II.2317 Cybersecurity (ISEP)

Nour El Madhoun

LAB Blockchain (Proof-of-Work)

nour.el-madhoun@isep.fr

You can work in a group of 2 or 3 students

Objectives of the LAB

- Understand what is mining and what is Proof-of-Work (PoW).
- Evaluate the performance according to different values of the blockchain difficulty.

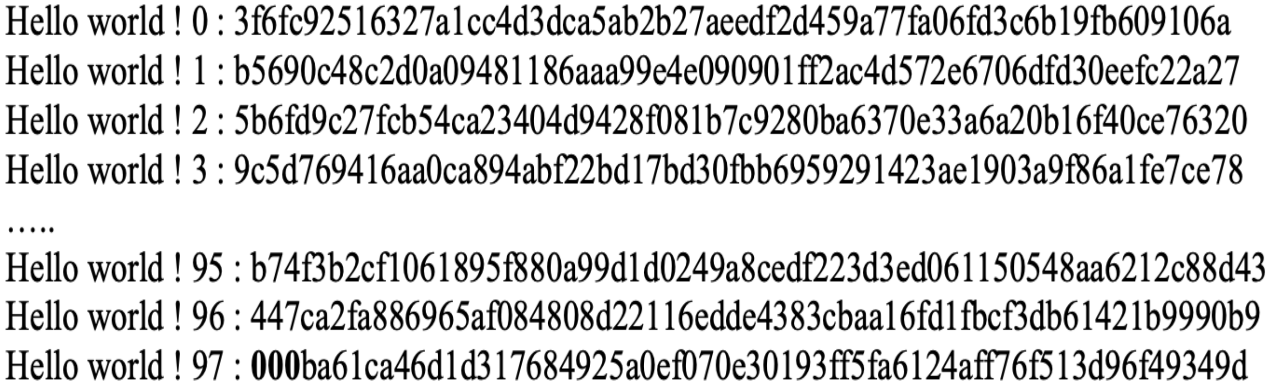
Introduction

Proof-of-Work (PoW) is a peer-to-peer consensus protocol at the heart of the Bitcoin cryptocurrency. The principle of Proof-of-Work is to solve a challenge whose solution is **hard to find and easy to verify**. Bitcoin uses Hashcash, a Proof-of-Work system designed by Adam Back, which challenge can only be solved by performing a task that is highly computationally intensive. That is, for a **given alphanumeric string**, adding **a random alphanumeric string** to it until the **Hash** (the digital fingerprint) of the whole is below a given threshold (**Target**).

$$\boxed{\textbf{Hash(data + nonce) < Target}}$$

This threshold is periodically and automatically updated so that the average interval between two blocks remains ten minutes.

Example: to find a Proof-of-Work on the string **"Hello world!"**, one can compute successive hashes with the SHA256 hash function by adding a higher number each time until one obtains a hash that starts with **000**. This is a raw power work, as there is no mathematical way to deduce which number could achieve such a result.



```
Hello world ! 0 : 3f6fc92516327a1cc4d3dca5ab2b27aeedf2d459a77fa06fd3c6b19fb609106a
Hello world ! 1 : b5690c48c2d0a09481186aaa99e4e090901ff2ac4d572e6706dfd30eefc22a27
Hello world ! 2 : 5b6fd9c27fcb54ca23404d9428f081b7c9280ba6370e33a6a20b16f40ce76320
Hello world ! 3 : 9c5d769416aa0ca894abf22bd17bd30fbb6959291423ae1903a9f86a1fe7ce78
.....
Hello world ! 95 : b74f3b2cf1061895f880a99d1d0249a8cedf223d3ed061150548aa6212c88d43
Hello world ! 96 : 447ca2fa886965af084808d22116edde4383cbaa16fd1fbcf3db61421b9990b9
Hello world ! 97 : 000ba61ca46d1d317684925a0ef070e30193ff5fa6124aff76f513d96f49349d
```

The proof of work was found after **98 calculations**. This calculation is done to validate a block that represents a set of transactions as shown in Figure 1 (see also Figure 2). For each validation there is a reward in bitcoin. The block header has a fixed size of **80 bytes** (see Figure 3). Although this very limited size does not allow storing transactions, the header nevertheless contains the root of a Merkle Tree of transactions which allows to easily verify the presence of a specific transaction in the blockchain without having to download the whole transaction. Thin clients are in most cases content with only downloading the block headers.

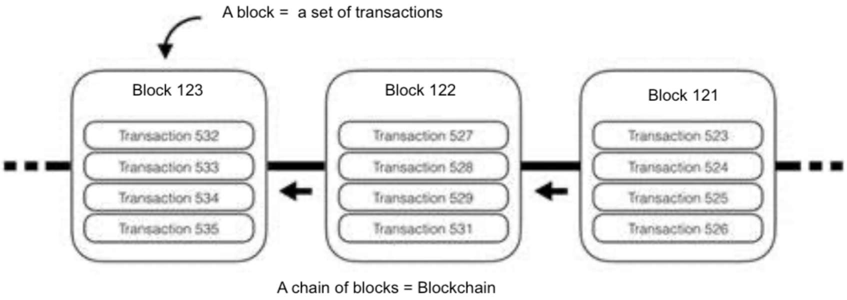
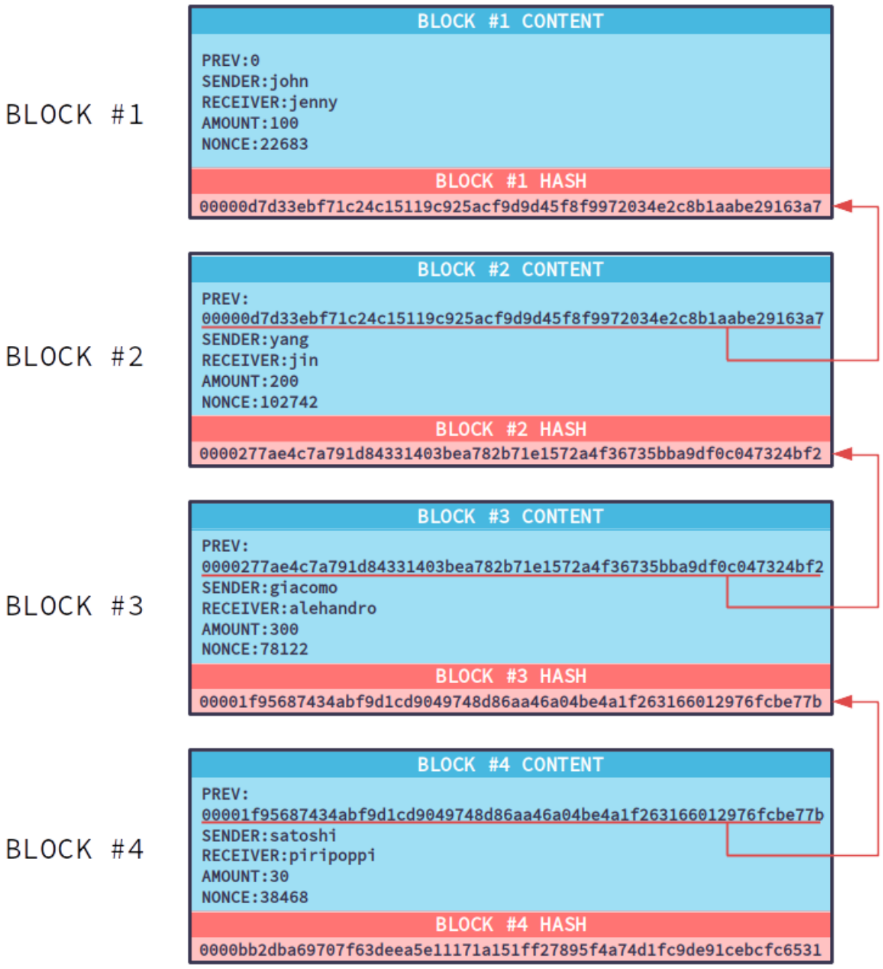
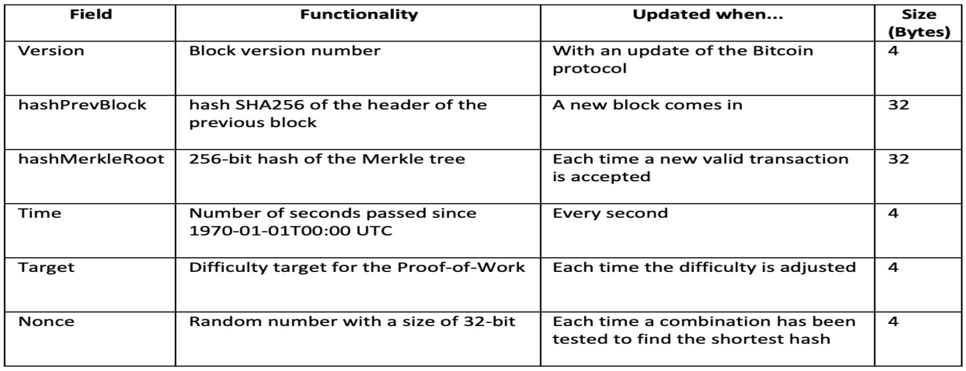


Figure 1: Chain of Blocks or Blockchain



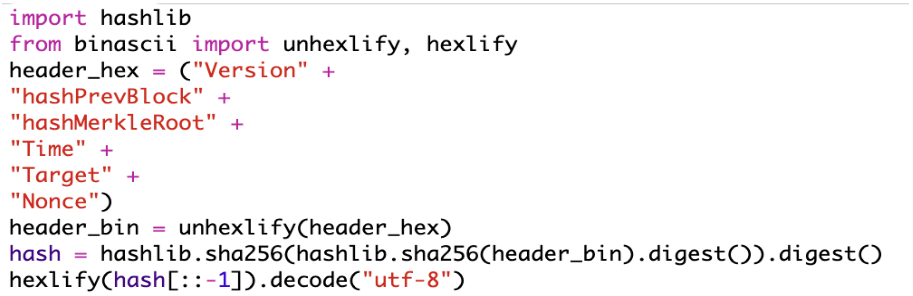
```
BLOCK #1 CONTENT
PREV:0
SENDER:john
RECEIVER:jenny
AMOUNT:100
NONCE:22683
BLOCK #1 HASH
00000d7d33ebf71c24c15119c925acf9d9d45f8f9972034e2c8b1aabe29163a7

BLOCK #2 CONTENT
PREV:
00000d7d33ebf71c24c15119c925acf9d9d45f8f9972034e2c8b1aabe29163a7
SENDER:yang
RECEIVER:jin
AMOUNT:200
NONCE:102742
BLOCK #2 HASH
0000277ae4c7a791d84331403bea782b71e1572a4f36735bba9df0c047324bf2

BLOCK #3 CONTENT
PREV:
0000277ae4c7a791d84331403bea782b71e1572a4f36735bba9df0c047324bf2
SENDER:giacomo
RECEIVER:alehandro
AMOUNT:300
NONCE:78122
BLOCK #3 HASH
00001f95687434abf9d1cd9049748d86aa46a04be4a1f263166012976fcbe77b

BLOCK #4 CONTENT
PREV:
00001f95687434abf9d1cd9049748d86aa46a04be4a1f263166012976fcbe77b
SENDER:satoshi
RECEIVER:piripoppi
AMOUNT:30
NONCE:38468
BLOCK #4 HASH
0000bb2dba69707f63deea5e11171a151ff27895f4a74d1fc9de91cebcfc6531
```

Figure 2: In a Blockchain, Each Block Points to its Predecessor



Field	Functionality	Updated when...	Size (Bytes)
Version	Block version number	With an update of the Bitcoin protocol	4
hashPrevBlock	hash SHA256 of the header of the previous block	A new block comes in	32
hashMerkleRoot	256-bit hash of the Merkle tree	Each time a new valid transaction is accepted	32
Time	Number of seconds passed since 1970-01-01T00:00 UTC	Every second	4
Target	Difficulty target for the Proof-of-Work	Each time the difficulty is adjusted	4
Nonce	Random number with a size of 32-bit	Each time a combination has been tested to find the shortest hash	4

Figure 3: What it Contains a Header of a Bitcoin Block



```
import hashlib
from binascii import unhexlify, hexlify
header_hex = ("Version" +
"hashPrevBlock" +
"hashMerkleRoot" +
"Time" +
"Target" +
"Nonce")
header_bin = unhexlify(header_hex)
hash = hashlib.sha256(hashlib.sha256(header_bin).digest()).digest()
hexlify(hash[::-1]).decode("utf-8")
```

Figure 4: Code in Python

Part I: Understanding the Bitcoin Header

Bitcoin uses SHA256 (SHA256 (Block_Header)) to generate the block hash. You need to generate this block hash to search for the block in question on the website: https://www.blockchain.com/fr/explorer

- The content of the file *header_block1.txt* is the following: *0100000081cd02ab7e569e8bcd9317e2fe99 f2de44d49ab2b8851ba4a308000000000000e320b6c2 fffc8d750423db8b1eb942ae710e951ed797f7affc8 892b0f1fc122bc7f5d74df2b9441a42a14695*

- But you have to pay attention to the byte order because the header in the file *header_block1.txt* is written in **little-endian in hexadecimal notation**.

- Modify the code in python above to calculate the hash of the block. The script should display the hash of the block. Copy this hash and put it in the relevant field on the website: https://www.blockchain.com/fr/explorer

  Steps to follow:

  - Write the script in a file called *script.py*
  - Compile and run the script with the command: *python script.py*
  - When was this block mined? What is the number of confirmations for this block?

Part 2: Implementation of a PoW

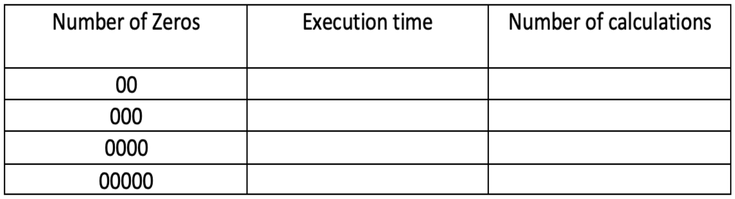
In this part you will mine a real Bitcoin block but with a reduced difficulty. The *babyhash.java* file contains the Proof-of-Work algorithm which can be described as follows:

While (hash value) $>=$ threshold:

- **Calculate the hash of the block header**
- **If (hash value) $<$ threshold, return the hash, else, increment the nonce by 1 (= add 1 to the nonce)**

Complete the *main* function (the loop) and test the correct operation of the program:

- Compilation of the program *babyhash.java* : *javac babyhash.java*
- Execution of the *babyhash* program : *java babyhash*
- After the execution of the previous command, you will be asked to enter the value of the block header in hexa. Copy the content of the file *header_block2.txt* and paste it.
- Fill in the table below by modifying the Java code (in particular the number of x characters in the variable babyHash) and executing it again.



Number of Zeros	Execution time	Number of calculations
00		
000		
0000		
00000		

Figure 5: Table to fill in ...