

```
1  from Crypto.Random import get_random_bytes
2  from Crypto.Cipher import DES, AES
3  from Crypto.Util.Padding import pad,unpad
4  from Crypto.Util import Counter
5
6  class AES_CIPHER:
7
8      BLOCK_SIZE_AES = 16 # AES: Bloque de 128 bits
9
10     def __init__(self, key):
11         """Inicializa las variables locales"""
12         self.key=key
13     def cifrar(self, cadena, IV):
14         """Cifra el parámetro cadena (de tipo String) con una IV específica, y
15             devuelve el texto cifrado binario"""
16         # Crea un mecanismo de cifrado AES en modo CTR
17         cadena = cadena.encode("utf-8")
18         cypher = AES.new(self.key, AES.MODE_CTR, nonce=IV)
19
20         # Cifro haciendo que el bloque sea múltiplo del tamaño del bloque
21         cyphertext = cypher.encrypt(pad(cadena, self.BLOCK_SIZE_AES))
22         return cyphertext
23
24     def descifrar(self, cifrado, IV):
25         """Descifra el parámetro cifrado (de tipo binario) con una IV específica, y
26             devuelve la cadena en claro de tipo String"""
27         # Creo un mecanismo de descifrado AES en modo CTR
28         decipher_aes = AES.new(self.key, AES.MODE_CTR, nonce=IV)
29
30         # Descifro, elimino el padding y recupero la cadena
31         new_data = unpad(decipher_aes.decrypt(cifrado),self.BLOCK_SIZE_AES).decode("utf-8",
32 "ignore")
33         return new_data
34
35 key = get_random_bytes(16) # Clave aleatoria de 128 bits
36 IV = get_random_bytes(8)  # IV aleatorio de 64 bits (tamaño del bloque / 2)
37 datos = "Hola Mundo con AES en modo CTR"
38 print(datos)
39 d = AES_CIPHER(key)
40 cifrado = d.cifrar(datos, IV)
41 print("Texto cifrado")
42 print(cifrado)
43 descifrado = d.descifrar(cifrado, IV)
44 print("Texto descifrado")
45 print(descifrado)
46 #####
47 #####
48 #####
49
```