[ACTF新生赛2020]crypto-aes

原创

2er0!=O 于 2021-07-26 21:31:07 发布 109 收藏

分类专栏： wp Crypto aes 文章标签： buu

本文链接：https://blog.csdn.net/m0_52727862/article/details/119118888


wp 同时被 3 个专栏收录

54 篇文章 0 订阅

订阅专栏


Crypto

48 篇文章 1 订阅

订阅专栏


aes

1 篇文章 0 订阅

订阅专栏

[ACTF新生赛2020]crypto-aes

附件：

名称	压缩后大小	原始大小	类型	修改日期
..				
._aes.py	212	212	JetBrains PyCharm Co...	2020/3/5
._output	212	212	_OUTPUT 文件	2020/3/5
aes.py	350	350	JetBrains PyCharm Co...	2020/3/5
output	179	179		2020/3/5

aes.py:

```
from Cryptodome.Cipher import AES
import os
import gmpy2
from flag import FLAG
from Cryptodome.Util.number import *

def main():
    key=os.urandom(2)*16
    iv=os.urandom(16)
    print(bytes_to_long(key)^bytes_to_long(iv))
    aes=AES.new(key,AES.MODE_CBC,iv)
    enc_flag = aes.encrypt(FLAG)
    print(enc_flag)
if __name__=="__main__":
    main()
```

output:

```
91144196586662942563895769614300232343026691029427747065707381728622849079757
b'\x8c-\xcd\xde\xa7\xe9\x7f.b\x8aKs\xf1\xba\xc75\xc4d\x13\x07\xac\xa4&\xd6\x91\xfe\xf3\x14\x10|\xf8p'
```

关键函数和方法

urandom:

```
语法  os.urandom（size）
参数：
 size:字符串随机字节的大小
 返回值：该方法返回一个字符串，该字符串表示适合加密使用的随机字节。
例 os.urandom(1)
输出：b'\x91'
二进制：10010001 （8bits）
```

AES.new(key, mode, *args, **kwargs):

```
param key(参数密钥):
在对称密码中使用的秘密密钥。

它必须为16、24或32个字节长（分别用于AES-128， AES-192或AES-256）。
```

```
mode（模式）
模式（支持的MODE_*常量之一）-用于加密或解密的链接模式。
学习链接：https://pycryptodome.readthedocs.io/en/latest/src/cipher/aes.html
```

```
Keyword Arguments（关键字参数）：
IV（字节，字节组，memoryview） - （只适用于MODE_CBC，MODE_CFB，MODE_OFB，和MODE_OPENPGP模式）。

用于加密或解密的初始化向量。

对于MODE_CBC，MODE_CFB和MODE_OFB它必须是16个字节。
```

解题思路：

解密 flag 我们需要获取到 key 和 iv 的值，由条件：
key=os.urandom(2)*16
iv=os.urandom(16)
可知：key是32bytes,256bits ；iv是16bytes ,128bits

key^iv，那么只有 iv 与 key的低128位相异或，所以key的高128位是固定不变的。所以输出结果的高128bits,就是key的高128bits,进而可以得到key的所有值256bits。

之后key的低128bits，与输出结果的低128bits 相异或，所得结果就是 iv的值了

key,iv得到后直接aes.decrypt()解密就ok了

```python
from Crypto.Cipher import AES
import os
from gmpy2 import*
from Crypto.Util.number import*

xor = 91144196586662942563895769614300232343026691029427747065707381728622849079757
enc_flag = b'\x8c-\xcd\xde\xa7\xe9\x7f.b\x8aKs\xf1\xba\xc75\xc4d\x13\x07\xac\xa4&\xd6\x91\xfe\xf3\x14\x10|\xf8p'
out = long_to_bytes(xor)
key = out[:16]*2
# print(key)
iv = bytes_to_long(key[16:])^bytes_to_long(out[16:])
# print(iv)
iv = long_to_bytes(iv)
# print(iv)
aes = AES.new(key,AES.MODE_CBC,iv)
flag = aes.decrypt(enc_flag)
print(flag)
```

运行得：

```
b'actf{W0W_y0u_can_so1v3_AES_now!}'

进程已结束，退出代码为 0
```



```
actf{W0W_y0u_can_so1v3_AES_now!}
```

另一个解密脚本：

```
key_iv=911441965866629425638957696143002323430266910294277470657073817286228490797 57
flag_encrypt=b'\x8c-\xcd\xde\xa7\xe9\x7f.b\x8aKs\xf1\xba\xc75\xc4d\x13\x07\xac\xa4&\xd6\x91\xfe\xf3\x14\x10|\xf8
p'
print(hex(key_iv))
key=hex(key_iv)[2:6]*16
iv=key_iv^eval('0x'+key)
import Crypto.Util.number
iv=Crypto.Util.number.long_to_bytes(iv)
key=Crypto.Util.number.long_to_bytes(eval('0x'+key))
import Crypto.Cipher.AES
decrypt=Crypto.Cipher.AES.new(key,Crypto.Cipher.AES.MODE_CBC,iv)
print(decrypt.decrypt(flag_encrypt))
```

运行得：

```
0xc981c981c981c981c981c981c981c9814eed98e380b1356763849930c850b9cd
b'actf{W0W_y0u_can_so1v3_AES_now!}'

进程已结束，退出代码为 0
```