Project 1 : AES encryption through CBC and CTR modes.

My implementation, key and IV

For both methods, CTR and CBC, I use the get_random_bytes(bytes size) function to generate a random and unique key and/or IV for the length I want.

For CTR, my implementation is :

```
BLOCK_SIZE = 16
KEY_SIZE = 32

# Generating a random unique key
KEY = get_random_bytes(KEY_SIZE)
```

For CBC, my implementation is :

```
BLOCK_SIZE = 16
KEY_SIZE = 32

# Generating a random unique key
KEY = get_random_bytes(KEY_SIZE)
IV = get_random_bytes(BLOCK_SIZE)
```

The IV needs to be the same size as the 16 bytes blocks.

- For the CBC mode, encrypt function :
  When I go through each block to encrypt them, I check if it is the 1st block, then I use the IV. I not, then I use the previous ciphered block, and XOR it, through a XOR function that I created.

- For the CBC mode, decrypt function:
  I use the IV if it is the 1st block or I use the previously decrypted block. Then apply the XOR.

- For the CTR mode, encrypt function:
  It is the same as for CBC but instead of using the previously ciphered block, I use a counter that is an incrementation of the IV (counter is initialized as IV then incremented).
  Then I use it in the XOR function.

- For the CTR mode, decrypt function:
  I also use the IV and the counter the same way as for the encrypt function.

Messages encoding

I chose to encode in ASCII as I saw many people use it. I could have used utf-8 as well.

I encode the plaintext before it is encrypted so that I can apply the encrypt() function on bytes in both modes. Then I encode it to b64 when I return the encrypted text.

I also decode it once the ciphertext is decrypted to get a user-friendly string. I first decode the base 64.

Executions time

I imported the time module to calculate de delta time. I compute the delta time in the main function. The of execution changes every time I execute my code, this is the execution times that I recorded.

Code without the libraries time execution :
EBC:
Execution time: 0.11472678184509277 second

CBC :
Execution time: 0.9516000747680664 second

CTR :
Execution time: 1.0020771026611328 second

➔ CBC is 0.050477027893066406 second faster.

Code with the library time execution :
CBC :
Execution time: 0.13481879234313965 second

CTR :
Execution time: 0.15780305862426758 second

➔ CBC is 0.02125382423400879 second faster.

Globally, EBC is the fastest then comes CBC then CTR. The more complex the mode is, the more time it takes to encrypt data.