CSE-406

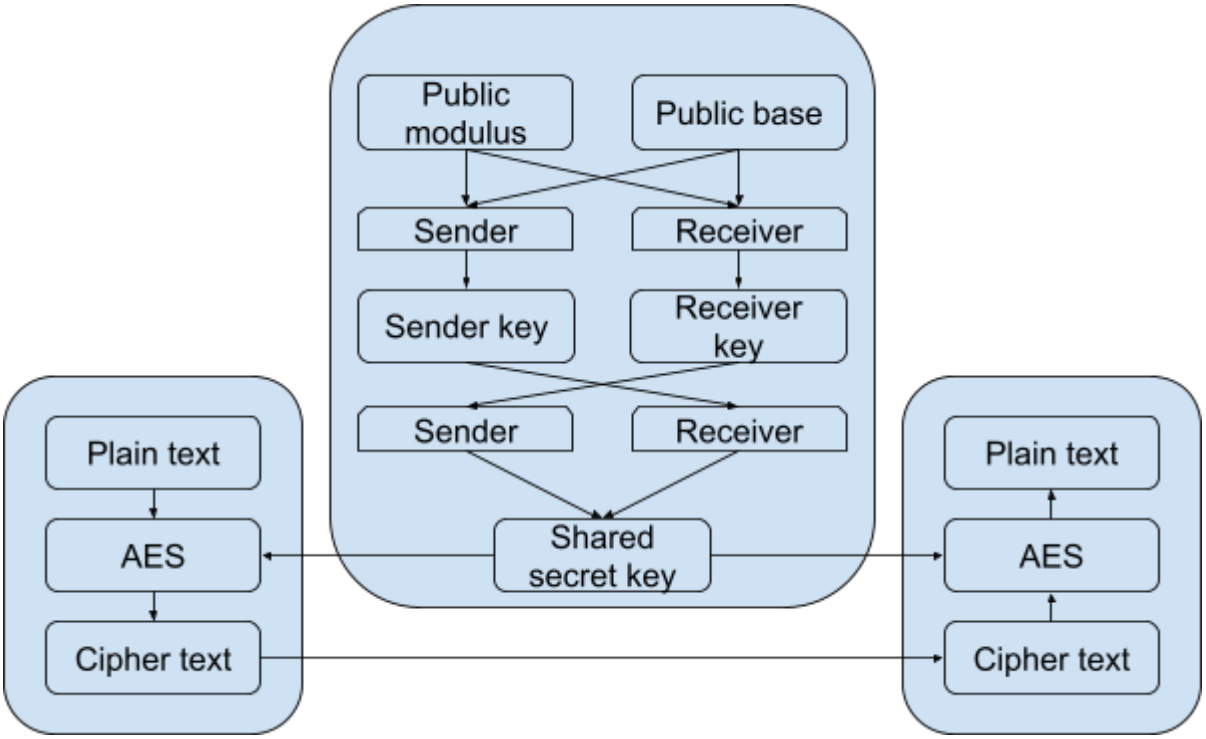
Computer Security Sessional

January 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 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:

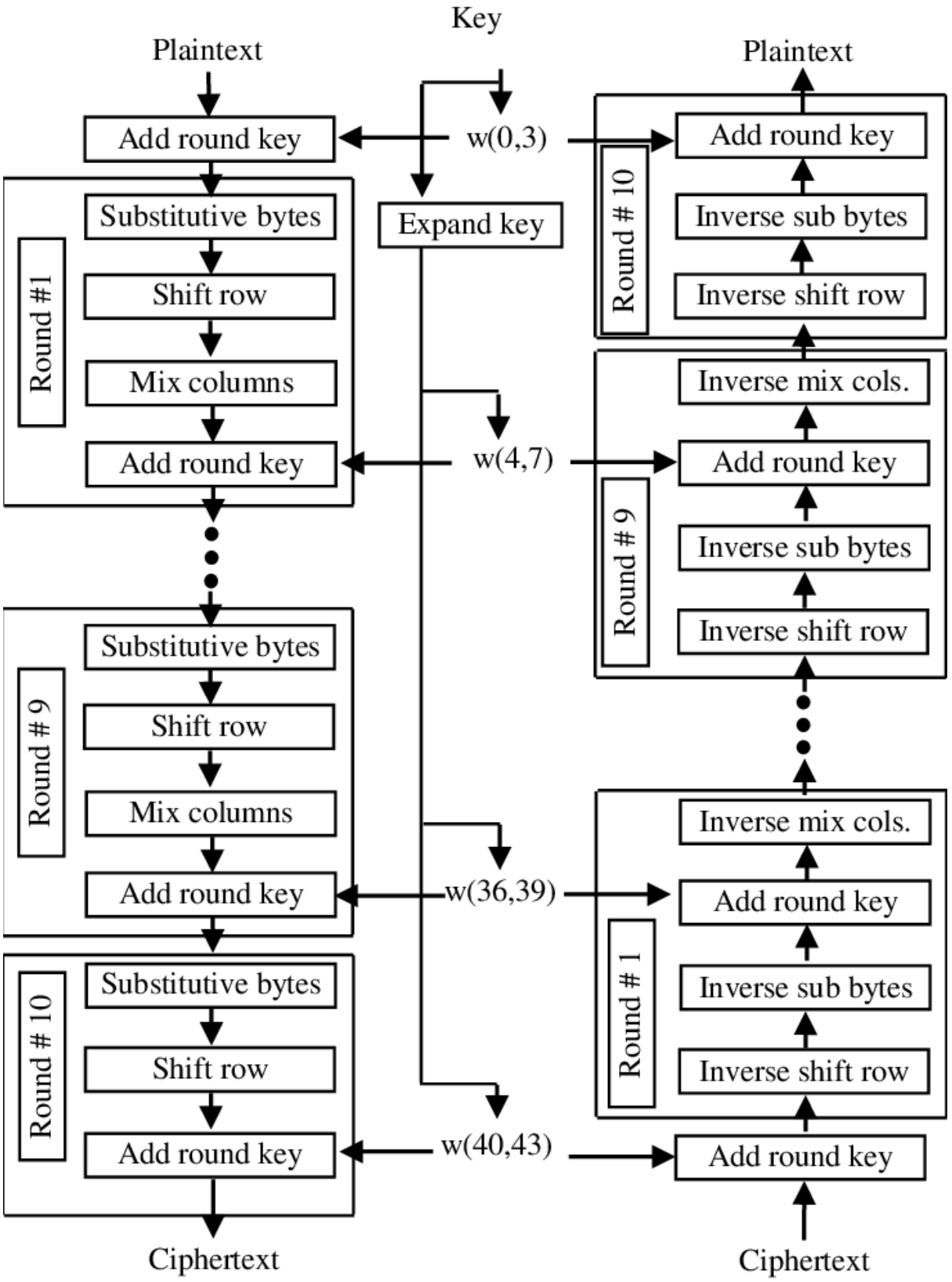
1. **subBytes:** For each byte in the array, use its value as an index into a fixed 256-element lookup table, and replace its value in the state with the byte value stored at that location in the table. You can find the table and the inverse table on this wikipedia page. Also provided in the slide and 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
Plaintext
Add round key
w(0,3)
Add round key
Round # 10
Inverse sub bytes
Inverse shift row
Substitutive bytes
Shift row
Mix columns
Add round key
Round #1
Expand key
Inverse mix cols.
Add round key
Inverse sub bytes
Inverse shift row
Round # 9
w(4,7)
Substitutive bytes
Shift row
Mix columns
Add round key
Round # 9
Inverse mix cols.
Add round key
Inverse sub bytes
Inverse shift row
Round # 1
w(36,39)
Substitutive bytes
Shift row
Add round key
Round # 10
w(40,43)
Add round key
Ciphertext
Ciphertext

Overview of Diffie-Hellman

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

1. **Public modulus and base generation:**
   Both sender and receiver agree on a public modulus p and a public base g. The base g is a generator, i.e., primitive root modulo p. We will take a large prime as the modulus p. Miller-Rabin primality test can be utilized to generate large primes faster. For more info, see this link.

2. **Private and public key generation:**
   Both sender and receiver selects their secret private keys – say a and b. These keys can be any integer but again we will take large primes for them as well. The public keys are then $A = g^a \pmod p$ and $B = g^b \pmod p$ for sender and receiver respectively. Finally, only these public keys are shared through any (might not be secured) transmission channel.

3. **Formation of the shared secret key:**
   Now, both sender and receiver raises the other's public key to the exponent of their own private key taking modulo p. So, sender computes $B^a \pmod p$ and receiver computes $A^b \pmod p$. 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. 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 a and b at least $(k/2)$ bits long.

Tasks

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.
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:

```
Plain Text:
In ASCII: Can They Do This
In HEX: 43616e205468657920446f2054686973

Key:
In ASCII: BUET CSE18 Batch
In HEX: 4255455420435345313820426174636368

Cipher Text:
In HEX: 805130b2867c18302e4b3d211dda6434
In ASCII: ▯Q0²▯|▯0.K=!▯Úd4

Deciphered Text:
In HEX: 43616e205468657920446f2054686973
In ASCII: Can They Do This

Execution time details:
Key Scheduling :  0.0017476081848144531  seconds
Encryption Time :  0.1716303825378418  seconds
Decryption Time :  0.21892333030700684  seconds
```

Independent Implementation of Diffie-Hellman

1. Generate a large prime p which is at least k bits long, take k as parameter.
2. Find a primitive root g for mod p. Take two parameters – *min* and *max* – both less than p. Then g should be in the range [*min*,*max*]. If finding the primitive root seems difficult, take a prime in that range.
3. Take two more primes – a and b – both at least ($k/2$) bits long.
4. Compute $A = g^a \pmod p$ and $B = g^b \pmod p$.
5. Compute $A^b \pmod p$ and $B^a \pmod p$. Verify that they are equal.
6. Report time-related performance in the following format. Take an average of at least 5 trials.

k	**Computation time for**				
	p	g	a or b	A or B	shared key
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 p, g and $g^a \pmod p$ to BOB. BOB, after receiving these, will send $g^b \pmod p$ 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 (image, pdf, etc.) besides text files.
2. Generalize your AES implementation to support 192 and 256 bit keys.
3. Implement RSA for key exchange.
4. Implement RSA for authentication.

Breakdown of Marks

Task	Marks
Independent Implementation of AES	30
Independent Implementation of Diffie-Hellman	40
Implementation of the Whole Cryptosystem using sockets	10
Viva	20
Bonus: AES with other file types	5
Bonus: AES with 192 and 256 bit keys	5
Bonus: Implementation of RSA (key exchange)	10
Bonus: Implementation of RSA (authentication)	5

Deduction of marks:

- AES implementation
    - Unable to handle keys of other lengths: –5
    - Unable to handle texts of other lengths: –10
- Diffie-Hellman implementation
    - External library for verification of large primes: –5
    - Unable to find primitive roots: –10
    - Unable to provide self implemented modular exponent: –5

Submission Guidelines

1. Create a directory and name it by your seven-digit student id <1805YYY>.
2. Rename the source file by your student id <1805YYY.py>. If you have multiple files, use this format <1805YYY_f1.py>, <1805YYY_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 <1805YYY.zip>
4. Submit the zip file only.

Deadline

June 23, 2023, Friday, 11:55 pm

Plagiarism

You can easily find the implementation of AES and 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.