

# Ramadan CTF WriteUp

By: cicakberlari

## Eventually (Cryptography)

- In this challenge, we got 2 files from extracted **evl.7z** file. First file is python file that contains code to encrypt the flag and the second file is txt file.

### 1. Challenge.py file

```
Challenge.py 3 X
ev2 > Challenge.py > ...
1 from Cryptodome.Cipher import AES
2 from Cryptodome.Util.Padding import pad, unpad
3 import hashlib
4 import os
5 from secret import shared_secret
6
7
8 FLAG = b'Flag{00000000000000000000000000000000}'
9
10 def encrypt_flag(shared_secret: int):
11     sha1 = hashlib.sha1()
12     sha1.update(str(shared_secret).encode('ascii'))
13     key = sha1.digest()[:16]
14     iv = os.urandom(16)
15     cipher = AES.new(key, AES.MODE_CBC, iv)
16     ciphertext = cipher.encrypt(pad(FLAG, 16))
17     data = {}
18     data['iv'] = iv.hex()
19     data['encrypted_flag'] = ciphertext.hex()
20     return data
21
22
23 print(encrypt_flag(shared_secret))
24 #('iv': '96a7f5c23a3344689b465a821ea0cf39', 'encrypted_flag': '07ba868266c6c1f8a5600d8fafd984169222d32bdb786f03f06c327c02651be2a975cbaa20e8f89f17ce9ca2579a1a74')
25
```

### 2. File.txt

```
file.txt X
ev2 > file.txt
1 g = 3
2 p =
241831242635703258855207602219756607485695045802459942654116941958108831682612228890093858261341614673227141477904012196503648957050582631942730786080092230627347453410734066962460145893616597740410271
69249453200378729434170325843778659198143763193776859869524088940195577346119843545301547043747207749969763750084308926339295559968882457872412993810129138294592999947926365264059284647209730384947211681
434464714438488520940127459844288059336526896320919633919
3 a =
1122187411395429088885643595343734240130162497729196269223790757199033448352887751380927262561051206115906173760854728855866287968508668429962448174286501692406500055526797783014474036446797720655591478
1236397216033885882207640219686011643468275165718132888489024688846101943642459655423609111976363316080620471528236879737944217503462265615774743189863758784409788192383460779888641161568318746958174777
72477121232820827728424890845769152726027520772901423784
4 b =
197393425814907036907657727149200859802493419256509515552190494112904362171906051908249347873627922078500978353181450766138511122063929258040196339624065676869119737979175531770768861808581110311903548
5674240392644856613389952129078033000241654699770994942847228318456539853927914082647120912935082749471324084023198121104626411438045770633585919066824069468026116021060950689184279386829767261962592400
1403035676872189455767944077542198004499486164431451944
5
12129724605220753447833756660700537760331108332735677863862813666578639518899293226399921252049650315636129053951452368544433347745559822048578957163832157054989703953795266987614689321472006505136260
28263449605756611895255213431429792650448680940566754924112559738717300646014537975998627291919067598887389420895685177333103974784031245522135458991072698281920342199272973829645282036553759182547255
9989488215839368811962969067647494762616719047466973581
6
```

- 3. By doing some research on google, we know that file.txt are some variables that related to Diffie-Hellman key exchange protocol.

4. We code a simple python code to get the shared secret(to use inside *Challenge.py*).

```
Shared.py X
evl > Shared.py > ...
1 g = 3
2 p =
241031242635703258855207022197566074856958548502459942654116941958188816826122289809385826134161467327141477904012196503648957058582613942730786885089223062734745340734066962460145893616597740410271
69249451208378729434170325843778659198143761937768598695248894019557734611984354538154784747207749969763750804308926339295559668824578724129938101291302945929999479263652640592846472097308384947211681
434464714430488520940127459044288859336526896320919633919
3 a =
11221074113954298888564359534373424013816249729319626922379075199033448352887751389927262561051206115906173768547288558662879685886684299624481742865016924065000552679778301447403644679772065591478
123639721603380588220764021968601164346827516571813288848902468884610194364245965542360911197636316080620471928236879737944217503462265615774743189863758784409788192383460779088641161568318746958174777
72477121232080272842408045769152726027520772901423784
4 b =
1973934258149070369878577271492088598024934192565095155521904941129043621719060519082493478733627922878580978351814507661385111220639329358048196339626065676869110737979175531770768861808581110311903548
56242403926448456613309952219070813000241654609770949420472283184565398539279148026471209129358027494713240840231981211042641143804577063358591906682406946802611602106095068918427936829767261962592400
1403035676872189455767944077542198064499486164431451944
5
1212972460522075344783375666070053776033110833273567863862813666578639518899293226399921252049650315636129053951452368544433347745559822048578957163832157054989703953795266987614689321472006505136260
2826344960579566118952552134314297926504084094056754924112559738717300646014537975998627219199067598887389420895685177333103974784011245522135458991072698281920342199272973829645282036553759182547255
9909840821593936011962969067647494762616719047466973501
6
7 shared_secret = pow(A, b, p)
8 print(shared_secret)
```

5. Run the *Shared.py* and we will get the shared secret.

```
PS D:\Cybersecurity\CTF\RamadanCTF\evl> python .\Shared.py
504353011130463311841181943875464043231412685807038060628611295335507777072593376801989742833898756528561413566575025110523306301059942489339235482470041389179915182503716079090726091960066394164202725682615390128122662
015735083001586309124553082057621703226429300419824256494867427207732269658157662507115434790172778535020070092929122731552091516457943164322807667640291518433369993966897015028967365703025958639929458206668294398571463
7819933196409531946
```

6. Now moving to python file *Challenge.py* we modify some of the code :-

- Change the import from Cryptodome to Crypto
- Add some import from hashlib and os
- Add shared\_secret variables that we got from *Shared.py*
- Add some function to decrypt the flag

Here the modified code :

```
Challenge.py X
evl > Challenge.py > decrypt_flag
1 from Crypto.Cipher import AES
2 from Crypto.Util.Padding import pad, unpad
3 import hashlib
4 import os
5 shared_secret =
5043530111304633118411819438754640432314126858070380606286112953355077770725933768019897428338987565285614135665750251105233063010599424893392354824700413891799151825037160790907260919600663941642027256
82615390128122662015735083001586309124553082057621703226429300419824256494867427207732269658157662507115434790172778535020070092929122731552091516457943164322807667640291518433369993966897015028967365703025958639929458206668294398571463
838259586399294582066682943985714637819933196409531946
6
7
8 FLAG = b'Flag{XXXXXXXXXXXXXXXXXXXXXXXXXXXX}'
9
10 def encrypt_flag(shared_secret: int):
11     sha1 = hashlib.sha1()
12     sha1.update(str(shared_secret).encode('ascii'))
13     key = sha1.digest()[:16]
14     iv = os.urandom(16)
15     cipher = AES.new(key, AES.MODE_CBC, iv)
16     ciphertext = cipher.encrypt(pad(FLAG, 16))
17     data = {}
18     data['iv'] = iv.hex()
19     data['encrypted_flag'] = ciphertext.hex()
20     return data
21
22
23 def decrypt_flag(shared_secret: int, iv: str, encrypted_flag: str):
24     sha1 = hashlib.sha1()
25     sha1.update(str(shared_secret).encode('ascii'))
26     key = sha1.digest()[:16]
27     iv = bytes.fromhex(iv)
28     cipher = AES.new(key, AES.MODE_CBC, iv)
29     ciphertext = bytes.fromhex(encrypted_flag)
30     plaintext = unpad(cipher.decrypt(ciphertext), 16)
31     return plaintext
32
33 # Replace these values with the actual iv and encrypted_flag from the output of encrypt_flag
34 iv = '96a7f5c23a334489946a821eabc939'
35 encrypted_flag = '07ba868266cc1f0a5600d8faf098416922d32bdb786f03f0ec327c02651be2a975cbaa20e8f89f17ce9ca2579a1a74'
36
37 print(decrypt_flag(shared_secret, iv, encrypted_flag))
38
39 # print(encrypt_flag(shared_secret))
40 # {'iv': '96a7f5c23a334489946a821eabc939', 'encrypted_flag': '07ba868266cc1f0a5600d8faf098416922d32bdb786f03f0ec327c02651be2a975cbaa20e8f89f17ce9ca2579a1a74'}
41
```

7. Now we just run the file and then we got the decrypted flag.

```
PS D:\Cybersecurity\CTF\RamadanCTF\evl> python .\Challenge.py
b'Flag{D1ffi3H31l4n_S4r1n9_Secr375}'
```