# give me your passport

*发表于 2021-01-04 分类于 Challenge， 2019， 湖湘杯， Crypto*
*Challenge | 2019 | 湖湘杯 | Crypto | give me your passport*

*点击此处获得更好的阅读体验*

---

## *题目考点*

- *AES字节翻转*

## *解题思路*

*本题提供源码，以交互的方式进行。开始交互时题目会在后台生成8-12个随机字符组成的name,然后生成随机的加密初始向量iv，之后利用未知的key进行CBC方式的AES对name加密，返回给我们iv和name的加密结果的hex编码。用户输入数据，服务器会以输入的前16字节作为新的iv对其余数据进行解密，如果解密出来的name是'Admin'则给出flag。采用CBC的加密方式，可以想到的攻击方式有字节翻转，根据加密原理已知`(pad(name) ^iv) = deAES(out)`要使`(pad(payload) ^newiv) = deAES(out)`可得`newiv = pad(name) ^iv ^pad(payload)`其中name未知，问题的关键在于求name。关键问题代码在此：*

```
def check_pad(s, block_size):
    assert len(s) % block_size == 0
    assert ord(s[-1]) <= block_size
    for i in range(ord(s[-1])):
        assert s[-1-i] == s[-1]#mark
    return s[:-1-i]
```

*以及主程序的一段判断：*

```
try:
    print "padplain_text:" + plain_text
    plain_text = check_pad(plain_text, AES.block_size)
    print "plain_text:"+plain_text
except:
    print "padding error"
    sys.stdout.flush()
    return
if plain_text == 'Admin':
    print "Welcome admin, your flag is %s" % FLAG
    sys.stdout.flush()
    return
else:
    print "YOU. SHALL. NOT. PASS!"
    sys.stdout.flush()
    return
```

*问题在于check_pad中要求pad的每一位都要一样，在解密不为admin的情况下也会又两种情况，一种通过check_pad返回`YOU. SHALL. NOT. PASS`，一种不通过返回`padding error`,可以根据这两点的不同来逐位计算出name。*

*具体过程： 首先计算出位数，不同位数pad的值不一样，设定name = 'q'i，i取8-12位,payload = '~'15,计算newiv,只有取到正确的i时才可以过check_pad。 之后利用类似的方法由后至前逐位求出name,最后求出newiv，获得flag。脚本如下：*

```python
# -*- coding: utf-8 -*-
# @Date:   2019-11-09 13:16:32
# @Last Modified time: 2019-11-09 16:16:32
from pwn import *
from Crypto.Cipher import AES
def pad(s, block_size):
    return s + (block_size - len(s) % block_size) * chr(block_size - len(s) % block_size)
if __name__ == "__main__":
    robj = remote("183.129.189.62", 19206)
    temporary = robj.recvline()[:-1]
    iv = str(temporary[-64: -32])
    cipher = str(temporary[-32:])
    temporary = robj.recvline()
    # cacl name suffix
    payload = '~'*15
    for i in range(7, 13):
        name = 'q'*i
        iwt = int(iv, 16) ^ int(pad(payload, AES.block_size).encode('hex'),
                                16) ^ int(pad(name, AES.block_size).encode('hex'), 16)
        iwt = hex(iwt)[2:] + cipher
        robj.sendline(iwt)
        temporary = robj.recvline()[:-1]
        if 'padding error' not in temporary:
            break
    length = i
    the_true_name = ''
    for i in range(length):
        for j in range(33, 128):
            name = '~' * (length - 1 - i) + chr(j) + the_true_name
            payload = '~' * (len(name)-1 - i)
            iwt = int(iv, 16) ^ int(pad(payload, AES.block_size).encode('hex'), 16) ^ int(
                pad(name, AES.block_size).encode('hex'), 16)
            iwt = hex(iwt)[2:] + cipher
            robj.sendline(iwt)
            temporary = robj.recvline()[:-1]
            if 'padding error' not in temporary:
                the_true_name = chr(j) + the_true_name
                break
    user_role = "Admin"
    iwt = int(iv, 16) ^ int(pad(user_role, AES.block_size).encode('hex'),
                            16) ^ int(pad(the_true_name, AES.block_size).encode('hex'), 16)
    iwt = hex(iwt)[2:] + cipher
    robj.sendline(iwt)
    flag = robj.recvline()[:-1]
    print 'flag:' + str(flag)
```



*# Challenge # 2019 # Crypto # 湖湘杯*
*DES*
*rsa*