Implement the above ECMH scheme

赵嵘晖 202100460100

1 实验环境

编辑器：Visual Studio Code
操作系统:Windows11
编译语言:Python 3.10
CPU: 12th Gen Intel(R) Core(TM) i5-12500H 2.50 GHz

2 实现方法

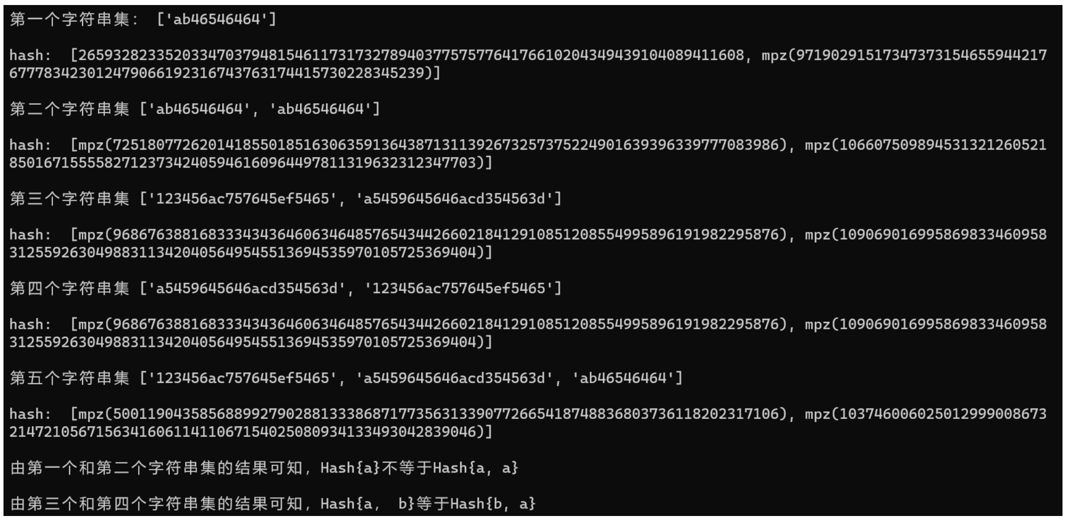
实现 ECMH 方案，如得到 Hash(a)，需要先将 a 经 sm3 哈希，然后作为 x 代入到 $y^2 = x^3 + ax + b$ 的方程中求 y。这里需要使用二次剩余求出对应的 y。将初始点对应为 (0, 0), 初始点与该点相加。即得到 Hash(a)。（椭圆曲线的点的加法）

若得到 Hash(a,b)。只需在 Hash(a) + Hash(b)。先将 b 经 sm3 哈希，然后作为 x 代入到 $y^2 = x^3 + ax + b$ 的方程中求 y。然后与 Hash(a) 相加。

计算 Hash(a,b,c) - Hash(c) 的过程与相加类似，只不过椭圆曲线上的加法改成减法。

3 实验结果

实验结果如下。



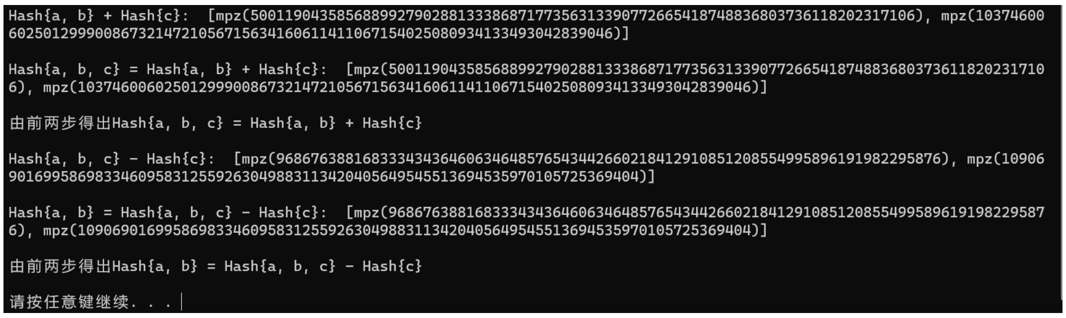
```
第一个字符串集： ['ab46546464']

hash:  [26593282335203347037948154611731732789403775757764176610204349439104089411608, mpz(971902915173473731546559442176777834230124790661923167437631744157302283452 39)]

第二个字符串集 ['ab46546464', 'ab46546464']

hash:  [mpz(72518077262014185501851630635913643871311139267325737522490163939633977708 3986), mpz(1066075098945313212605218501671555582712373424059461609644978113196323123 47703)]

第三个字符串集 ['123456ac757645ef5465', 'a5459645646acd354563d']

hash:  [mpz(96867638816833343436460634648576543442660218412910851208554995896191982295876), mpz(10906901699586983346095831255926304988311342040564954551369453597010572536 9404)]

第四个字符串集 ['a5459645646acd354563d', '123456ac757645ef5465']

hash:  [mpz(96867638816833343436460634648576543442660218412910851208554995896191982295876), mpz(10906901699586983346095831255926304988311342040564954551369453597010572536 9404)]

第五个字符串集 ['123456ac757645ef5465', 'a5459645646acd354563d', 'ab46546464']

hash:  [mpz(50011904358568899279028813338687177356313390772665418748836803736118202317106), mpz(10374600602501299900867321472105671563416061141106715402508093413349304283 9046)]

由第一个和第二个字符串集的结果可知，Hash{a}不等于Hash{a, a}

由第三个和第四个字符串集的结果可知，Hash{a,  b}等于Hash{b, a}
```

图 1: ECMH 实验结果

可以看出：

Hash(a) $\neq$ Hash(a,a)

Hash(a, b) = Hash(b, a)



```
Hash{a, b} + Hash{c}:  [mpz(50011904358568899279028813338687177356313390772665418748836803736118202317106), mpz(10374600602501299900867321472105671563416061141106715402508093413349304283 9046)]

Hash{a, b, c} = Hash{a, b} + Hash{c}:  [mpz(50011904358568899279028813338687177356313390772665418748836803736118202317106), mpz(10374600602501299900867321472105671563416061141106715402508093413349304283 9046)]

由前两步得出Hash{a, b, c} = Hash{a, b} + Hash{c}

Hash{a, b, c} - Hash{c}:  [mpz(96867638816833343436460634648576543442660218412910851208554995896191982295876), mpz(10906901699586983346095831255926304988311342040564954551369453597010572536 9404)]

Hash{a, b} = Hash{a, b, c} - Hash{c}:  [mpz(96867638816833343436460634648576543442660218412910851208554995896191982295876), mpz(10906901699586983346095831255926304988311342040564954551369453597010572536 9404)]

由前两步得出Hash{a, b} = Hash{a, b, c} - Hash{c}

请按任意键继续. . .
```

图 2: ECMH 实验结果

可以看出：

Hash(a,b) + Hash(c) = Hash(a, b, c)

Hash(a, b, c) - Hash(c) = Hash(a,b)

4 代码

如下是核心的代码。

4.1 ECMH

```python
# ECMH
def ECMH(data):
    Infinty  = Add(0, 0, 0, 0)
    for item in data:
        item = sm3(item)  #都是字符串类型
        item = int(item, 16)
        item1 = (pow(item, 3) + a * item + b) % p
        item_y = QR(item1, int(p))
        Infinty  = Add(Infinty[0], Infinty [1], item, item_y)
    return Infinty

def ECMH_ADD(data1, data2):
    data2 = sm3(data2[0])
    data2 = int(data2, 16)
    data2_x = (pow(data2, 3) + a * data2 + b) % p
    data2_y = QR(data2_x, p)
    result  = Add(data1[0], data1[1], data2, data2_y)
    return result

def ECMH_REMOVE(data1, data2):
    data2 = sm3(data2[0])
    data2 = int(data2, 16)
    data2_x = (pow(data2, 3) + a * data2 + b) % p
    data2_y = QR(data2_x, p)
    result  = Add(data1[0], data1[1], data2, p − data2_y)
    return result

#示例
str1  = [’ab46546464’]
str2  = [’ab46546464’, ’ab46546464’]
str3  = [’123456ac757645ef5465’, ’a5459645646acd354563d’]
str4  = [’a5459645646acd354563d’, ’123456ac757645ef5465’]
str5  = [’123456ac757645ef5465’, ’a5459645646acd354563d’, ’ab46546464’]

strx  = [’ab46546464’, ’3265752a23434c’]
stry  = [’3265752a23434c’, ’ab46546464’]
result1  = ECMH(str1)
result2  = ECMH(str2)
result3  = ECMH(str3)
result4  = ECMH(str4)
result5  = ECMH(str5)
print(”第一个字符串集：”, str1, ’\n’)
print(”hash:␣”, result1 , ’\n’)

print(”第二个字符串集”, str2, ’\n’)
print(”hash:␣”, result2 , ’\n’)

```

```
    print("第三个字符串集", str3, '\n')
    print("hash:␣", result3 , '\n')

    print("第四个字符串集", str4, '\n')
    print("hash:␣", result4 , '\n')

    print("第五个字符串集", str5, '\n')
    print("hash:␣", result5 , '\n')

    if result1 != result2:
        print("由第一个和第二个字符串集的结果可知，Hash{a}不等于Hash{a,␣a}\n")

    if result3 == result4:
        print("由第三个和第四个字符串集的结果可知，Hash{a，␣b}等于Hash{b,␣a}\n") #这个有问题


    result6 = ECMH_ADD(result3, str1)
    if result6 == result5:
        print("Hash{a,␣b}␣+␣Hash{c}:␣", result6, '\n')
        print("Hash{a,␣b,␣c}␣=␣Hash{a,␣b}␣+␣Hash{c}:␣", result5, '\n')
        print("由前两步得出Hash{a,␣b,␣c}␣=␣Hash{a,␣b}␣+␣Hash{c}\n")

    result7 = ECMH_REMOVE(result5, str1)
    if result7 == result3:
        print("Hash{a,␣b,␣c}␣−␣Hash{c}:␣", result7, '\n')
        print("Hash{a,␣b}␣=␣Hash{a,␣b,␣c}␣−␣Hash{c}:␣", result3, '\n')
        print("由前两步得出Hash{a,␣b}␣=␣Hash{a,␣b,␣c}␣−␣Hash{c}\n")


```