```python
from Crypto.Random import get_random_bytes
from Crypto.Cipher import AES, AES
from Crypto.Util.Padding import pad,unpad
from Crypto.Util import Counter

# Datos necesarios
key = get_random_bytes(16) # Clave aleatoria de 128 bits
IV = get_random_bytes(16) # IV aleatorio de 128 bits para CBC
BLOCK_SIZE_AES = 16 # Bloque de 128 bits
data1 = "Hola amigos de la seguridad".encode("utf-8") # Datos a cifrar
data2 = "Hola amigas de la seguridad".encode("utf-8") # Datos a cifrar
print(data1)
print(data2, "\n")



# CIFRADO ###############################################################################
# Creamos un mecanismo de cifrado AES en modo CBC con un vector de inicialización IV
cipher = AES.new(key, AES.MODE_CBC, IV)

# Ciframos, haciendo que la variable “data1” sea múltiplo del tamaño de bloque
ciphertext1 = cipher.encrypt(pad(data1,BLOCK_SIZE_AES))
print(ciphertext1)
ciphertext2 = cipher.encrypt(pad(data2,BLOCK_SIZE_AES))
print(ciphertext2,"\n")


# AESCIFRADO ############################################################################
# Creamos un mecanismo de (AES)cifrado AES en modo CBC con un vector de inicialización IV para CBC
# Ambos, cifrado y AEScifrado, se crean de la misma forma
decipher_aes = AES.new(key, AES.MODE_CBC, IV)

# AESciframos, eliminamos el padding, y recuperamos la cadena
new_data1 = unpad(decipher_aes.decrypt(ciphertext1), BLOCK_SIZE_AES).decode("utf-8", "ignore")
new_data2 = unpad(decipher_aes.decrypt(ciphertext2), BLOCK_SIZE_AES).decode("utf-8", "ignore")

# Imprimimos los datos AEScifrados
print(new_data1)
print(new_data2)
```