

A Report On

Cryptographic Secure Messaging

System Using AES

Submitted by

Name	Entry Number
ARINDAM SAL	2024JCS2041
AYAN SHIL	2024JCS2048

1. Introduction & Goal of the Assignment

This assignment implements a secure messaging application that ensures confidentiality, integrity, message freshness, and username privacy using Advanced Encryption Standard (AES) encryption in CBC mode. We use password-derived keys to secure client-server communication, enabling safe message storage and retrieval by encrypting usernames and messages.

The following figure demonstrates the flow of messages and key generation. The sender uses a password to derive a key using PBKDF2, which, combined with a random Initialization Vector (IV), encrypts both the username and message content before sending it to the receiver.

Representation:

- **Sender (A) → Receiver (B)**

- **Key Derivation (PBKDF2):** Converts password to key
- **AES-CBC Mode:** Encrypts messages with confidentiality
- **IV (Initialization Vector):** Unique per message for replay protection

2. Implementation Features

This section highlights the implementation features, with code snippets demonstrating each feature's fulfillment of assignment requirements.

2.1 Password-Derived Encryption Key (Using PBKDF2)

- The password is converted into a secure key using PBKDF2. This function generates a key using a salt to add entropy.

```
def derive_key(password, salt):
    return PBKDF2(password, salt, dkLen=AES_KEY_SIZE)
```

2.2 AES Encryption with Random IV for Confidentiality

- AES encryption is performed in CBC mode, with a unique 16-byte IV for each message, ensuring each encrypted output remains distinct.

```
iv = get_random_bytes(AES_BLOCK_SIZE)
cipher = AES.new(key, AES.MODE_CBC, iv)
```

2.3 Integrity and Replay Prevention with Random IVs

- A unique, random IV is generated for each message, preventing replay attacks and ensuring identical messages have unique encrypted outputs.

```
iv = get_random_bytes(AES_BLOCK_SIZE)
```

2.4 Socket-Based Client-Server Communication

- The client and server communicate over a secure socket, transferring encrypted messages and usernames.

```
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
```

```
    s.connect((SERVER_IP, SERVER_PORT))
```

```
    s.sendall(str(data_packet).encode())
```

2.5 Database Storage(SQLite3) for Encrypted Messages

- Each message is securely stored in an SQLite database, along with encrypted usernames, IVs, and salts.
- Additional admin features:
 - Admin can delete individual messages or clear the entire database.
 - Admin can decrypt messages if the correct password is provided.

```
def store_message(user_id, message, iv_user, iv_message, salt_user, salt_message):  
  
    conn = sqlite3.connect("messages.db")  
  
    # ... SQL insertion of encrypted message data ...
```

3. Security Objectives

This section describes how the implementation achieves the assignment's security objectives.

3.1 Confidentiality

- The encryption ensures that only intended recipients with the correct password can decrypt and read messages.
 - **Screenshot:** Encrypted message in the database.

```
Enter Username: arindamsal
Enter Password:
Enter Message: Test!
Message sent securely!

Enter Username: ayanshil
Enter Password:
Enter Message: Hii
Message sent securely!

PROBLEMS DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\Arindam Sal\Desktop\IIT Data Base Semester-1\CS-COL759\Assignment\Assignment-3\SecureMessagingApp> python server/server.py
Welcome Admin!
Enter Master Password to Start the Server:
Server started. Waiting for messages...
No messages received. Type 'exit' to stop or press Enter to check for new messages:

ID | Encrypted Username | Encrypted Message | Timestamp
-----
1 | b'\x19\xac\xaf\x04\x97\xbf\x05\xf5\x8e\x02\xfb\xea' | b'\x06\xca\xbd\xcf\xd3\x8a\x05\xc5\xde\x8a' | 2024-11-12 22:26:15,580214
2 | b'\xf0\xfb\x03\x12\x81\x21\x02\x0f\x0d\x8c\x6\x0e' | b'\x0d\x0f\xfb\x0c\x85\x00\x7f\x8a\x01\x09\xdb\x80' | 2024-11-12 22:26:59,315764
Enter Message ID to view, 'delete <ID>' to delete a message, 'drop table' to delete all messages, or 'exit' to quit:
|
```

3.2 Integrity

- The AES-CBC mode ensures that tampered messages fail to decrypt correctly, preserving the integrity of each message.

3.3 Message Freshness and Replay Attack Prevention

- Each message has a unique IV, ensuring identical messages appear differently in encrypted form, preventing replay attacks.

3.4 Username Privacy

- Usernames are encrypted separately and stored with a fixed length (16 bytes max), ensuring privacy.
- **Screenshot:** Encrypted usernames stored in the database

ID	Encrypted Username	Encrypted Message	Timestamp
1	b'Y\xfa\x94l\xc2\xc7~U%\xf1\xaa\xad\x18r\xf4'	b":\x08{s&\x98\xf4\xc8\xcb'\xebQ[4\xf5\xfe"	2024-11-12 22:30:30.385707
2	b'gx.\xe8\x86\xd8\xf6\x8a\x19\xa3X#F\xcf\xd2'	b'- \xeb-\xJ\xc4Ro\x8bIn\x967:)'	2024-11-12 22:30:58.332645

3.5 Extra Layer of Security

- A master password protects sensitive operations on the server, like message deletion or database reset, adding an extra layer of security.
- **Screenshot:** Prompt for master password before performing admin actions

PS C:\Users\Arindam Sait\Desktop\IITD Data base\Semester-1\CNS-COL759\Assignment\Assignment-3\SecureMessagingApp> python server/server.py		
Welcome Admin!		
Enter Master Password to Start the Server:		
Server started. Waiting for messages...		
ID Encrypted Username Encrypted Message Timestamp		
1 b'Y\xfa\x94l\xc2\xc7~U%\xf1\xaa\xad\x18r\xf4'	b":\x08{s&\x98\xf4\xc8\xcb'\xebQ[4\xf5\xfe"	2024-11-12 22:30:30.385707
2 b'gx.\xe8\x86\xd8\xf6\x8a\x19\xa3X#F\xcf\xd2'	b'- \xeb-\xJ\xc4Ro\x8bIn\x967:)'	2024-11-12 22:30:58.332645
Enter Message ID to view, 'delete <ID>' to delete a message, 'drop table' to delete all messages, or 'exit' to quit: █		

4. Environment Setup & Running the Application

Details on dependencies and the commands to set up and run the application.

4.1. Dependencies

- **Python packages:** Need to Install required packages(`pycryptodome==3.14.1`, `sqlite3` etc) at both client and server sides:

```
pip install -r client/requirements.txt
```

```
pip install -r server/requirements.txt
```

4.2. Commands

- **Running the Server:** `python server/serevr.py`
- **Running the client:** `python client/client.py`

4.3. Master password:

5. Testing Strategy

Each test case below verifies different aspects of the application.

5.1. Basic Message Encryption and Decryption

- Ensures the message encrypts and decrypts successfully.
 - **Screenshot:** Encrypted and decrypted messages match.

```
Enter Username: ayanshil
Enter Password:
Enter Message: This is a trial message....
Message sent securely!

PROBLEMS DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\Arindam.Sal\Desktop\IITD Data base\Semester-1\CNS-COL759\Assignment\Assignment-3\SecureMessagingApp> python server/server.py
Welcome Admin!
Enter Master Password to Start the Server:
Server started. Waiting for messages...
No messages received. Type 'exit' to stop or press Enter to check for new messages:

ID | Encrypted Username | Encrypted Message | Timestamp
1 | b'^~\xc4\xc3!\lx2\xd5\xe4!\xc)\xcc9\x94\xb2' | b'\xa6\xcc9\xfe\x878\xd8P\xe0\xd9\x1e91\x0cF\xd0\xad\x86\xba\xd5U\x14\xb2NA\xdc*\xb8' | 2024-11-12 22:38:00,363052
Enter Message ID to view, 'delete <ID>' to delete a message, 'drop table' to delete all messages, or 'exit' to quit: 1
Enter Password:
Decrypted Username: ayanshil
Decrypted Message: This is a trial message....
```

5.2. Maximum Username Length (16 Bytes)

- Verifies that usernames with exactly 16 bytes are encrypted and stored correctly.
 - **Screenshot:** Showing storage of 16-byte username.

```
Enter Username: abcdefghijklmno
Enter Password:
Enter Message: 16 bit username length
Message sent securely!

Enter Username: [ ]
```

PROBLEMS DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\Arindam\Sal\Desktop\IITD Data base\Semester-1\CHS-COL759\Assignment\Assignment-3\SecureMessagingApp> python server/server.py

Welcome Admin!

Enter Master Password to Start the Server:

Server started. Waiting for messages...

ID | Encrypted Username | Encrypted Message | Timestamp

1 | b'5\xc4\xc3"\x12\xd5\xe41\xc3\xc0\x94\xb2' | b'\xa6\xcc\xfe\x878\xd8P\xe8&\xd91\xd1e\x191\x0cf\x10\xad\x86\xba\xd5\x14\xb2NA\xdc*\xb8' | 2024-11-12 22:38:00.363052

22:40:42.647723

Enter Message ID to view, 'delete <ID>' to delete a message, 'drop table' to delete all messages, or 'exit' to quit: 2

Enter Password:

Decrypted Username: abcdefghijklmno

Decrypted Message: 16 bit username length

5.3. Replay Attack Prevention

- Ensures unique IV for each identical message, preventing replays.
 - **Screenshot:** Encrypted output differs for identical messages.

```
Enter Username: arindamsal
Enter Password:
Enter Message: Hello world
Message sent securely!

Enter Username: arindamsal
Enter Password:
Enter Message: Hello world
Message sent securely!

PROBLEMS DEBUG CONSOLE TERMINAL PORTS

Enter Message ID to view, 'delete <ID>' to delete a message, 'drop table' to delete all messages, or 'exit' to quit: 1
Enter Password:
Decrypted Username: arindamsal
Decrypted Message: Hello world

ID | Encrypted Username | Encrypted Message | Timestamp
---|---|---|---
1 | b'\x7f\xfa\x0d1\xc2\xc7\x0\xcf\xaf\xaa\x1\x18\xcf4' | b'\x08\x0\x98\xf4\xcb\xeb0[4\xf5\xfe' | 2024-11-12 22:30:30.385707
2 | b'gx\x08\xb6\xdb\xfc\x8a\x19\xaa\xef\xcf\xd2' | b'-\x eb-\x\c4\xb1n\x9e67;) | 2024-11-12 22:30:58.332645
Enter Message ID to view, 'delete <ID>' to delete a message, 'drop table' to delete all messages, or 'exit' to quit: 2
Enter Password:
Decrypted Username: arindamsal
Decrypted Message: Hello world

ID | Encrypted Username | Encrypted Message | Timestamp
---|---|---|---
1 | b'\x7f\xfa\x0d1\xc2\xc7\x0\xcf\xaf\xaa\x1\x18\xcf4' | b'\x08\x0\x98\xf4\xcb\xeb0[4\xf5\xfe' | 2024-11-12 22:30:30.385707
2 | b'gx\x08\xb6\xdb\xfc\x8a\x19\xaa\xef\xcf\xd2' | b'-\x eb-\x\c4\xb1n\x9e67;) | 2024-11-12 22:30:58.332645
Enter Message ID to view, 'delete <ID>' to delete a message, 'drop table' to delete all messages, or 'exit' to quit: |
```

5.4. Invalid Password for Decryption

- Checks that decryption fails with an incorrect password.
 - **Screenshot:** Failed decryption message.

5.5. Special Characters in Username and Message

- Confirms that non-ASCII characters encrypt and decrypt correctly.
 - **Screenshot:** Display of special characters after decryption.

5.6. Multiple Clients with the Same Password

- Checks that multiple clients can send messages securely.
 - **Screenshot:** Shows encrypted messages from different clients.

5.7. Empty Message

- Ensures that empty messages are encrypted and stored without errors.
 - **Screenshot:** Show encrypted empty message.

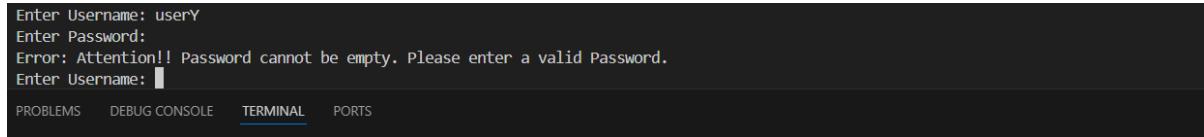
5.8. Username Exceeds 16 Bytes

- Verifies that usernames longer than 16 bytes are rejected.
 - **Screenshot:** Error message for username length limit.

```
Enter Username: abcdefghijklmnopqrstuvwxyz
Enter Password:
Enter Message: Username Exceeds 16 Bytes
Error: Username cannot exceed 16 bytes in UTF-8. Please use a shorter username.
Enter Username: [REDACTED]
```

5.9. Empty Password

- Checks that an empty password triggers an error.
- **Screenshot:** Error message for empty password.



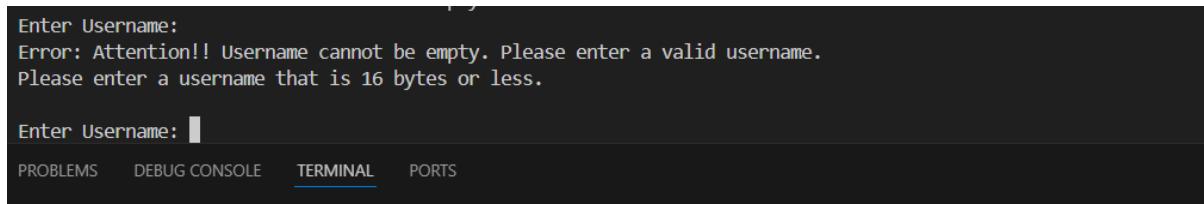
```
Enter Username: userY
Enter Password:
Error: Attention!! Password cannot be empty. Please enter a valid Password.
Enter Username: 
```

The screenshot shows a terminal window with the following text:
Enter Username: userY
Enter Password:
Error: Attention!! Password cannot be empty. Please enter a valid Password.
Enter Username:

Below the terminal window, there is a navigation bar with tabs: PROBLEMS, DEBUG CONSOLE, TERMINAL (which is underlined), and PORTS.

6. Empty Username

- Verifies that empty usernames are rejected.
- **Screenshot:** Error message for empty username.



```
Enter Username:
Error: Attention!! Username cannot be empty. Please enter a valid username.
Please enter a username that is 16 bytes or less.

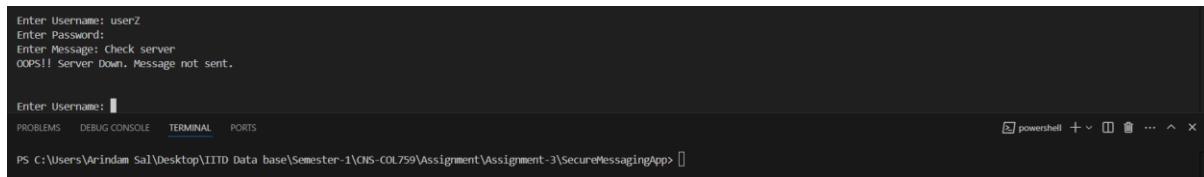
Enter Username: 
```

The screenshot shows a terminal window with the following text:
Enter Username:
Error: Attention!! Username cannot be empty. Please enter a valid username.
Please enter a username that is 16 bytes or less.
Enter Username:

Below the terminal window, there is a navigation bar with tabs: PROBLEMS, DEBUG CONSOLE, TERMINAL (which is underlined), and PORTS.

7. Server is Down

- Client fails to send message
- **Screenshot:** Handling when the server is down.



```
Enter Username: userZ
Enter Password:
Enter Message: Check server
OOPS!! Server Down. Message not sent.

Enter Username: 
```

The screenshot shows a terminal window with the following text:
Enter Username: userZ
Enter Password:
Enter Message: Check server
OOPS!! Server Down. Message not sent.
Enter Username:

Below the terminal window, there is a navigation bar with tabs: PROBLEMS, DEBUG CONSOLE, TERMINAL (which is underlined), and PORTS. In the top right corner, there is a powershell icon and some other icons.

6. Future Scope & Conclusion

6.1. Conclusion: This assignment effectively demonstrates secure message encryption, with additional features to enhance confidentiality, integrity, and replay protection in a simulated client-server setup. By ensuring encrypted storage of messages and adding a master password for server operations, this application lays a foundation for real-world secure communication systems.

6.2. Future Scope: Future improvements could include:

- Implementing a more robust key exchange protocol.
- Enhancing password security with multi-factor authentication.
- Adding message forwarding or deletion timestamps for better traceability.