Lab 6: AES

Task 1

We consider AES with 128-bit block length and 128-bit key length. What is the output of the first round of AES if the plaintext consists of 128 ones, and the first subkey also consists of 128 ones? Use a 4 × 4 array to present the state of the algorithm and provide the intermediate states. Ref.

- **State matrix (message after adding the master key) = first subkey =**

$$\begin{bmatrix} FF & FF & FF & FF \\ FF & FF & FF & FF \\ FF & FF & FF & FF \\ FF & FF & FF & FF \end{bmatrix}$$

- **Step 1 - SubBytes (From S-Box)**

$$\begin{bmatrix} 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \end{bmatrix}$$

- **Step 2 - ShiftRows:** has no effect as all rows are identical.

$$\begin{bmatrix} 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \end{bmatrix}$$

- **Step 3 - MixColumns:**

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \bullet \begin{bmatrix} 16 \\ 16 \\ 16 \\ 16 \end{bmatrix} = \begin{bmatrix} (02 \oplus 03 \oplus 01 \oplus 01) \bullet 16 \\ (01 \oplus 02 \oplus 03 \oplus 01) \bullet 16 \\ (01 \oplus 01 \oplus 02 \oplus 03) \bullet 16 \\ (03 \oplus 01 \oplus 01 \oplus 02) \bullet 16 \end{bmatrix} = \begin{bmatrix} 16 \\ 16 \\ 16 \\ 16 \end{bmatrix}$$

  Notes:

  - The previous computation is done for each column, in our case, columns are identical.
  - Symbol ● represents the Galois field multiplication with 128 as the modulus.
  - Symbol $\oplus$ represents the usual XOR operation.

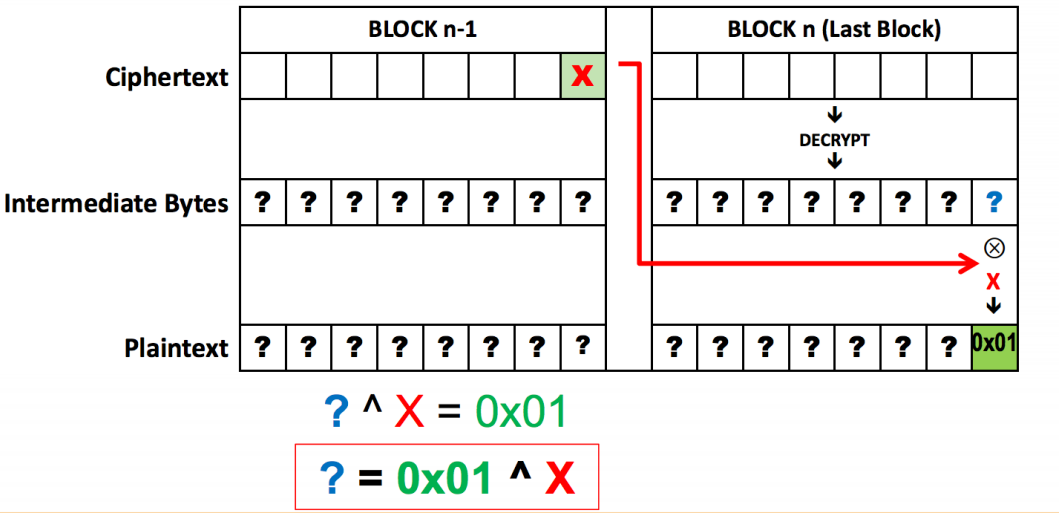
- **Step 4: Add Round Key**

$$\begin{bmatrix} 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \end{bmatrix} \oplus \begin{bmatrix} FF & FF & FF & FF \\ FF & FF & FF & FF \\ FF & FF & FF & FF \\ FF & FF & FF & FF \end{bmatrix} = \begin{bmatrix} E9 & E9 & E9 & E9 \\ E9 & E9 & E9 & E9 \\ E9 & E9 & E9 & E9 \\ E9 & E9 & E9 & E9 \end{bmatrix}$$

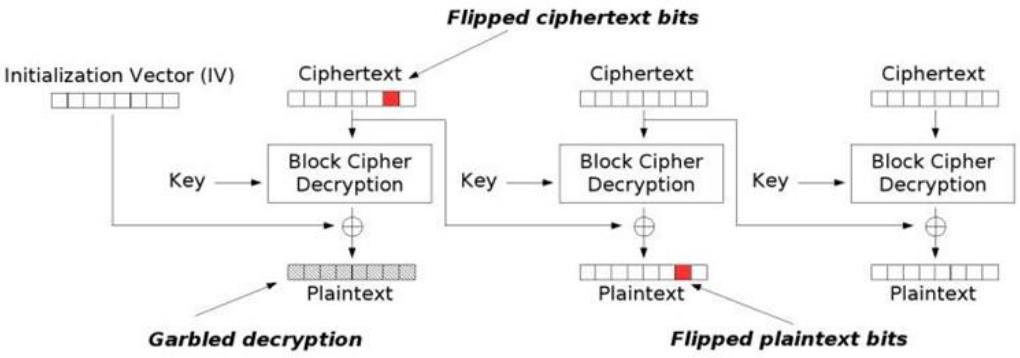
Task 2

Examine attacks called “Padding Oracle Attack” and “CBC Byte Flipping Attack”. Describe the process of their appliance.

- **Padding Oracle Attack:**
  - Allows an attacker to decrypt ciphertext of CBC encryption without knowing the key, given that the server leaks data about whether a given ciphertext has a valid padding after being decrypted or not.
  - The attacker can brute-force a certain byte knowing that only the correct brute-forced value will give valid padding, this will allow them to get an expression for the decrypted text before being XORed, this will later allow decrypting the rest of the text.



- **CBC Byte Flipping Attack:**
  - This attack allows changing a data in the plaintext by corrupting data in the ciphertext, compromising message integrity.
  - It uses the fact that in CBC, the N-th plaintext block is the result of XORing the decrypted N-th ciphertext block with the ciphertext of the N-1 block.
  - By flipping a single bit in that ciphertext, we can get a corresponding flipped bit in the next plaintext block.



Modification attack on CBC

Task 3

In this exercise we check the diffusion properties of AES after a single round. Let:

> W = (w0, w1, w2, w3) = (0x01000000, 0x00000000, 0x00000000, 0x00000000)

be the input in 32-bit chunks to a 128-bit AES. The subkeys for the computation of the result of the first round of AES are W0, ..., W7 with 32-bits each are given by:

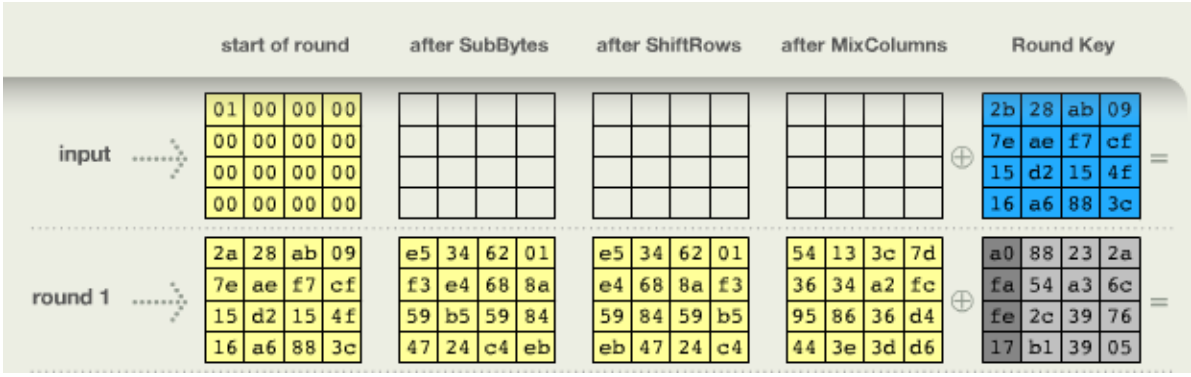
> W0 = (0x2B7E1516)
> W1 = (0x28AED2A6)
> W2 = (0xABF71588)
> W3 = (0x09CF4F3C)
> W4 = (0xA0FAFE17)
> W5 = (0x88542CB1)
> W6 = (0x23A33939)
> W7 = (0x2A6C7605)

For this exercise it might be wise to write a short computer program or use an existing one that shows the intermediate states.

- Compute the output of the first round of AES to the input W and the subkeys W0, . . . ,W7.
- Compute the output of the first round of AES for the case that all input bits are zero (using the same key).
- How many output bits have changed?

For the given input

- **First round calculations**

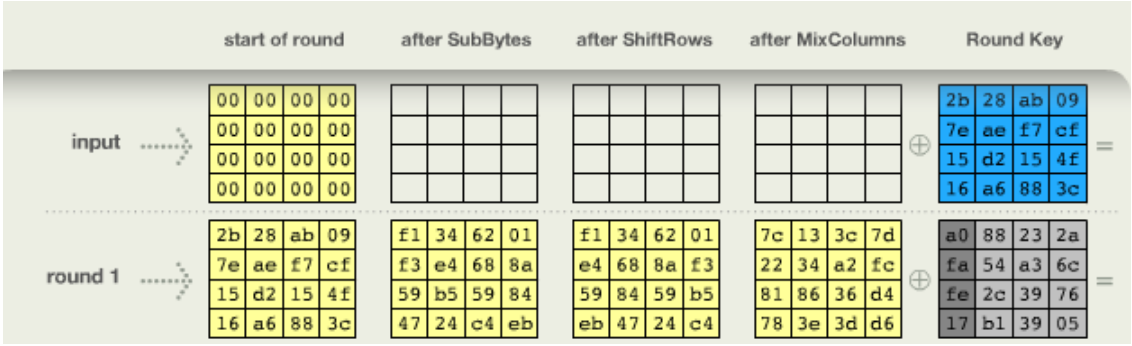


- **Output:**

$$\begin{bmatrix} F4 & 9B & 1F & 57 \\ CC & 60 & 01 & 90 \\ 6B & AA & 0F & A2 \\ 53 & 8F & 04 & D3 \end{bmatrix}$$

If all inputs were zeros

- **First round calculations**



- **Output:**

$$\begin{bmatrix} DC & 9B & 1F & 57 \\ D8 & 60 & 01 & 90 \\ 7F & AA & 0F & A2 \\ 6F & 8F & 04 & D3 \end{bmatrix}$$

- We notice that only the first column of output has changed.
- By XORing the two outputs together we can get the number of bits changed.

```
F4CC6B53 9B60AA8F1F010F045790A2D3
XOR
DCD87F6F 9B60AA8F1F010F045790A2D3
=
11110100110011000110101101010011 ...
XOR
11011100110110000111111101101111 ...
=
00101000000101000000101000011110 0 00...
```

- So, in total, **10 bits** have changed.

Task 4

Derive the bit representation for the round constants RC[8], RC[10] within the key schedule: There are three key schedules known for AES. How do you know by looking at the constants about which key schedule we are talking?

- **Formula for RC[i]**

$$rc_i = \begin{cases} 1 & \text{if } i = 1 \\ 2 \cdot rc_{i-1} & \text{if } i > 1 \text{ and } rc_{i-1} < 80_{16} \\ (2 \cdot rc_{i-1}) \oplus 11\text{B}_{16} & \text{if } i > 1 \text{ and } rc_{i-1} \geq 80_{16} \end{cases}$$

- **Applying the formula we get:**
  - $RC[8] = (80)_{16} = (10000000)_2$
  - $RC[10] = (36)_{16} = (00110110)_2$

- AES uses up to RC[10] for AES-128, up to RC[8] for AES-192, and up to RC[7] for AES-256
- Therefore, we are talking about **AES-128** here.