CSE-406

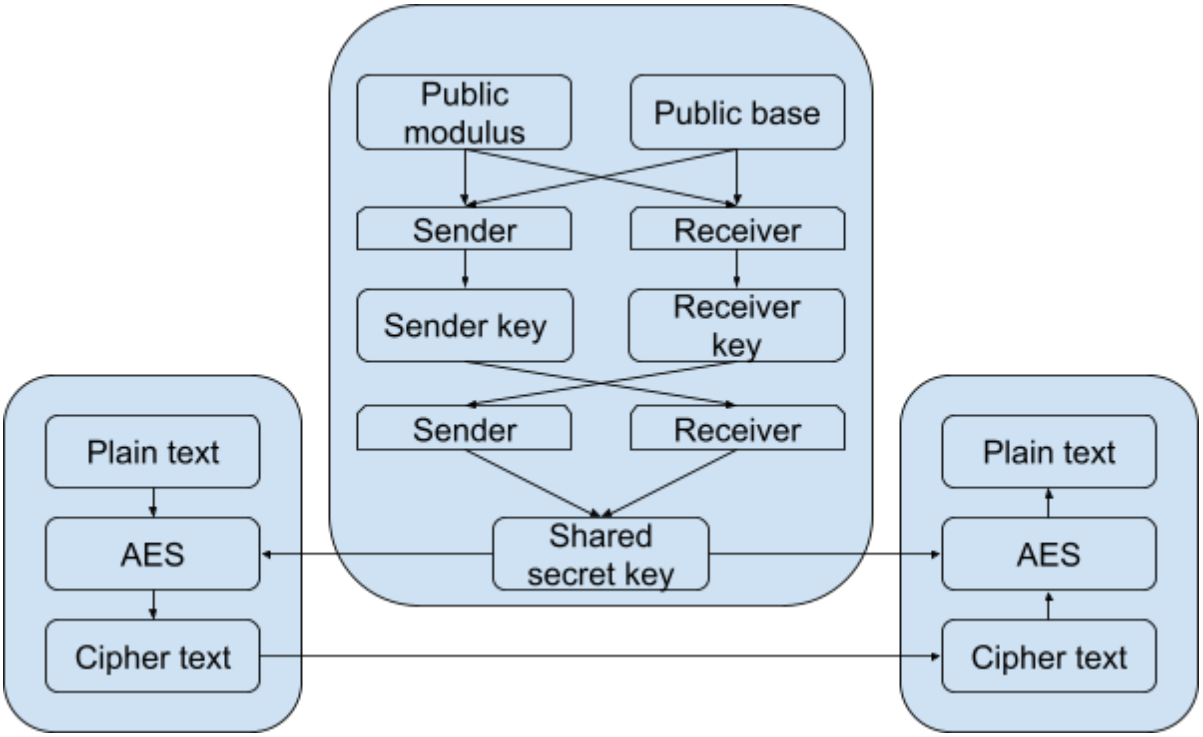
Computer Security Sessional

July 2023

Assignment 01

The goal of this assignment is to implement a cryptosystem that uses a symmetric key for encryption and decryption. The symmetric key is securely shared by the Elliptic Curve Diffie-Hellman key exchange method.

Overview of the cryptosystem



As shown in the figure, both sender and receiver first agree on a shared secret key. Sender uses this key to encrypt the plain text using AES. The AES ciphertext will then be sent through any communication channel. Finally, the shared key is used by the receiver for the AES decryption of the ciphertext.

Overview of AES

Advanced Encryption Standard (AES) is a popular and widely adopted symmetric key encryption algorithm. AES uses repeat cycles or "rounds". There are 10, 12, or 14 rounds for keys of 128, 192, and 256 bits, respectively.

Each round of the Algorithm consists of four steps:

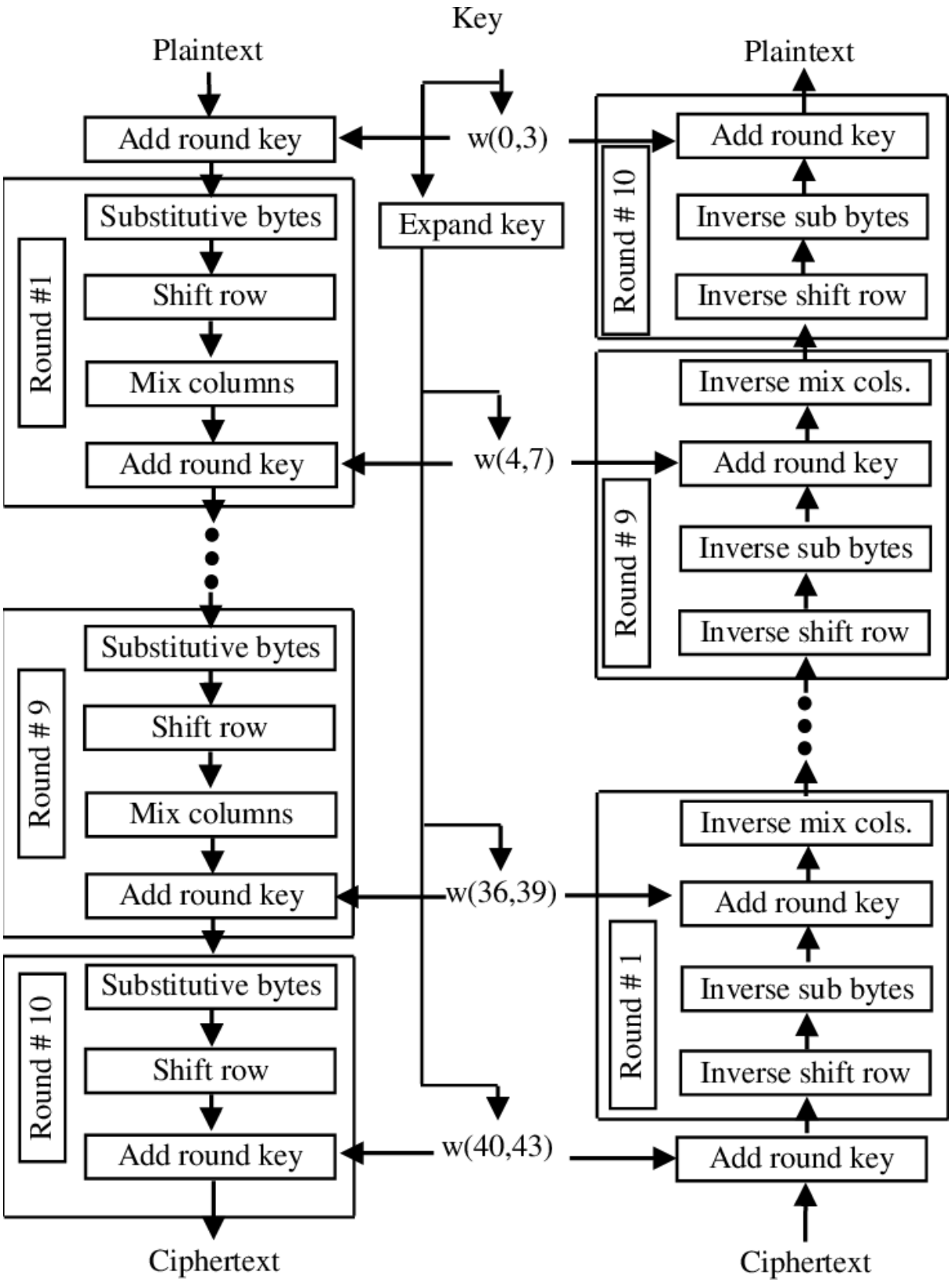
1. **subBytes:** For each byte in the array, use its value as an index into a fixed 256-element lookup table, and replace its value in the state with the byte value stored at that location in the table. You can find the table and the inverse table on this wikipedia page. Also provided in the slide and the code.

2. **shiftRows:** Let R_i denote the i-th row in the state (4x4 matrix). For encryption, shift R_0 left 0 bytes (i.e., no change); shift R_1 left 1 byte; shift R_2 left 2 bytes; shift R_3 left 3 bytes. The shifts are circular. They do not affect the individual byte values themselves. Shift right for decryption.

3. **mixColumns:** For each column of the state, replace the column by its value multiplied by a fixed 4x4 matrix of integers (in a particular Galois Field). This is a complicated step, you can use the BitVector library provided to make your life easier.

4. **addRoundKey:** XOR the state with a 128-bit round key derived from the original key K by a recursive process.

The final round has 3 steps omitting the mixColumns step. The decryption of AES is the inverse of the encryption steps.


Key
Plaintext
Add round key
w(0,3)
Expand key
Round #1
Substitutive bytes
Shift row
Mix columns
Add round key
w(4,7)
Round # 9
Substitutive bytes
Shift row
Mix columns
Add round key
w(36,39)
Round # 10
Substitutive bytes
Shift row
Add round key
w(40,43)
Ciphertext
Plaintext
Round # 10
Add round key
Inverse sub bytes
Inverse shift row
Round # 9
Inverse mix cols.
Add round key
Inverse sub bytes
Inverse shift row
Round # 1
Inverse mix cols.
Add round key
Inverse sub bytes
Inverse shift row
Add round key
Ciphertext

Overview of Elliptic Curve Diffie-Hellman

Elliptic curve Diffie–Hellman key exchange is a mathematical method of securely exchanging cryptographic keys over a public channel and was one of the most secured public-key protocols. It consists of the following steps.

1. The group points G can be defined as an elliptic curve over the finite field. The elliptic curve has the equation $y^2 = x^3 + ax + b$; parameters a and b are agreed as the public parameters. This curve function may generate points $G = (X_g, Y_g)$. Also P, which will be used as the modulus and will be shared over a public medium.

2. Alice chooses a secret and random number key K_a, and performs a scalar multiplication with the generated points ($K_a * G \bmod P = A$), and sends the resulting points $A = (X_A, Y_A)$ to Bob.

3. Bob chooses a secret and random number key K_b , and performs a scalar multiplication with the generated points ($K_b * G \bmod P = B$), and sends the resulting points $B = (X_B, Y_B)$ to Bob.

4. Alice is now able to compute $R = K_b * K_a * G \bmod P$

5. Bob is now able to compute $R = K_a * K_b * G \bmod P$

We can mathematically show that these two values are identical and hence this is the shared secret key.

We will use this shared key in AES. We can only use one co-ordinate (usually the x co-ordinate as the AES key). As AES keys need to be 128, 192 or 256 bits long, we will take the bit length as a parameter, say *k*. Then, the modulus must be at least k bits long. We will also keep both K_a and K_b at least (k) bits long.

Tasks

Task 1. Independent Implementation of (128 bit) AES

1. The Encryption key will be provided by the user as an ASCII string. The key should be 16 characters long, i.e., 128 bits. Your program should also handle keys of other sizes. Implement the key scheduling algorithm as described in this link. In this task you **must** use CBC with space padding for text input.
2. **Encryption:** The program will encrypt a block of text of 128 bits with the keys generated in the previous step. If the text is longer than 128 bits, divide it into 128-bit chunks. If any chunk is smaller than 128 bits, handle it too.
3. **Decryption:** Decrypt the encrypted text blocks and observe if they match with the original text.
4. Report time-related performance in the code.

Sample I/O:

```
Key:
In ASCII: BUET CSE19 Batch
In HEX: 42 55 45 54 20 43 53 45 31 39 20 42 61 74 63 68

Plain Text:
In ASCII: Never Gonna Give you up
In HEX: 4e 65 76 65 72 20 47 6f 6e 6e 61 20 47 69 76 65
20 79 6f 75 20 75 70 00 00 00 00 00 00 00 00 00

Ciphered Text:
In HEX: bd 98 82 5a 07 ac ec 6c 55 f3 39 45 a7 ef c0 73
38 09 1d 8a ed 03 d8 73 97 bb 22 69 bf d2 3f f5
In ASCII: ½□□Z¬ìlUó9E§ïÀs8      ↔□í♥Øs□»"i¿Ò?õ

Deciphered Text:
In HEX: 4e 65 76 65 72 20 47 6f 6e 6e 61 20 47 69 76 65
20 79 6f 75 20 75 70 00 00 00 00 00 00 00 00 00
In ASCII: Never Gonna Give you up

Execution Time Details:
Key Schedule Time:  2.5221 ms
Encryption Time:  592.4183 ms
Decryption Time:  737.9683 ms
```

Task 2. Independent Implementation of Elliptic Curve Diffie-Hellman

1. Generate the shared parameters G, a, b, and P
2. Choose a secret and random number key K_a, and performs a scalar multiplication with the generated points (K_a * G mod P = A),
3. Choose a secret and random number key K_b, and performs a scalar multiplication with the generated points (K_b * G mod P = B),
4. Compute $R = K_a * K_b * G \bmod P$.
5. Report time-related performance in the following format. Take an average of at least 5 trials.

k	**Computation Time For**		
	A	B	shared key R
128			
192			
256			

Implementation of the Whole Cryptosystem

To demonstrate Sender and Receiver, use TCP Socket Programming. To make things easy, please refresh your brain using this guide. Suppose, ALICE is the sender and BOB is the receiver. They will first agree on a shared secret key. For this, ALICE will send *a, b*, *g* and K_a * g (mod *p*) to BOB. BOB, after receiving these, will send K_b * g (mod *p*) to ALICE. Both will then compute the shared secret key, store it and inform each other that they are ready for transmission. Now, ALICE will send the AES encrypted ciphertext (CT) to BOB using the sockets. BOB should be able to decrypt it using the shared secret key.

Bonus Tasks

1. Modify AES to support other types of files with the provision of **proper padding** (image, pdf, etc.) besides text files.
2. Generalize your AES implementation to support 192 and 256 bit keys.
3. Implementation of AES in CTR mode in addition with performance improvement via parallelization (i.e., Thread, GPU, AVX).

Breakdown of Marks

Task	Marks
Independent Implementation of AES with CBC	30
Independent Implementation of Elliptic Curve Diffie-Hellman	35
Implementation of the Whole Cryptosystem using sockets	20
Viva	10
Correct Submission	5
Bonus: AES with other file types	5
Bonus: AES with 192 and 256 bit keys	5
Bonus: Implementation of AES in CTR mode	10

Submission Guidelines

1. Create a directory and name it by your seven-digit student id <1905XXX>.
2. Rename the source file by your student id <1905XXX.py>. If you have multiple files, use this format <1905XXX _f1.py>, <1905XXX_f2.py>, etc. Put the file(s) in the directory created in step 1.
3. Zip the directory and name it by your seven-digit student id <1905XXX.zip>
4. Submit the zip file only.

Deadline

December 01, 2023, Friday, 11:55 pm

Plagiarism

You can easily find the implementation of AES and Elliptic Curve Diffie-Hellman on the Internet. Do not copy from any web source or friend or seniors. The persons involved in such activities will be penalized by **–100%** of the total marks. Also, the persons involved in such activities will be in risk of being **suspended for Two terms** as per BUET's central policy.