Blockchain Technology and Data Economics

Practise 1 - Programming bitcoin with Python

Version: 2020 Spring

Introduction

In this practise, we are going to introduce the following topics:

- The Raspberry Pi Bitcoin tools library and how to start using it
- How to generate private keys and public keys
- How to create a simple bitcoin address from the private keys and public keys you generated

To get started with bitcoin using Python, you must install Python 3.x and the bitcoin Python library called Pi Bitcoin tools in the system.

I. Pi Bitcoin tools library

To install the Pi Bitcoin tools library, open the command-line program and execute the following command:

```
pip install bitcoin
```

The best thing about this library is that it does not need to have a bitcoin node on your computer in order for you to start using it.

This library connects to the bitcoin network and pulls data from places such as Blockchain.info.
We shall start by writing the equivalent of a *Hello World* program for bitcoin in Python. In the *hello_bitcoin.py* script, the demonstration of a new bitcoin address is created using Python.
Go through the following steps to run the program:

1. Import the bitcoin library:

```
#!/usr/bin/env python
'''
Title - Hello Bitcoin
This program demonstrates the creation of
- private key,
- public key
- and a bitcoin address.
'''


# import bitcoin
from bitcoin import *
```

2. Generate a private key using the random key function:

```
        my_private_key = random_key()
```

3. Display the private key on the screen:

```
        print("Private Key: %s\n" % my_private_key)
```

II. How to generate private keys and public keys

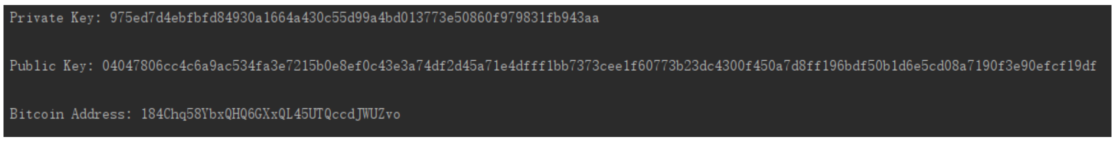
With the private key, a public key is generated. Perform this step by passing the private key that was generated to the *privtopub* function, as shown here:

```
# Generate Public Key
my_public_key = privtopub(my_private_key)
print("Public Key: %s\n" % my_public_key)
```

Now, with the public key, generate a bitcoin address. Do this by passing the public key that is generated to the *pubtoaddr* function:

```
# Create a bitcoin address
my_bitcoin_address = pubtoaddr(my_public_key)
print("Bitcoin Address: %s\n" % my_bitcoin_address)
```

The following screenshot shows the private key, public key and bitcoin address that is generated:



```
Private Key: 975ed7d4ebfbfd84930a1664a430c55d99a4bd013773e50860f979831fb943aa

Public Key: 04047806cc4c6a9ac534fa3e7215b0e8ef0c43e3a74df2d45a71e4dfff1bb7373cee1f60773b23dc4300f450a7d8ff196bdf50b1d6e5cd08a7190f3e90efcf19df

Bitcoin Address: 184Chq58YbxQHQ6GXxQL45UTQccdJWUZvo
```

Bitcoin address

A bitcoin address is a single-use token. Just as people use email addresses to send and receive emails, you can use this bitcoin address to send and receive bitcoins. Unlike email addresses, however, people have many different bitcoin addresses, and a unique address should be used for each transaction.

III. Creating a multi-signature bitcoin address

A multi-signature address is an address that is associated with more than one private key; therefore, we need to create three private keys.

Go through the following steps to create a multi-signature bitcoin address:

1. Create three private keys:

```
#!/usr/bin/env python
'''
Title - Create multi-signature address

This program demonstrates the creation of
Multi-signature bitcoin address.
'''
# import bitcoin
from bitcoin import *


# Create Private Keys
my_private_key1 = random_key()
my_private_key2 = random_key()
my_private_key3 = random_key()

print("Private Key1: %s" % my_private_key1)
print("Private Key2: %s" % my_private_key2)
print("Private Key3: %s" % my_private_key3)
print('\n')
```

2. Create three public keys from those private keys using the *privtopub* function:

```
# Create Public keys
my_public_key1 = privtopub(my_private_key1)
my_public_key2 = privtopub(my_private_key2)
my_public_key3 = privtopub(my_private_key3)

print("Public Key1: %s" % my_public_key1)
print("Public Key2: %s" % my_public_key2)
print("Public Key3: %s" % my_public_key3)
print('\n')
```

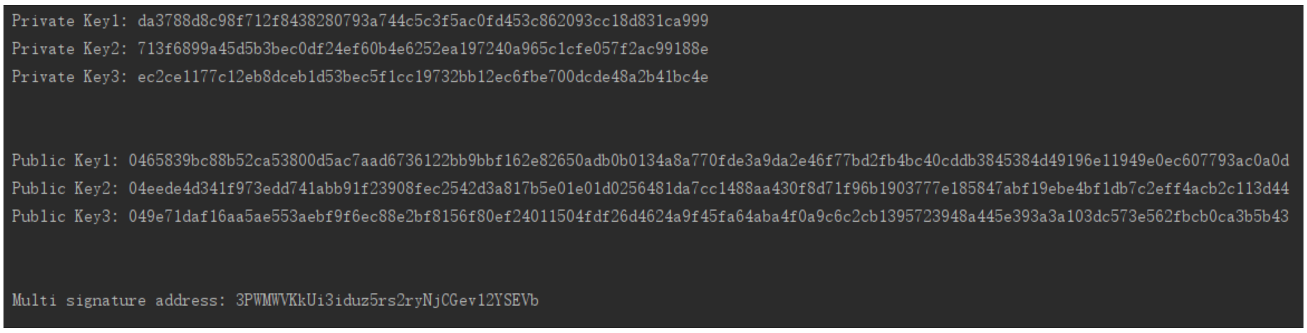
3. After generating the public keys, create the *multisig* by passing the three private

keys to the *mk_multi-sig_script* function. The resulting *multisig* is passed to the *addr* script function to create the *multisignature* bitcoin address.

```
# Create Multi-signature address
my_multi_sig = mk_multisig_script(my_private_key1,
my_private_key2, my_private_key3, 2,3)
my_multi_address = scriptaddr(my_multi_sig)
print("Multi signature address: %s" % my_multi_address)
```

4. Print the *multisignature* address and execute the script.

The following screenshot shows the output for the *multisig* bitcoin address:

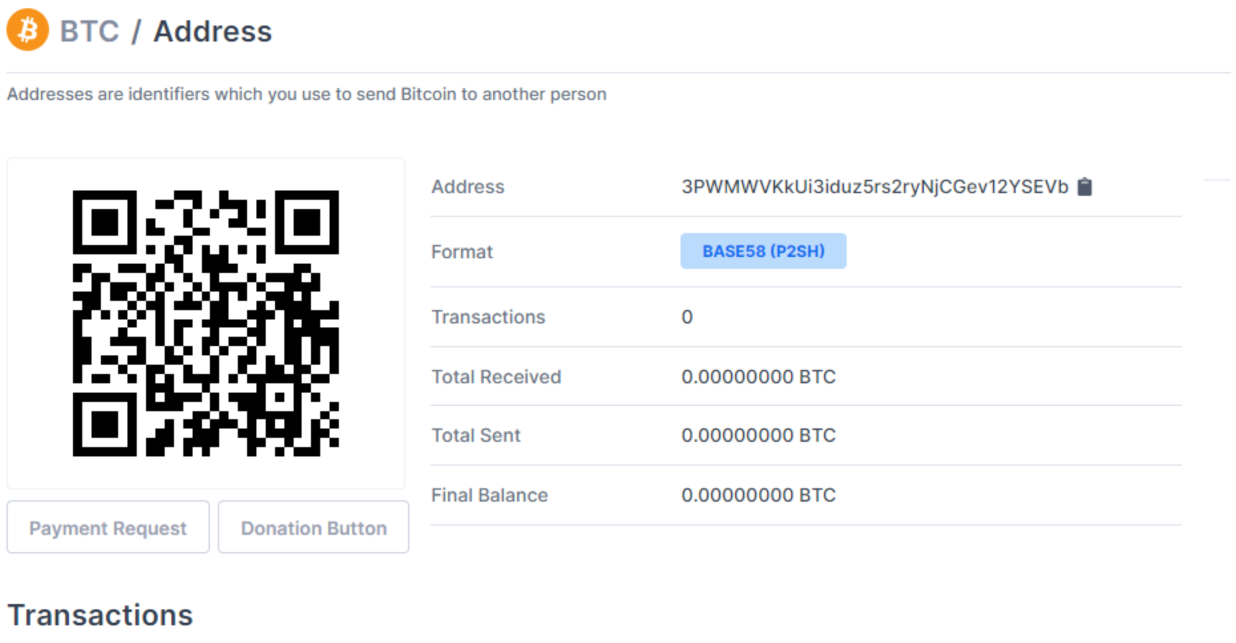


```
Private Key1: da3788d8c98f712f8438280793a744c5c3f5ac0fd453c862093cc18d831ca999
Private Key2: 713f6899a45d5b3bec0df24ef60b4e6252ea197240a965c1cfe057f2ac99188e
Private Key3: ec2ce1177c12eb8dceb1d53bec5f1cc19732bb12ec6fbe700dcde48a2b41bc4e


Public Key1: 0465839bc88b52ca53800d5ac7aad6736122bb9bbf162e82650adb0b0134a8a770fde3a9da2e46f77bd2fb4bc40cddb3845384d49196e11949e0ec607793ac0a0d
Public Key2: 04eede4d341f973edd741abb91f23908fec2542d3a817b5e01e01d0256481da7cc1488aa430f8d71f96b1903777e185847abf19ebe4bf1db7c2eff4acb2c113d44
Public Key3: 049e71daf16aa5ae553aebf9f6ec88e2bf8156f80ef24011504fdf26d4624a9f45fa64aba4f0a9c6c2cb1395723948a445e393a3a103dc573e562fbcb0ca3b5b43


Multi signature address: 3PWMWVKkUi3iduz5rs2ryNjCGev12YSEVb
```

You can also look at the preexisting bitcoin addresses' transactional history.
We will first get a valid address from Blockchain.info.
The following screenshot shows the copied address of a bitcoin block:



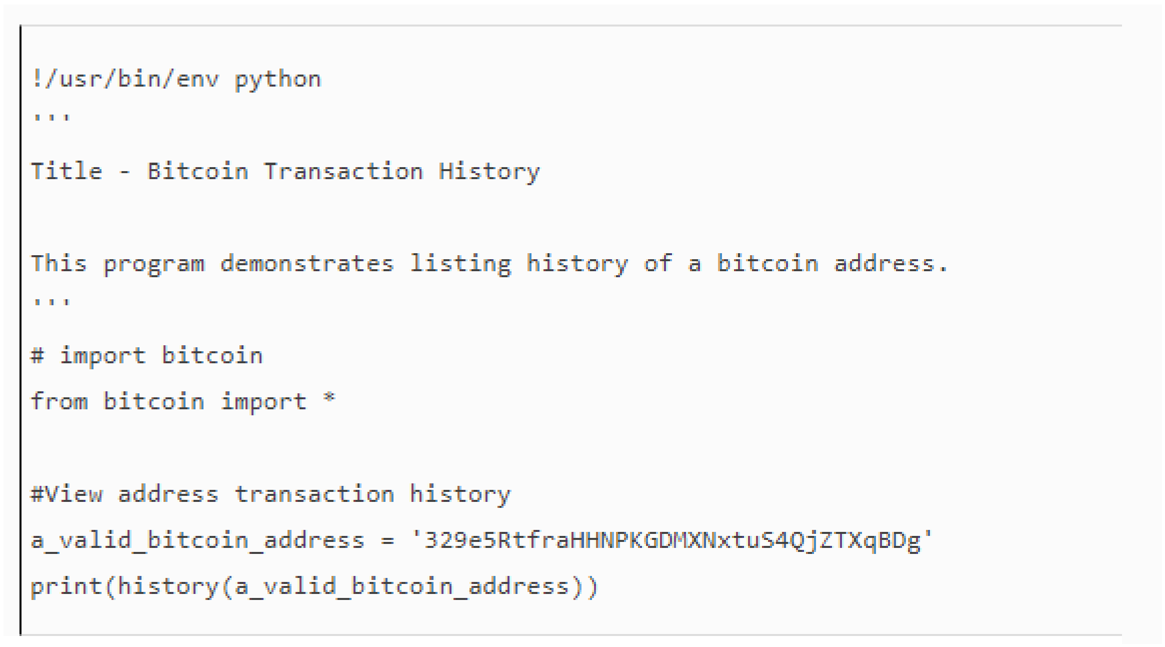
BTC / Address

Addresses are identifiers which you use to send Bitcoin to another person

Address	3PWMWVKkUi3iduz5rs2ryNjCGev12YSEVb
Format	BASE58 (P2SH)
Transactions	0
Total Received	0.00000000 BTC
Total Sent	0.00000000 BTC
Final Balance	0.00000000 BTC

Transactions

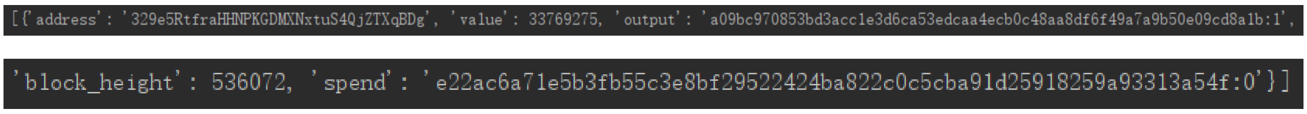
Pass the copied address to the *history* function, as shown in the following code, along with the output to get the history of the bitcoin address, including the transactional information:



```
!/usr/bin/env python
'''
Title - Bitcoin Transaction History

This program demonstrates listing history of a bitcoin address.
'''
# import bitcoin
from bitcoin import *

#View address transaction history
a_valid_bitcoin_address = '329e5RtfraHHNPKGDMXNxtuS4QjZTXqBDg'
print(history(a_valid_bitcoin_address))
```



```
[{'address': '329e5RtfraHHNPKGDMXNxtuS4QjZTXqBDg', 'value': 33769275, 'output': 'a09bc970853bd3acc1e3d6ca53edcaa4ecb0c48aa8df6f49a7a9b50e09cd8a1b:1',
'block_height': 536072, 'spend': 'e22ac6a71e5b3fb55c3e8bf29522424ba822c0c5cba91d25918259a93313a54f:0'}]
```

Well done!! Show your results to TA!

The End of Practice 1