CRYPTO - An Encrypted Secret

Code Given

```python
import os
from Crypto.Cipher import AES
from secret import FLAG

secret_key = os.urandom(16)
iv = os.urandom(8)

# Super secure AES encryption
def encrypt(plaintext):
        cipher = AES.new(secret_key, AES.MODE_CTR, nonce=iv)
        return cipher.encrypt(plaintext).hex()

# Encrypt my favourite lyrics
lyrics = b"""Never gonna give you up
Never gonna let you down
Never gonna run around and desert you
Never gonna make you cry
Never gonna say goodbye
Never gonna tell a lie and hurt you"""

print(f"Encrypted lyrics: {encrypt(lyrics)}")
print(f"Encrypted flag: {encrypt(FLAG)}")

# Output
# Encrypted lyrics:
134d0c946b93c5da8d22e31cadbbc3e27efd8c37542626fc6c79b0ef8b6a81e1a8e5a068258cf32a603
0ebe7d925140acc7335685c43946c266051515394232f195dab794f5dd87faa7bc9c0238ec0e2ee9b87
950ff271d53bc2e4aa1dea5afb0151cc86f99802f32709e55bb69189c89999d8c0dca08942c27d70532
d4ad75c3bb3af90bcde9aab8decfd26dd30d7061b7e98cbf5621bf4fdaf34ee1092bd9f55caa7a85148
9ddf480ef7
# Encrypted flag:
3e511e946ac8c381da2ab70ffdb787e138e580764d6061921024f5b2cd78d0ecf6e8f078788dfa
```

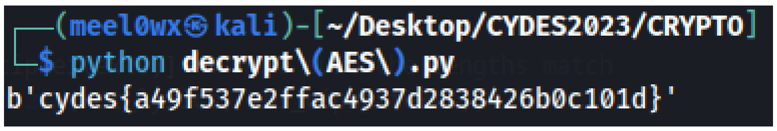
From the code that we get, we can see that it’s using AES, CTR mode to encrypt the data. AES is a symmetric encryption algorithm, meaning it uses the same key for both encryption and decryption. In the CTR mode, a nonce (Number Once, essentially an initialization vector or IV) and a counter is used to generate a new key stream for every block. However, there's an issue here that could potentially lead to breaking the encryption because it’s using the same nonce

(IV) for both the lyrics and FLAG. In the CTR mode, reusing a nonce with the same key can lead to vulnerabilities. If we have two ciphertexts that are encrypted with the same key and the same nonce, we can XOR them together to get the XOR of the two plaintexts. Here is how we can decrypt it

Decrypt Script

```python
import os
from Crypto.Cipher import AES

# Convert hex to bytes
encrypted_lyrics = 
bytes.fromhex('134d0c946b93c5da8d22e31cadbbc3e27efd8c37542626fc6c79b0ef8b6a81e1a8e5
a068258cf32a6030ebe7d925140acc7335685c43946c266051515394232f195dab794f5dd87faa7bc9c
0238ec0e2ee9b87950ff271d53bc2e4aa1dea5afb0151cc86f99802f32709e55bb69189c89999d8c0dc
a08942c27d70532d4ad75c3bb3af90bcde9aab8decfd26dd30d7061b7e98cbf5621bf4fdaf34ee1092b
d9f55caa7a851489ddf480ef7')
encrypted_flag = 
bytes.fromhex('3e511e946ac8c381da2ab70ffdb787e138e580764d6061921024f5b2cd78d0ecf6e8
f078788dfa')

# Encrypt my favourite lyrics
lyrics = b"""Never gonna give you up
Never gonna let you down
Never gonna run around and desert you
Never gonna make you cry
Never gonna say goodbye
Never gonna tell a lie and hurt you"""

# XOR the two ciphertexts
xor_ciphertexts = bytes(a ^ b for a, b in zip(encrypted_lyrics, encrypted_flag))

# XOR with lyrics to get FLAG
xor_lyrics = lyrics[:len(xor_ciphertexts)]  # Make sure the lengths match
FLAG = bytes(a ^ b for a, b in zip(xor_lyrics, xor_ciphertexts))

print(FLAG)
```

Output



```
┌──(meel0wx㉿kali)-[~/Desktop/CYDES2023/CRYPTO]
└─$ python decrypt\(AES\).py
b'cydes{a49f537e2ffac4937d2838426b0c101d}'
```