FHE stealth address

June 27, 2023

1 Bob wallet generation

[4]:
```
# config env
#!pip install py_ecc
#!pip uninstall sha3
#!pip install ecdsa
#!pip install pysha3
nbconvert[webpdf]
!pip install phe
```

```
Requirement already satisfied: phe in ./opt/anaconda3/lib/python3.9/site-
packages (1.5.0)
```

[5]:
```
import hashlib
from py_ecc.secp256k1 import *
import sha3
from eth_account import Account
from ecdsa import SigningKey, SECP256k1
from phe import paillier
```

[6]:
```
print(secp256k1.G)
```

```
(55066263022277343669578718895168534326250603453777594175500187360389116729240,
32670510020758816978083085130507043184471273380659243275938904335757337482424)
```

[7]:
```
def sk_PK_HPK_WA(sk):
    # PK = secp256k1.privtopub(s.to_bytes(32, "big"))
    PK = secp256k1.multiply(secp256k1.G, secp256k1.bytes_to_int(sk.to_bytes(32,␣
 ↪"big")))
    HPK = sha3.keccak_256(PK[0].to_bytes(32, "big")+PK[1].to_bytes(32, "big")).
 ↪hexdigest()
    WA = "0x"+ HPK[-40:]
    return PK, HPK, WA

sk_bob = int(0x13803677559c049a98a1576ad2f1fd15bdd9f81886f225c0850e883bd7d4cf63)
PK_bob, HPK_bob, WA_bob = sk_PK_HPK_WA(sk_bob)
```

```
name = 'Bob'
print(name)
#print('==== G: \n', secp256k1.G)
print('==== sk_bob: \n', sk_bob)
print('==== PK_bob=G*sk_bob \n', PK_bob)
#print('==== HPK=h256(PK) \n', HPK_bob)
print('==== WA2=-40(HPK): \n', WA_bob)


def get_PK(sk):
    PK = secp256k1.multiply(secp256k1.G, secp256k1.bytes_to_int(sk.to_bytes(32,␣
 ↪"big")))
    return PK
```

```
Bob
==== sk_bob:
 88204764589258015223816929548307529848956589629393118884214852204568178072 03
==== PK_bob=G*sk_bob
 (66308977532324777246755440929893538803702139848914800508388379439121578598121,
74282901316977640338205813941139843601962206750695816950367302342758470583341)
==== WA2=-40(HPK):
 0x8a31e51ad0cf970cfb4de3c7c9929a0813fbfea1
```

2 Bob secret key generation (Paillier)

[8]:
```
public_key, private_key = paillier.generate_paillier_keypair()
print('==== sk_bob_fhe: \n', private_key)
print('==== PK_bob_fhe: \n', public_key)
```

```
==== sk_bob_fhe:
 <PaillierPrivateKey for <PaillierPublicKey 1755f1254f>>
==== PK_bob_fhe:
 <PaillierPublicKey 1755f1254f>
```

[9]:
```
C2 = public_key.encrypt(sk_bob)
print('==== C2: \n', C2)
```

```
==== C2:
 <phe.paillier.EncryptedNumber object at 0x7ff22a28ff70>
```

3 Alice secret key generation (Paillier)

[10]:
```
sk_alice_fhe =␣
 ↪int(0xd952fe0740d9d14011fc8ead3ab7de3c739d3aa93ce9254c10b0134d80d26a30)
PK_alice_fhe = get_PK(sk_alice_fhe)
```

```
name = 'Alice'
print(name)
print('==== sk_alice_fhe: \n', sk_alice_fhe)
print('==== PK_alice_fhe=G*sk_alice_fhe \n', PK_alice_fhe)

def get_PK(sk):
    PK = secp256k1.multiply(secp256k1.G, secp256k1.bytes_to_int(sk.to_bytes(32,␣
 ↪"big")))
    return PK
```

```
Alice
==== sk_alice_fhe:
 98298522841001936806542785690725155579423954746529030545381741127082542524976
==== PK_alice_fhe=G*sk_alice_fhe
 (22246744184454969143801186698733154500632648736073949898323976612504587645286,
110772761940586493986212935445517909380300793379795289150161960681985511655321)
```

[15]:
```
PK_fhe_alice = secp256k1.add(PK_alice_fhe, PK_bob)
print('==== PK_fhe_alice: \n', PK_fhe_alice)
```

```
==== PK_fhe_alice:
 (56643590336203163844655999682952879186150511458997791576780669002181049948650,
80447776592323555958913524344157008921432373288044289322709521989151125370528)
```

[16]:
```
HPK_fhe_alice = sha3.keccak_256(PK_fhe_alice[0].
 ↪to_bytes(32,'big')+PK_fhe_alice[1].to_bytes(32,'big')).hexdigest()
WA_fhe_alice = "0x"+ HPK_fhe_alice[-40:]
print('==== WA_z: \n', WA_fhe_alice)
```

```
==== WA_z:
 0x59c787fb7b25627bd93272b4695c673340b0959d
```

[17]:
```
C1 = public_key.encrypt(sk_alice_fhe)
print('==== C1: \n', C1)
```

```
==== C1:
 <phe.paillier.EncryptedNumber object at 0x7ff229716e50>
```

4 Bob side

[18]:
```
C = C1 + C2
print('==== C: \n', C)
```

```
==== C:
 <phe.paillier.EncryptedNumber object at 0x7ff229716370>
```

[19]:
```
sk_fhe = private_key.decrypt(C)   # only bob has
print('==== sk_fhe: \n', sk_fhe)
print('==== sk_alice_fhe + sk_bob: \n', sk_alice_fhe + sk_bob)
```

```
==== sk_fhe:
 107118999299927738328924478645555908564319613709468342433803226347539360332179
==== sk_alice_fhe + sk_bob:
 107118999299927738328924478645555908564319613709468342433803226347539360332179
```

[20]:
```
PK_fhe_bob = get_PK(sk_fhe)
print('==== PK_fhe: \n', PK_fhe_bob)
```

```
==== PK_fhe:
 (566435903362031638446559996829528791861505114589977915767806690021810499486 50,
804477765923235559589135243441570089214323732880442893227095219891511253705 28)
```

[21]:
```
PK_fhe_alice == PK_fhe_bob
```

[21]:
```
True
```

[23]:
```
HPK_fhe_bob = sha3.keccak_256(PK_fhe_bob[0].to_bytes(32,'big')+PK_fhe_bob[1].
 ↪to_bytes(32,'big')).hexdigest()
WA_fhe_bob = "0x"+ HPK_fhe_bob[-40:]
print('==== WA_z: \n', WA_fhe_bob)
```

```
==== WA_z:
 0x59c787fb7b25627bd93272b4695c673340b0959d
```

[24]:
```
WA_fhe_alice == WA_fhe_bob  #Alice and bob has the same wallet address but only␣
 ↪bob can decrypt it
```

[24]:
```
True
```