

Implement SM2 2P decrypt with real network communication

赵嵘晖 202100460100

1 实验环境

编辑器: Visual Studio Code

操作系统: Windows 11

编译语言: Python 3.10

CPU: 12th Gen Intel(R) Core(TM) i5-12500H 2.50 GHz

2 实验原理

流程图如下所示。

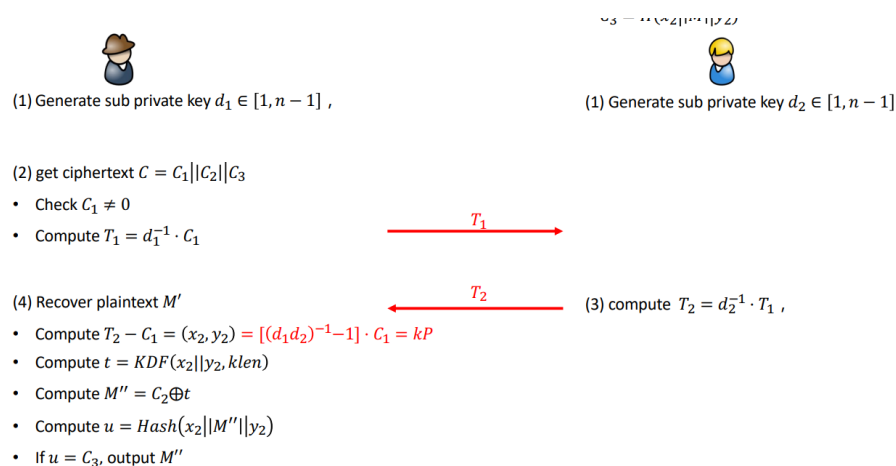


图 1: SM2_2P_DECRYPT 流程图

3 实现方法

实现签名算法, 使用 python 的 TCP 通信。通信双方为 A 和 B。其中, A 向 B 发送 T_1 , B 向 A 发送 T_2 。

在代码 SM2_2P_DECRYPT_A.py 中，实现 A 的功能。在该代码里实现加密，生成 d_1 和 d_2 以及公钥 P，使用 SM2 加密明文得到密文 C_1 、 C_2 、 C_3 。并将 d_2 发送给 B。然后开始实现解密函数。

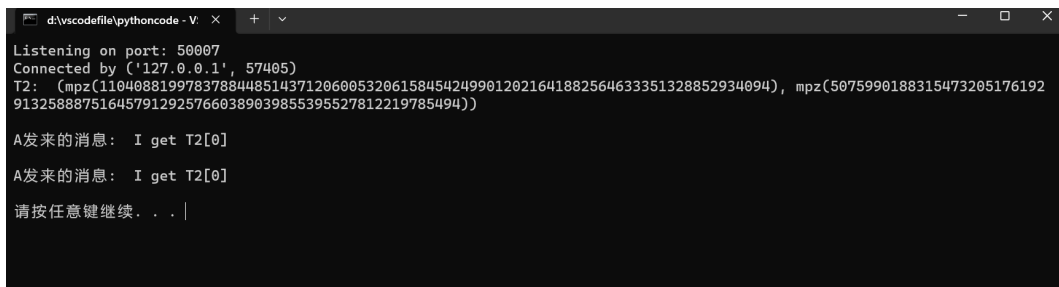
第一步，A 得到 d_1 ，然后计算 T_1 ，将 T_1 发给 B。第二步，A 收到 B 发送的 T_2 ，解密得到数据，具体方法和实验原理中的一致。

在代码 SM2_2P_DECRYPT_B.py 中，实现 B 的功能。第一步，B 收到 d_2 。第二步，收到 A 发送的 T_1 后回复确认，然后计算 T_2 ，并将其发送给 A。

4 实验结果

实验结果如下。

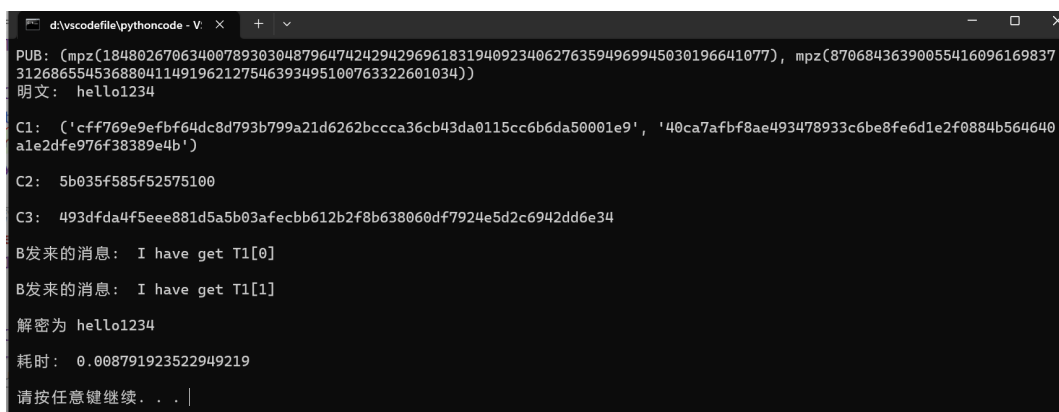
B 的结果展示。



```
d:\vscodefile\pythoncode - V
Listening on port: 50007
Connected by ('127.0.0.1', 57405)
T2: (mpz(110408819978378844851437120600532061584542499012021641882564633351328852934094), mpz(50759901883154732051761929132588875164579129257660389039855395527812219785494))
A发来的消息: I get T2[0]
A发来的消息: I get T2[0]
请按任意键继续. . .
```

图 2: B 通信实验结果

A 的结果展示。



```
d:\vscodefile\pythoncode - V
PUB: (mpz(18480267063400789303048796474242942969618319409234062763594969945030196641077), mpz(87068436390055416096169837312686554536880411491962127546393495100763322601034))
明文: hello1234
C1: ('cfff769e9efb64dc8d793b799a21d6262bccca36cb43da0115cc6b6da50001e9', '40ca7afb8ae493478933c6be8fe6d1e2f0884b564640a1e2dfe976f38389e4b')
C2: 5b035f585f52575100
C3: 493dfda4f5eee881d5a5b03afecbb612b2f8b638060df7924e5d2c6942dd6e34
B发来的消息: I have get T1[0]
B发来的消息: I have get T1[1]
解密为 hello1234
耗时: 0.008791923522949219
请按任意键继续. . .
```

图 3: A 通信实验结果

如上图所示，一次签名耗时为：0.008791923522949219s。

5 代码

如下是核心的代码。

5.1 SM2_2P_DECRYPT_B

```
1  import gmpy2
2  import random
3  import socket
4  #from os.path import commonprefix
5  from threading import Thread
6
7  HOST = ''
8  PORT = 50007
9
10 def SIGN_2P_B(d2, conn, addr):
11     while True:
12         # 接收T1
13         t1_0 = conn.recv(1024).decode()
14         conn.sendall("I_have_get_T1[0]".encode())
15         t1_1 = conn.recv(1024).decode()
16         conn.sendall("I_have_get_T1[1]".encode())
17         t1_0 = int(t1_0)
18         t1_1 = int(t1_1)
19         T1 = (t1_0, t1_1)
20
21
22
23         #计算T2
24         T2 = Mul_Add(T1[0], T1[1], gmpy2.invert(d2, n))
25
26         # 给A发送T2
27         conn.sendall(str(T2[0]).encode())
28         data_1 = conn.recv(1024).decode()
29         print("A发来的消息:", data_1, '\n')
30         conn.sendall(str(T2[1]).encode())
31         data_2 = conn.recv(1024).decode()
32         print("A发来的消息:", data_2, '\n')
33
34         break
35     conn.close()
36
37
38 sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```

```

39 sock.bind((HOST, PORT))
40 sock.listen(10)
41 print('Listening_on_port:', PORT)
42 conn, addr = sock.accept()
43 print('Connected_by', addr)
44 d2 = conn.recv(1024).decode()
45 d2 = int(d2)
46 mthread = Thread(target = SIGN_2P_B, args = (d2, conn, addr))
47 mthread.start()
48 sock.close()
49

```

5.2 SM2_2P_DECRYPT_A

```

1  HOST = '127.0.0.1'
2  PORT = 50007
3  SOCKET = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
4  try:
5      SOCKET.connect((HOST, PORT))
6  except Exception as e:
7      print('Server_not_found_or_not_open')
8  sys.exit()
9  # 解密算法
10 def Decrypt_A(d1, C1, C2, C3):
11     T1 = Mul_Add(int(C1[0], 16), int(C1[1], 16), gmpy2.invert(d1, n))
12
13     # 把T1发给B
14     SOCKET.sendall(str(T1[0]).encode())
15     data1 = SOCKET.recv(1024).decode()
16     print("B发来的消息:", data1, '\n')
17     SOCKET.sendall(str(T1[1]).encode())
18     data2 = SOCKET.recv(1024).decode()
19     print("B发来的消息:", data2, '\n')
20
21     # 接收B发来的T2
22     t2_0 = SOCKET.recv(1024).decode()
23     SOCKET.sendall("I_get_T2[0]".encode())
24     t2_1 = SOCKET.recv(1024).decode()
25     SOCKET.sendall("I_get_T2[0]".encode())
26
27     t2_0 = int(t2_0)
28     t2_1 = int(t2_1)
29     T2 = (t2_0, t2_1)
30

```

```

31
32     Temp = Add(T2[0], T2[1], int(C1[0], 16), p - int(C1[1], 16))
33     x2, y2 = '{:0256b}'.format(Temp[0]), '{:0256b}'.format(Temp[1])
34     klen = len(C2) * 4
35     t = KDF(x2 + y2, klen)
36     M = int(C2, 16) ^ int(t, 2)
37     M = Int_bin(M)
38     Tlen = len(M)
39     m = '0' * (klen - Tlen) + M
40     u = sm3(hex(int(x2 + m + y2, 2))[2:])
41     if u != C3:
42         return False
43     M = hex(int(m, 2))[2:]
44     M = binascii.a2b_hex(M).decode()
45     return M
46
47 # 示例
48 d1 = random.randint(1,n-1)
49 d2 = random.randint(1,n-1)
50 pub = Mul_Add(Gx, Gy, gmpy2.invert(d1 * d2, n) - 1)
51 print("PUB:", pub)
52 SOCKET.sendall(str(d2).encode())
53 data = "hello1234"
54 c1, c2, c3 = encrypt(data, pub)
55 print("明文:", data, '\n')
56 print("C1:", c1, '\n')
57 print("C2:", c2, '\n')
58 print("C3:", c3, '\n')
59
60 start = time.time()
61 M1 = Decrypt_A(d1, c1, c2, c3)
62 end = time.time()
63 print("解密为", M1, '\n')
64 print("耗时: ", end - start, '\n')
65

```