HTB Uni CTF - Finals 2021 - Double Agents write-up

hur - SIGINT

After a long investigation we have revealed the enemy's service, which provides their agents with any needed documents. Recent events indicate that there are double agents among us. We need to read the double_agents.txt file in order to identify their names and treat them accordingly. Can you do it?

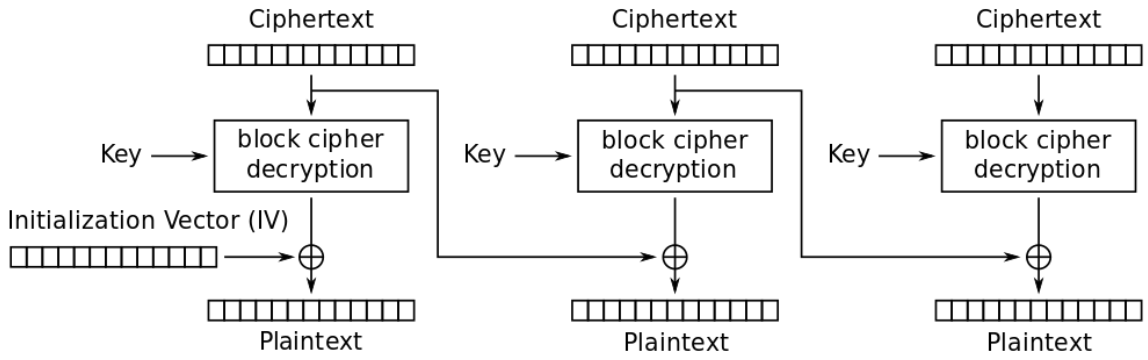
We're given code for a server that accepts hex input and attempts to decrypt it using AES CBC to open a file. The server will also give us the decrypted hex back.

We notice an interesting thing about the server code:

```
cipher = AES.new(key, AES.MODE_CBC, key)
```

They are using the secret key as the IV every time! If we can leak the IV, we can forge a message and open the `double_agents.txt` file.

CBC decryption works as follows (where the ciphertext is separated into blocks):



Cipher Block Chaining (CBC) mode decryption

Let's denote the decryption function $D(x)$ and let $\oplus$ stand for the XOR operation.

We have $D(cipher_0) \oplus IV = plaintext_0$ and $D(cipher_1) \oplus cipher_0 = plaintext_1$

Thus, if we XOR them together, we get

$D(cipher_0) \oplus D(cipher_1) \oplus cipher_0 \oplus IV = plaintext_0 \oplus plaintext_1$

If both ciphertext blocks are zeroes, we have the following:

$D(0) \oplus D(0) \oplus 0 \oplus IV = IV$, thus we have $IV = plaintext_0 \oplus plaintext_1$ and $IV = key$, thus we can leak the key by sending two blocks of zeroes and xoring the output blocks together! We can then use the key to create the ciphertext that decrypts to `double_agents.txt`.

Using this, I wrote up a Python solution.

```
from Crypto.Cipher import AES
from Crypto.Util.Padding import pad
from Crypto.Util.number import long_to_bytes
from pwn import *
```

```
payload = "00" * AES.block_size * 2

r = remote("docker.hackthebox.eu", 31083)
r.recvline()
r.sendline(payload)
decrypted = r.recvline()[len("File not found: "):-1] # extract the decrypted hex

pt_1, pt_2 = decrypted[:len(decrypted)//2], decrypted[len(decrypted)//2:] # split
in half

# XOR the plaintexts and store the result bytes
key = long_to_bytes(int(pt_1, 16) ^ int(pt_2, 16))

# create final ciphertext
a = AES.new(key, AES.MODE_CBC, key)
# The server does padding and unpadding so we'll do it too just to be safe!
final_payload = a.encrypt(pad(b"double_agents.txt", AES.block_size)).hex()

r = remote("docker.hackthebox.eu", 31083)
r.recvline()
r.sendline(final_payload)

for _ in range(21):
    response = r.recvline().decode()
    if response.startswith("HTB"):
        print(response.strip())
```

The file contains a bunch of lines with names of double agents, as well as the flag. The above solution prints out only the flag, for brevity.

Let's run it!

```
[15:11] atte@vk-3:double_agent $ python3 autosolve.py
[+] Opening connection to docker.hackthebox.eu on port 31083: Done
[+] Opening connection to docker.hackthebox.eu on port 31083: Done
HTB{1v_sh01d_b3_r4nd0m}
```

and we have the flag!