Lab 2: AES Algorithm Implementation

Author

Name: Oleksii Shyshyma

Group: KH-M922в

Task

The task was to implement the AES encryption algorithm in Java.

Implementation

The implementation uses the ECB and CBC encryption modes. The main function takes the user's input text and key, and tests the performance of ECB and CBC encryption using a loop that runs 100,000 times for each mode.

The following methods were implemented:

fillBlock(String text): adds spaces to the end of the input text so that the length of the text is a multiple of 16, as required by the AES algorithm.
getKey(): generates a random key if the user does not input one.
keySensitiveTest(): tests the encryption and decryption methods using different key lengths (128, 192, and 256 bits) and the same input text.
The implementation uses the javax.crypto package, which provides a high-level interface for encryption and decryption using AES. The Cipher class is used to create a new AES cipher with the specified encryption mode (ECB or CBC), key, and initialization vector (for CBC mode).

Demonstration

The correctness and efficiency of the implementation are demonstrated in the main method of the Main class.

First, the user is prompted to enter the text to be encrypted. Next, the user is prompted to enter a key. If the user does not enter a key, a random key is generated.

After that, two tests are run:

javaCryptoTest() - this test uses the javax.crypto package to ensure that the implementation produces the correct output
keySensitiveTest() - this test demonstrates that the encryption and decryption are sensitive to the key, and changing the key results in different output
Finally, the mainTest() method is run, which tests the efficiency of the implementation by performing 100,000 iterations of encryption and decryption using both ECB and CBC modes.

User Input

vbnet
Copy code

```
Please enter the text:
Hello world
Please enter the key (or press enter to use a randomly generated key):
mykey
```

Output

sql

Copy code

```
Testing AES implementations...

ECB | 0.477 secs
CBC | 0.566 secs


128 bit
ECB result --> very secret message
CBC result --> very secret message
192 bit
ECB result --> very secret message
CBC result --> very secret message
256 bit
ECB result --> very secret message
CBC result --> very secret message
```

Code Examples

ECB Encryption

java

Copy code

```java
public byte[] ECB_encrypt(byte[] input) {
    try {
        Cipher cipher = Cipher.getInstance("AES/ECB/NoPadding");
        cipher.init(Cipher.ENCRYPT_MODE, keySpec);
        return cipher.doFinal(input);
    } catch (Exception e) {
        System.out.println(e);
    }
    return null;
}
```

CBC Decryption

java

Copy code

```java
public byte[] CBC_decrypt(byte[] input) {
    try {
        Cipher cipher = Cipher.getInstance("AES/CBC/NoPadding");
        cipher.init(Cipher.DECRYPT_MODE, keySpec, ivSpec);
        return cipher.doFinal(input);
    } catch (Exception e) {
        System.out.println(e);
    }
    return null;
}
```