2021【陇原战“疫”】crypto部分writeup

原创

mortal15 于 2021-11-09 15:40:18 发布 508 收藏
分类专栏： Crypto 文章标签： python 密码学 网络安全

这次BUUCTF月赛对我来说确实还行了，不会太难，也能学到东西，主要想说说比特翻转攻击的问题，最近这种题碰的特别多，CBC的，CFB的，GCM的，各种各样的AES模式的攻击问题，从最基本的CBC开始，系统地搞个专题。

这次月赛的全面wp请看2021陇原战疫WP - n03tAck

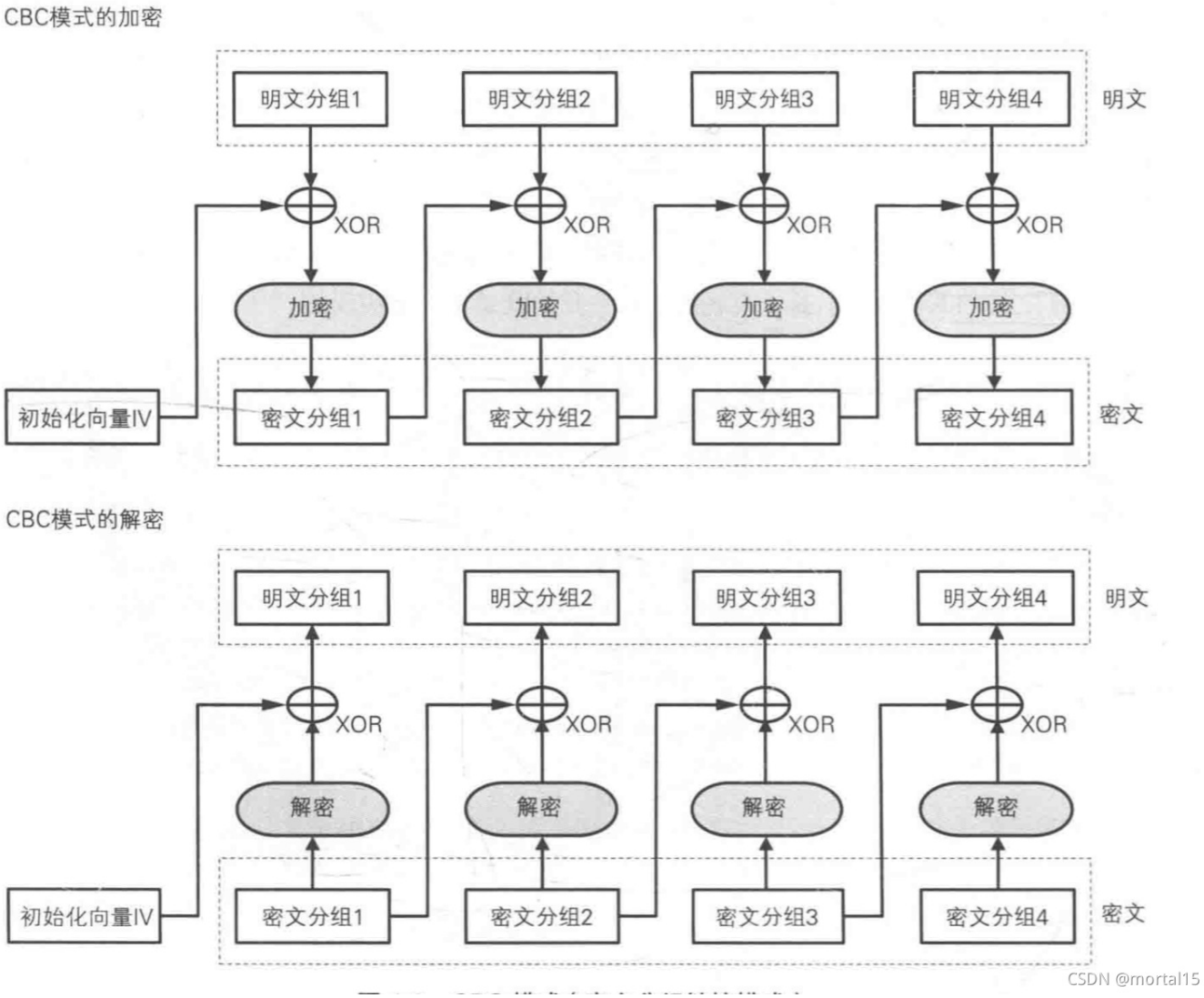
我比较菜，就把简单的题细说下。

mostlycommon

不想多说，简单的共模攻击，只是两个加密指数的最大公因数是2，最后得到的结果开个平方就是flag了。

Civet cat for Prince（比特翻转攻击）

重点说说下面这个问题，先上一张嫖来的CBC模式的执行图。



比特翻转的原理在于改变初始向量iv和密文分组，从而影响其明文，一般题目不会设太多组密文，两组或三组比较合适，一般难点就在于构造特殊的密文和iv来达成目的了。

具体到本题，下面是题目给的代码

```
from Crypto.Cipher import AES
import os
from hashlib import sha256
import socketserver
import signal
import string
import random

table = string.ascii_letters + string.digits
BANNER = br'''
 .d8888b.  d8b                   888                            888
d88P  Y88b Y8P                   888                            888
888    888                       888                            888
888        888 888  888  .d88b.  888888        .d8888b  8888b.  888888
888        888 888  888 d8P  Y8b 888          d88P"        "88b 888
888    888 888 Y88  88P 88888888 888          888      .d888888 888
Y88b  d88P 888  Y8bd8P  Y8b.     Y88b.        Y88b.    888  888 Y88b.
 "Y8888P"  888   Y88P    "Y8888   "Y888        "Y8888P "Y888888  "Y888



  d888                        8888888b           d8b
```

```
 .d888                         8888888b.          d8b
d88P"                          888   Y88b         Y8P
888                            888    888
888888  .d88b.  888d888        888   d88P 888d888 888 88888b.   .d8888b  .d88b.
888    d88""88b 888P"          8888888P"  888P"   888 888 "88b d88P"    d8P  Y8b
888    888  888 888            888        888     888 888  888 888      88888888
888    Y88..88P 888            888        888     888 888  888 Y88b.    Y8b.
888     "Y88P"  888            888        888     888 888  888  "Y8888P  "Y8888
'''

guard_menu = br'''
1.Tell the guard my name
2.Go away
'''

cat_menu = br'''1.getpermission
2.getmessage
3.say Goodbye
'''



def Pad(msg):
    return msg + os.urandom((16 - len(msg) % 16) % 16)


class Task(socketserver.BaseRequestHandler):
    def _recvall(self):
        BUFF_SIZE = 2048
        data = b''
        while True:
            part = self.request.recv(BUFF_SIZE)
            data += part
            if len(part) < BUFF_SIZE:
                break
        return data.strip()

    def send(self, msg, newline=True):
        try:
            if newline:
                msg += b'\n'
            self.request.sendall(msg)
        except:
            pass

    def recv(self, prompt=b'[-] '):
        self.send(prompt, newline=False)
        return self._recvall()

    def proof_of_work(self):
        proof = (''.join([random.choice(table) for _ in range(12)])).encode()
        sha = sha256(proof).hexdigest().encode()
        self.send(b"[+] sha256(XXXX+" + proof[4:] + b") == " + sha)
        XXXX = self.recv(prompt=b'[+] Give Me XXXX :')
        if len(XXXX) != 4 or sha256(XXXX + proof[4:]).hexdigest().encode() != sha:
            return False
        return True

    def register(self):
        self.send(b'')
        username = self.recv()
```

```
        return username

    def getpermission(self, name, iv, key):
        aes = AES.new(key, AES.MODE_CBC, iv)
        plain = Pad(name)+b"a_cat_permission"
        return aes.encrypt(plain)

    def getmessage(self, iv, key, permission):
        aes = AES.new(key, AES.MODE_CBC, iv)
        return aes.decrypt(permission)

    def handle(self):
        signal.alarm(50)
        if not self.proof_of_work():
            return
        self.send(BANNER, newline=False)
        self.key = os.urandom(16)
        self.iv = os.urandom(16)
        self.send(b"I'm the guard, responsible for protecting the prince's safety.")
        self.send(b"You shall not pass, unless you have the permission of the prince.")
        self.send(b"You have two choices now. Tell me who you are or leave now!")
        self.send(guard_menu, newline=False)
        option = self.recv()
        if option == b'1':
            try:
                self.name = self.register()
                self.send(b"Hello " + self.name)
                self.send(b"Nice to meet you. But I can't let you pass. I can give you a cat. She will play
                self.send(b'Miao~ ' + self.iv)
                for i in range(3):
                    self.send(b"I'm a magic cat. What can I help you")
                    self.send(cat_menu, newline=False)
                    op = self.recv()
                    if op == b'1':
                        self.send(b"Looks like you want permission. Here you are~")
                        permission = self.getpermission(self.name, self.iv, self.key)
                        self.send(b"Permission:" + permission)
                    elif op == b'2':
                        self.send(b"Looks like you want to know something. Give me your permission:")
                        permission = self.recv()
                        self.send(b"Miao~ ")
                        iv = self.recv()
                        plain = self.getmessage(iv, self.key, permission)
                        self.send(b"The message is " + plain)
                    elif op == b'3':
                        self.send(b"I'm leaving. Bye~")
                        break
                self.send(b"Oh, you're here again. Let me check your permission.")
                self.send(b"Give me your permission:")
                cipher = self.recv()
                self.send(b"What's the cat tell you?")
                iv = self.recv()
                plain = self.getmessage(iv, self.key, cipher)
                prs, uid = plain[16:],plain[:16]
                if prs != b'Princepermission' or uid != self.name:
                    self.send(b"You don't have the Prince Permission. Go away!")
                    return
                else:
                    self.send(b"Unbelievable! How did you get it!")
```

```
                    self.send(b"The prince asked me to tell you this:")
                    f = open('flag.txt', 'rb')
                    flag = f.read()
                    f.close()
                    self.send(flag)
            except:
                self.request.close()
        if option == b'2':
            self.send(b"Stay away from here!")
        self.request.close()


class ThreadedServer(socketserver.ThreadingMixIn, socketserver.TCPServer):
    pass


class ForkedServer(socketserver.ForkingMixIn, socketserver.TCPServer):
    pass


if __name__ == "__main__":
    HOST, PORT = '0.0.0.0', 10005
    print("HOST:POST " + HOST + ":" + str(PORT))
    server = ForkedServer((HOST, PORT), Task)
    server.allow_reuse_address = True
    server.serve_forever()
```

第一轮和门卫对话，没什么信息

和猫可以对话三次，刚开始会得到加密用的iv

第一次可以选择1，可以得到

$$enc(name \bigoplus iv) + enc(cat \bigoplus enc(name \bigoplus iv))$$

我们最终是想获得一个串，假如将其前16字节命名为c1，后16字节命名为c2，则其满足：

$$iv' \bigoplus dec(c1) == name$$

$$c1 \bigoplus dec(c2) == prince$$

很朴素地想到最终的结果必然要经过dec（解密），那么c1和c2的最外层内容必须是enc（加密），从而使两者抵消，而我们能得到的加密数据只有一组，即上面提到的

$$enc(name \bigoplus iv) + enc(cat \bigoplus enc(name \bigoplus iv))$$

为了调整方便，我们最好从后往前构造（后面的c2受c1影响），令

$$c2 = enc(cat \bigoplus enc(name \bigoplus iv))$$

$$dec(c2) = cat \bigoplus enc(name \bigoplus iv)$$

可以求出:

$$c1 = prince \bigoplus cat \bigoplus enc(name \bigoplus iv)$$

接下来iv受c1影响，求得

$$iv' = name \bigoplus dec(prince \bigoplus cat \bigoplus enc(name \bigoplus iv))$$

据此问题就转化到了得到c2,c1和iv'

c2,c1很好得到，组成的各个部分都知道，iv'则要通过和猫对话的选项2来得到（带dec的都得这么干，毕竟我们不知道key）。

这次对话用的iv可以用name，密文前16字节可以用c1，就可以得到iv' 了

之后把数据提交给门卫就可以得到flag。

```
from gmpy2  import *
from hashlib import *
from Crypto.Util.number import *
from pwn import *
import string

p=remote('node4.buuoj.cn',28621)
table = string.ascii_letters + string.digits
context.log_level='debug'

def pow_of_work(end,sha):
    for a in table:
        for b in table:
            for c in table:
                for d in table:
                    s=(a+b+c+d)+end
                    if sha256( s.encode() ).hexdigest()==sha:
                        return (a+b+c+d)
p.recvuntil(b'[+] sha256(XXXX+')
end=p.recv(8).decode()
p.recvuntil(b') == ')
sha=p.recvuntil(b'\n')[:-1].decode()
xxxx=pow_of_work(end,sha)
p.recvuntil(b'[+] Give Me XXXX :')
p.sendline(xxxx)
name=b'hellomotal151234'
premission=b'Princepermission'
p.recvuntil(b'2.Go away')
p.sendline(b'1')
p.sendline(b'hellomotal151234')
p.recvuntil(b'Miao~ ')
iv=p.recvuntil(b'\n')[:-1]
p.recvuntil(b'3.say Goodbye')
p.sendline(b'1')
p.recvuntil(b'Permission:')
enc=p.recvuntil(b'\n')[:-1]
print(enc.hex())

from Crypto.Util.strxor import *

dec_c2=strxor(enc[:16],b'a_cat_permission')
c1=strxor(b'Princepermission',dec_c2)


p.sendline(b'2')
p.recvuntil(b'Looks like you want to know something. Give me your permission:')
p.recvuntil(b'[-] ')
p.send(c1+enc[16:])
```

```
p.recvuntil(b'Miao~ ')
p.recvuntil(b'[-] ')
p.sendline(name)
p.recvuntil(b'The message is ')

plain=p.recvuntil('\n')[:-1]
print(plain)
dec_c1=strxor(iv,plain[:16])
ivv=plain[:16]

p.sendline(b'3')
p.recvuntil(b'Bye~')
p.recvuntil('[-] ')
p.sendline(c1+enc[16:])
p.recvuntil('[-] ')
p.sendline(ivv)
p.recvall()
```

借用下2021陇原战疫WP - n03tAck中的代码，但是大佬写的有点复杂了，这边按照我上面分析的思路改了下代码，更容易理解。

```
[x] Receiving all data
[x] Receiving all data: 0B
[DEBUG] Received 0x74 bytes:
    b'Unbelievable! How did you get it!\n'
    b'The prince asked me to tell you this:\n'
    b'flag{0287fc09-a1cb-4acc-b7d5-91c6a9011ed7}\n'
    b'\n'
[x] Receiving all data: 116B
[+] Receiving all data: Done (116B)
[*] Closed connection to node4.buuoj.cn port 28621
```


flag是动态的，每次开靶机都会不同。