CSE 406

Computer Security Sessional

January 2022

Assignment - 1

The ultimate goal of this assignment is to implement a hybrid cryptosystem using both symmetric and asymmetric cryptography mechanisms. We will use AES as the symmetric algorithm and RSA as the asymmetric cryptography algorithm.

Overview of the hybrid cryptosystem

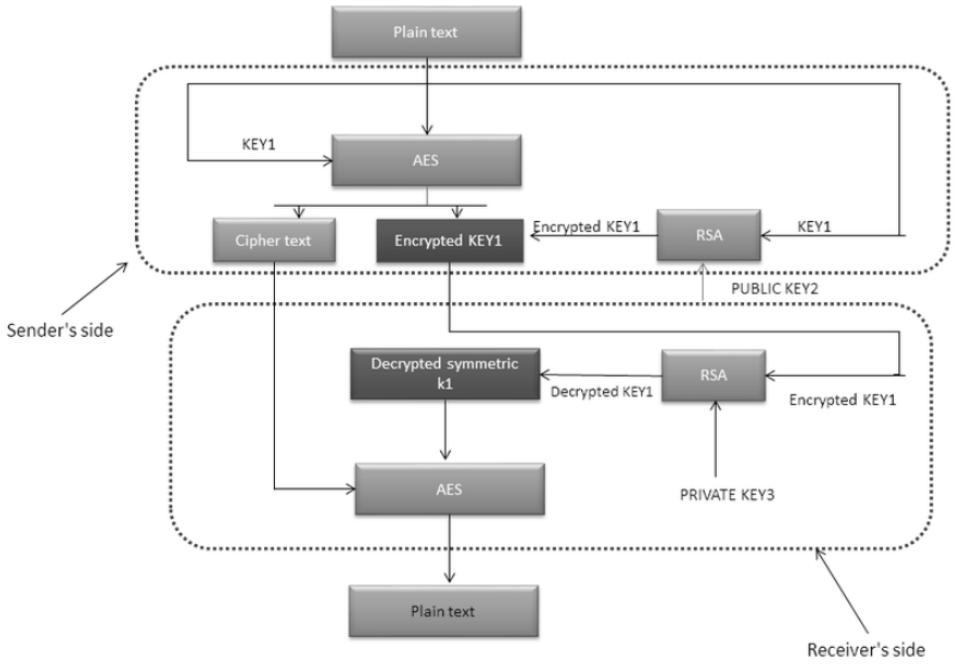


Figure 1: Hybrid Cryptosystem using AES and RSA

As shown in Figure 1, we will at first encrypt the plain text using AES. The key used to encrypt the plain text will be encrypted using RSA. The AES ciphertext and the encrypted AES key will be sent through any communication channel where the key will be at first RSA decrypted to be used 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. Shift R_0 in the state 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. These are circular shifts. They do not affect the individual byte values themselves. Shift left for decryption.

3. **mixColumns:** For each column of the state, replace the column by its value multiplied by a fixed 4 x 4 matrix of integers (in a particular Galois Field). This is a complicated step, you can use 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. The block diagram of AES can be seen in Figure 2.

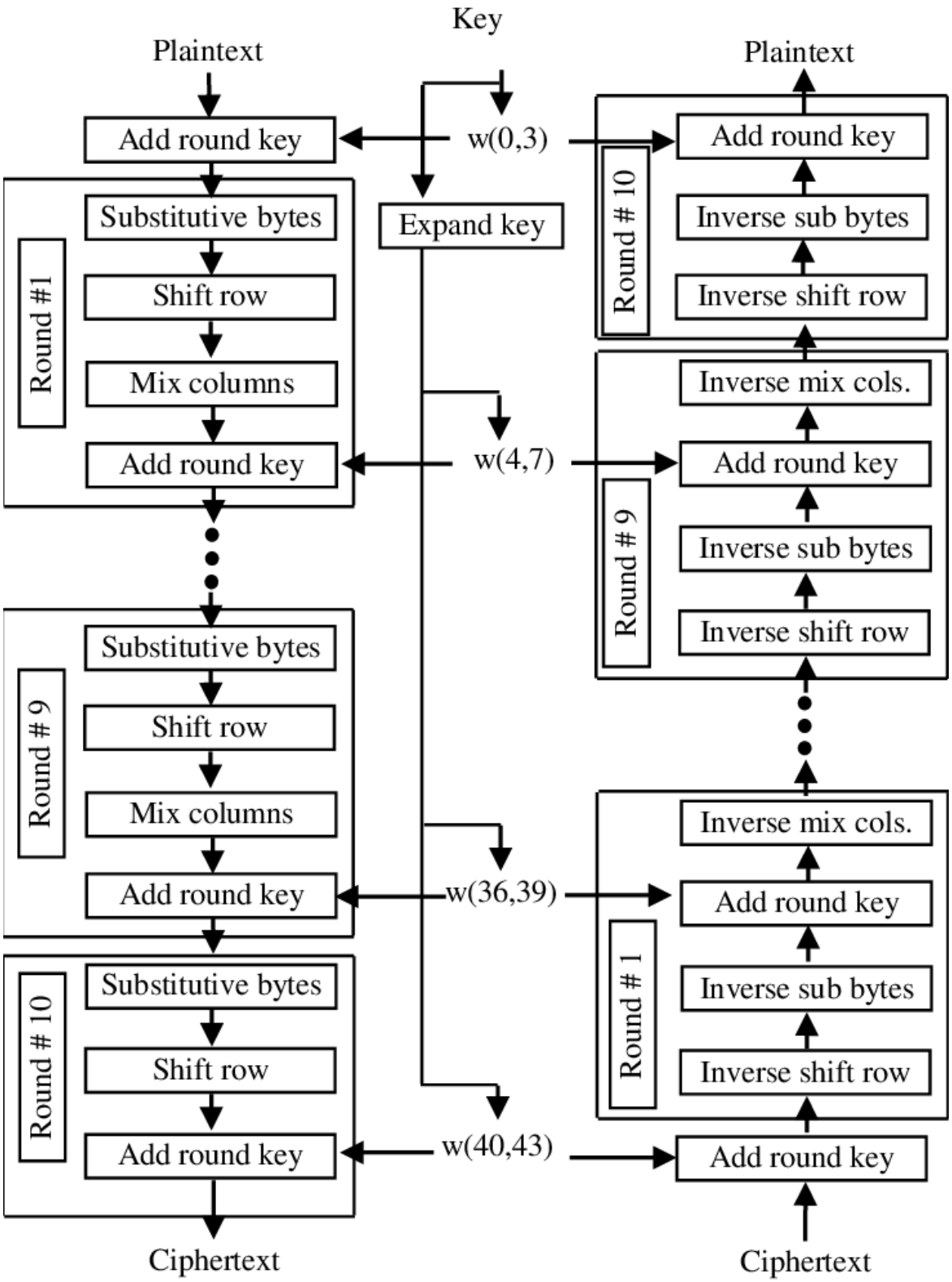


Figure - 2: Block Diagram of AES

Overview of RSA

RSA (Rivest–Shamir–Adleman) is a public-key asymmetric cryptosystem that is widely used for secure data transmission. It is comprised of three steps:

1. Key-Pair Generation:
    a. Choose two distinct prime numbers **p** and **q.** In our case, we will generate two prime numbers each being $\frac{k}{2}$ bits for a given key length **k**. Use Rabin Miller Primality testing to generate faster prime numbers. For more info, see this link.
    b. Calculate n = p * q.
    c. Calculate Φ(n) = (p-1) * (q-1).
    d. Select e such that, e is relatively prime to Φ(n), i.e. (e, Φ(n)) = 1 and 1 < e < Φ(n).
    e. Calculate d = e^{-1} mod Φ(n) or ed = 1 mod Φ(n). You can use extended euclidian algorithm to find the multiplicative inverse. For more info, see this link.
    f. Public key = {e, n}, private key = {d, n}.
2. Encryption:
    a. C = P^e mod n where, P < n where C = Ciphertext, P = Plain text, e = Encryption key and n=block size.
3. Decryption:
    a. P = C^d mod n. Plain text P can be obtained using the given formula. where d = Public key

Tasks

1. **Independent Implementation of AES**
    a. The Encryption key will be provided by the user as an ASCII string. The key will be of length 16. Your program must handle keys of any other size. Implement the key scheduling algorithm as described in this link.
    b. Encryption: The program will encrypt a block of text of 128 bits with the keys generated in the previous step.
    c. Decryption: Decrypt the encrypted text blocks and observe if they match with the original text.
    d. Report time-related performance in the code.

Sample I/O:

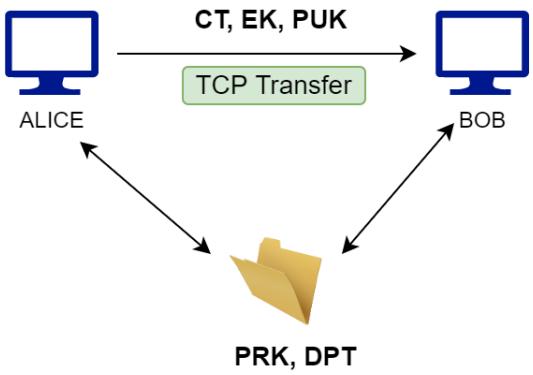
```
Plain Text:
CanTheyDoTheirFest [In ASCII]
43616e54686579446f546865697246657374 [In HEX]

Key:
BUET CSE17 Batch [In ASCII]
42554554204353453137204261746368 [In HEX]

Cipher Text:
27ddffcfcd41984de8423b847004badb [In HEX]
'ÝÿÏÍA▯MèB;▯p▯ºÛ [In ASCII]

Deciphered Text:
43616e54686579446f54686569724665 [In HEX]
CanTheyDoTheirFe In ASCII

Execution Time
Key Scheduling: 0.024462461471557617 seconds
Encryption Time: 0.31696224212646484 seconds
Decryption Time: 1.2586162090301514 seconds
```

2. Independent Implementation of RSA

a. Your program will at first generate the key pairs (both public and private) with an input of K (Take K = 16, 32, 64,128).
b. Take plain text as input and encrypt it character by character.
c. Decrypt the ciphertext and match it with the original plain text.
d. Report time-related performance in the following format:

K	Key-Generation	Encryption	Decryption
16			
32			
64			
128			

3. Implementation of the Hybrid Cryptosystem



Implement the Hybrid Cryptosystem just as in Figure 1. Encrypt the plain text using AES provided the AES key and the key using RSA. To demonstrate Sender and Receiver, we will 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. ALICE will send the AES

encrypted ciphertext (CT), the encrypted key (EK), and the public key (PUK) to BOB using the sockets. Furthermore, ALICE will store the private key (PRK) in a secret folder named “Don’t Open this”. BOB will poll the directory for the key and read the private key from the directory, decrypt the encrypted key using RSA, decrypt the ciphertext using AES decryption, and write the decrypted plain text (DPT) in the same folder. ALICE will finally read the DPT and match it with the original plain text.

4. **Bonus Tasks:**

   a. Modify your program to support not only text but also images, pdf, music, etc.
   b. Generalize your AES implementation to support keys of other lengths (for example - 192/256 bits).
   c. Implementation of some RSA and/or AES heuristics

Breakdown of Marks

Task	Marks
1	25
2	25
3	30
Viva	20
4 (Optional)	20

Submission Guidelines

1. Create a directory and name it by your seven-digit student id <1705YYY>.
2. Rename the source file by your seven-digit student id <1705YYY.py> and put it in the directory created in 1. For multiple files, use this format <1705YYY_f1.py>, <1705YYY_f2.py>, etc.
3. Zip the directory and name it by your seven-digit student id <1705YYY.zip>
4. Submit the zip file.

Deadline - May 28, 2022, 11:55 pm (Strict)

Plagiarism

You can easily find the implementation of AES and RSA 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.