EE 465: Cryptocurrency and Blockchain Technologies (Autumn 2019)
Instructor: Saravanan Vijayakumaran
Indian Institute of Technology Bombay

Assignment 1: 10 points | Date: August 20, 2019

1. [2 points] Let the SHA-256 hash of your roll number be y. Find another input whose SHA-256 hash coincides with y in the initial 24 bits. For ease of verification, **submit a Python script** of the following form.

```
import hashlib

h = lambda x: hashlib.sha256(x).hexdigest()

print h('16000001')       # Put your roll number here
print h('Your solution')
```

2. [2 points] The Internet Archive accepts Bitcoin donations (`https://archive.org/donate/cryptocurrency/`). The donation address is a P2PKH address which is given by `1Archive1n2C579dMsAu3iC6tWzuQJz8dN`. The leading `1` indicates that it is a P2PKH address (the address byte `0x00` is converted into the `1`). Such addresses are called *vanity addresses* due to their similarity to vanity registration number plates on cars.

   Create a P2PKH vanity address which begins with the **first four letters of your first name** (excluding letters which are not possible in the Base58 encoding). Provide the private key corresponding to this vanity address in Wallet Import Format (`https://en.bitcoin.it/wiki/Wallet_import_format`). **Submit a Python script which shows your method.**

   Hint: Use the `bit` *Python library (requires Python3). It can be installed on Linux systems using* `sudo pip3 install bit`*. You may need to install* `pip3` *via* `sudo apt install python3-pip`*. You can use the functions in* `https://github.com/ofek/bit/blob/master/bit/keygen.py`*.*

3. [2 points] The Brain Wallet feature at `www.bitaddress.org` uses the SHA-256 hash of a passphrase to calculate the private key. The SHA-256 output can be any 256-bit string but the private key is an integer in the range $\{1, 2, \ldots, n-1\}$ where n is the order of the secp256k1 elliptic curve group.

   To see why this is not a problem in practice, calculate the probability that the SHA-256 hash of a passphrase is larger than $n-1$ assuming that SHA-256 outputs are uniformly distributed on $\{0,1\}^{256}$. Express your answer in the form $x.yz \times 10^{-m}$. **Submit a Python script which shows your computations.**

   Hint: The `bit` *Python library has the group order in* `https://github.com/ofek/bit/blob/master/bit/curve.py`*. You can use the* `mpmath` *Python library to do high precision arithmetic. See* `http://mpmath.org/doc/1.1.0/basics.html` *for usage.*

4. [2 points] Show that the base point $P = (x, y)$ given by the following coordinates lies on the secp256k1 curve. **Submit a Python script which shows your computations.**

$$x = \texttt{0x79BE667EF9DCBBAC55A06295CE870B07029BFCDB2DCE28D959F2815B16F81798},$$
$$y = \texttt{0x483ADA7726A3C4655DA4FBFC0E1108A8FD17B448A68554199C47D08FFB10D4B8}.$$

   Hint: Use the functions in `https://github.com/ofek/bit/blob/master/bit/curve.py`*.*

5. [2 points] Complete the following steps.

   - Generate a testnet address using the generator at `https://bitcoinpaperwallet.com/bitcoinpaperwallet/generate-wallet.html?design=alt-testnet`.
   - Receive some test bitcoins into this address using the faucets at either `https://bitcoinfaucet.uo1.net/` or `https://coinfaucet.eu/en/btc-testnet/`.
   - Send all the bitcoins in the address (minus transaction fees) to the following address: `mtFSBwB8HDYerC2Y1H1v8GhESp5PF122De`.