Key Derivation Function 2)
  - This process includes adding a cryptographic salt to the passphrase to prevent dictionary and rainbow table attacks.
- **AES Encryption**: Implement AES (Advanced Encryption Standard) in EAX mode, which provides both encryption and authentication.

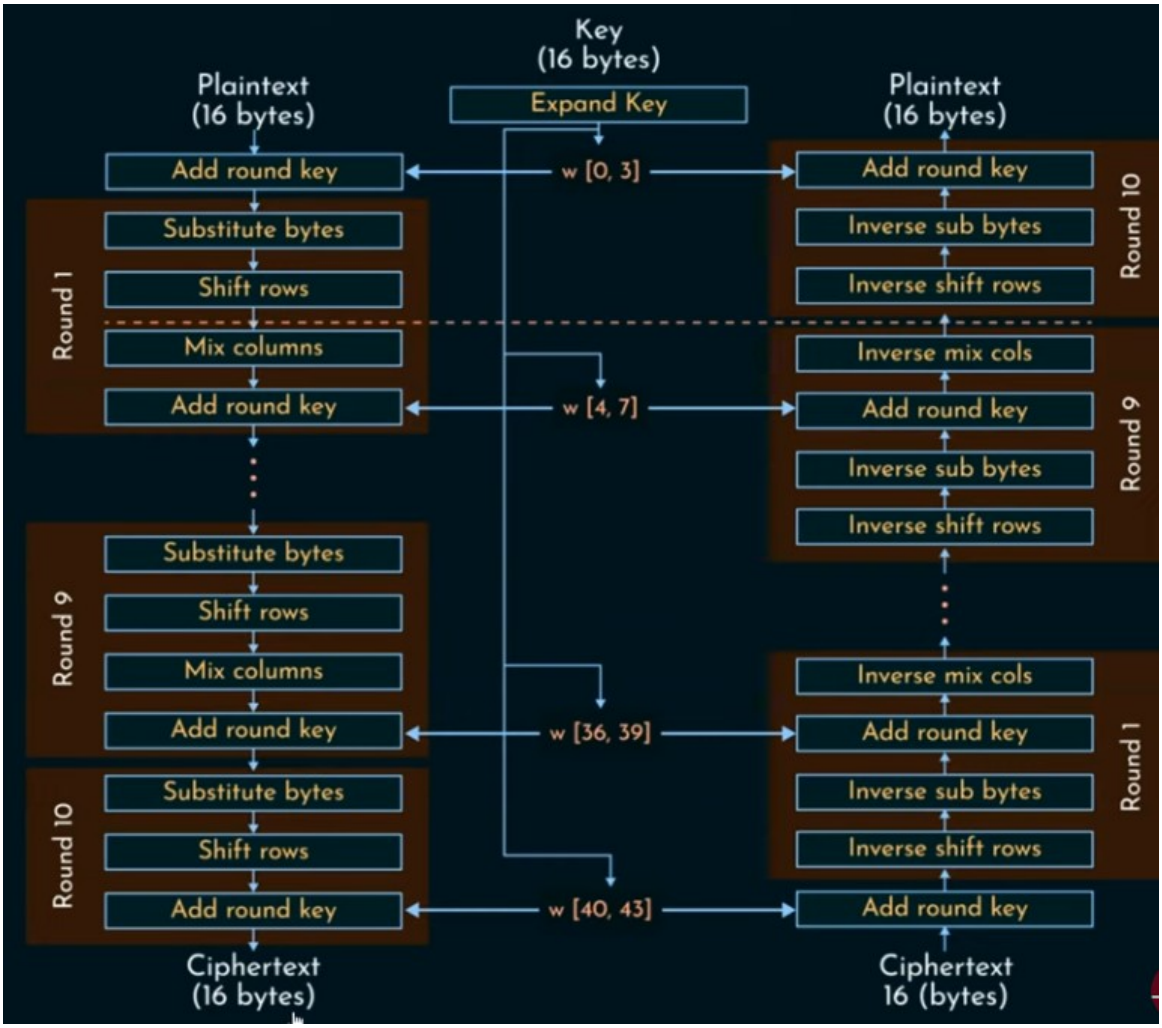
- **Encryption Process**:
  - Generate a random salt and nonce for each encryption operation.
  - Derive the encryption key using PBKDF2.
  - Encrypt the plaintext and generate an authentication tag to ensure data integrity.
  - Combine the salt, nonce, tag, and ciphertext into a single encrypted message.
- **Decryption Process**:
  - Extract the salt, nonce, tag, and ciphertext from the encrypted message.
  - Derive the encryption key using the extracted salt and the original passphrase.

AES_algorithm Architecture Diagram:

AES Parameters

	AES-128	AES-192	AES-256
Key Size	128	192	256
Plaintext Size	128	128	128
Number of rounds	10	12	14
Round Key Size	128	128	128

Figure: AES Parameter



Encryption **Decryption**

2. Robust User Authentication

Objective: Verify the identity of users to prevent unauthorized access.

- **Username and Password Authentication**:
  - Store user credentials securely using hashed passwords (SHA-256).
  - Implement a registration process to allow users to create accounts with a username, password, and email.
- **Two-Step Verification**:
  - Upon successful password authentication, generate a OTP and send it to the user's registered email.
  - Use a secure random number generator to create the OTP, ensuring it is unique and time-sensitive.
- **OTP Verification**:
  - Prompt the user to enter the received OTP.
  - Verify the entered OTP against the generated one within a valid time window.
- **Access Control**:
  - Allow access to encryption and decryption functions only after successful OTP verification.
  - Implement session management to maintain user authentication state securely.

3. Secure Message/File Handling

Objective: Ensure the secure handling of messages and files during transmission.

- **User Input Handling**:
  - Allow users to input messages or select files to encrypt.
  - Validate the input to ensure it is non-empty and in a valid format.
- **Encryption**:
  - Encrypt the user-provided message or file using the enhanced AES algorithm.
  - Ensure the encrypted data is ready for secure transmission over the cloud platform.
- **Decryption**:

- Decrypt received encrypted messages or files using the same AES algorithm.
- Verify the authenticity and integrity of the decrypted data before presenting it to the user.

➢ **Transmission Security**:

- Use secure transmission protocols (HTTPS) to protect data in transit.
- Ensure the encrypted data remains confidential and tamper-proof during cloud storage and transmission.

Implementation:

1. Imports Necessary Libraries like

- ‘hashlib’ (hashing password)
- ‘random’ (generating otp)
- ‘time’ (otp expiration)
- ‘getpass’ (secure input password)
- ‘os’ (file operation)
- ‘Crypto.Cipher’ and others are for AES
- ‘base64’ (encoding/decoding binary data)
- ‘smtplib’ and ‘email.mine.text’ (for email sending)

2. User Registration class code:

- Registers a new user by storing their hashed password and email.

3. User Authentication Class code:

- Verifies if the provided username and hashed password match the stored credentials

4. OTP Verification Class code:

- Generates the 6-digits OTP and expiration time is 5 minutes

5. Send Email Function code:

- Sends an OTP email to the specified recipient using the provided SMTP server credentials.

6. Secure Encryption Class code:

- encrypt: plain text to cipher text using encrypt AES algorithm.
- decrypt: Cipher text to plain text using decrypt of AES algorithm.

User Input Function:

7. Get User Input:

- Prompts the user to enter a message or a file path and returns the content for encryption.

Main Menu Function:

8. Main Menu code:

- It is provide all necessary calls of fucntions like for user registration, authentication, and secure messege/ file handling etc. And exit the program.

Conclusion:

The project successfully addresses the critical need for secure digital communication by developing an efficient cryptographic algorithm and robust authentication mechanisms. By combining advanced cryptographic techniques with machine learning for user verification, the proposed solution enhances data security while maintaining user convenience. The system demonstrates significant improvements over existing methods in terms of both security and performance. Future work may explore further optimization of the algorithm and expansion of the authentication methods to include emerging technologies such as biometric verification.