How does BitCoin work?

•••

What is Computer doing when mining the BitCoin?

Boming Zhang

Who am I?

My name is Boming Zhang. I am from Beijing, China. I came to the U.S. studying at Hampshire College in 2015. Currently, I am a PhD student at UMass focusing on A.I. Education.

Outline

1. Why BitCoin?
2. The Digital Signature
3. Cryptographic hash function
4. Let’s mine some UMassCoin!

Why BitCoin?

1. BitCoin is digital currency which can be used without a central bank
2. Over $600,000,000,000 involved in the BitCoin business
3. BitCoin could be a mainstream currency in the future
4. New platform for application

Centralized system: a ledger

Ledger

Alice pays Bob $20

Bob pays Charlie $40

Charlie pays You $30

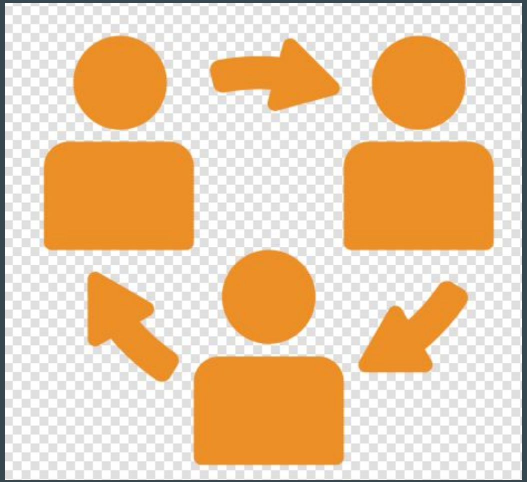
You pay Alice $10

Why BitCoin?

1. Less control from central bank system
2. High fee for transaction
3. No Inflation

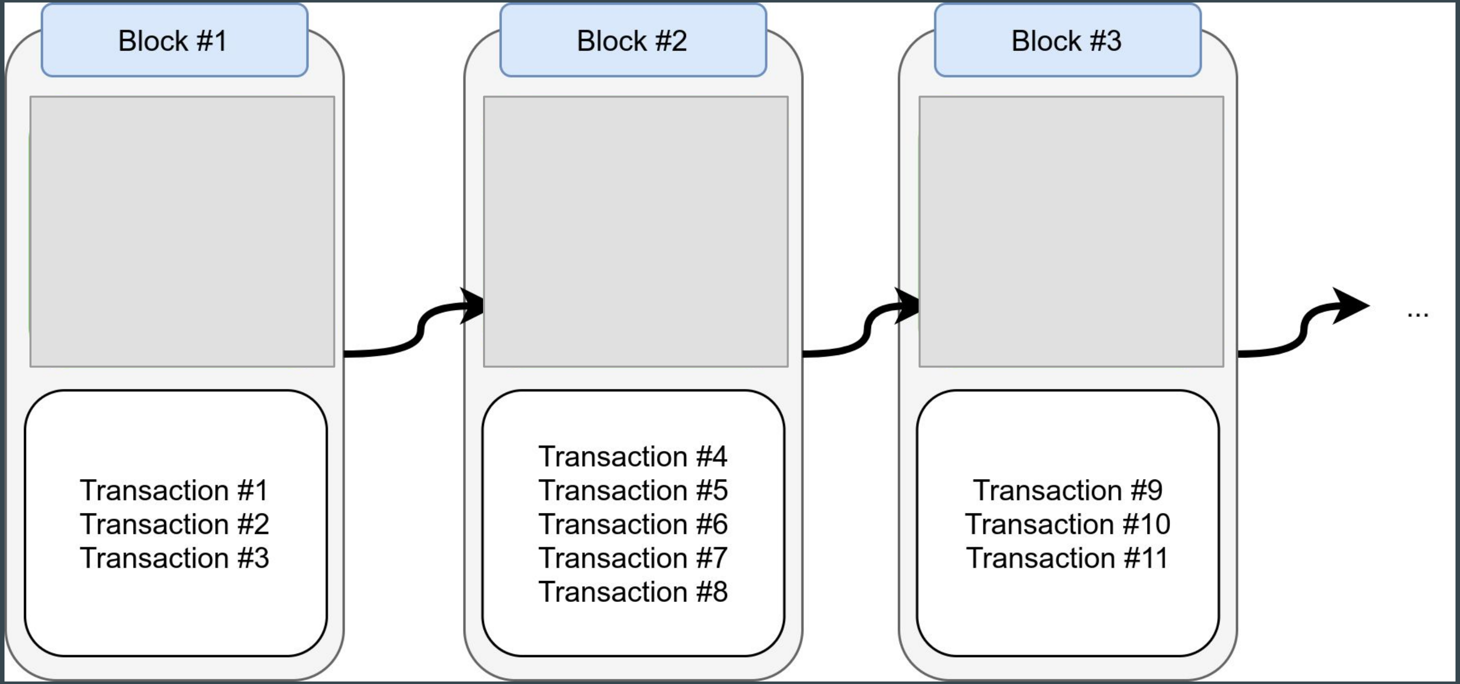
A basic solution:

Everyone takes turn to track the ledger



A basic solution:

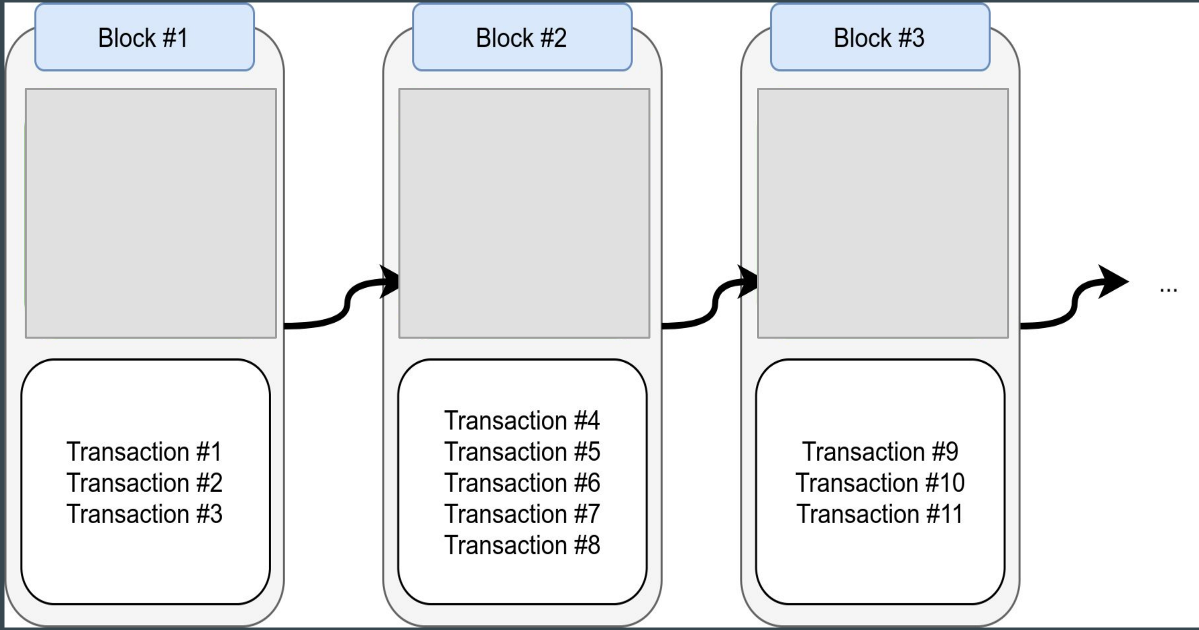
Each turn will create a new block for the ledger



Let's create a new Crypto currency

UMassCoin

1. Two volunteers(plus me) take turns to track the transaction
2. Each of us will track sender, receiver, amount of money, and transaction ID
3. When switching person to keep Transaction, create a new block



Transaction List

Day 1

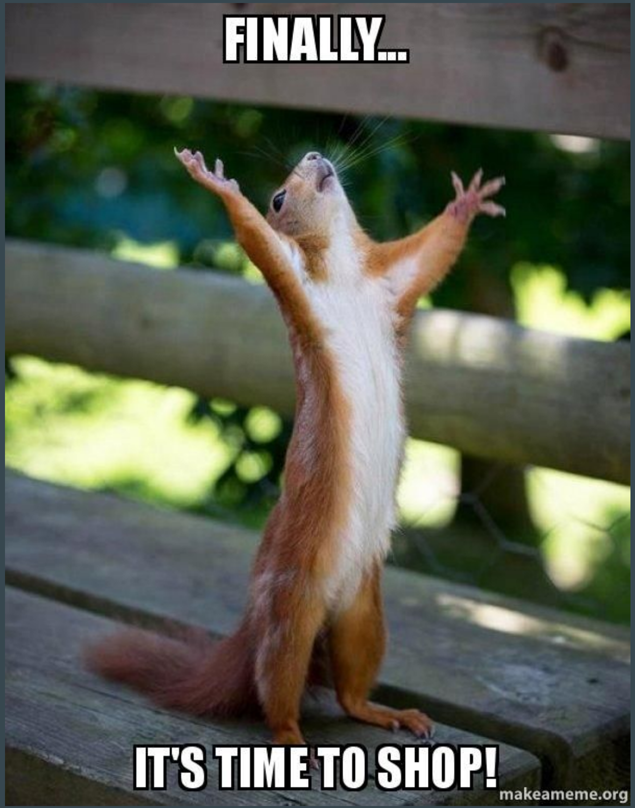
1. A sent B 10 UMassCoin for vegetables
2. C sent B 20 UMassCoin for vegetables

Day 2

3. B sent A 50 UMassCoin for meats
4. C sent A 20 UMassCoin for meats

Day3

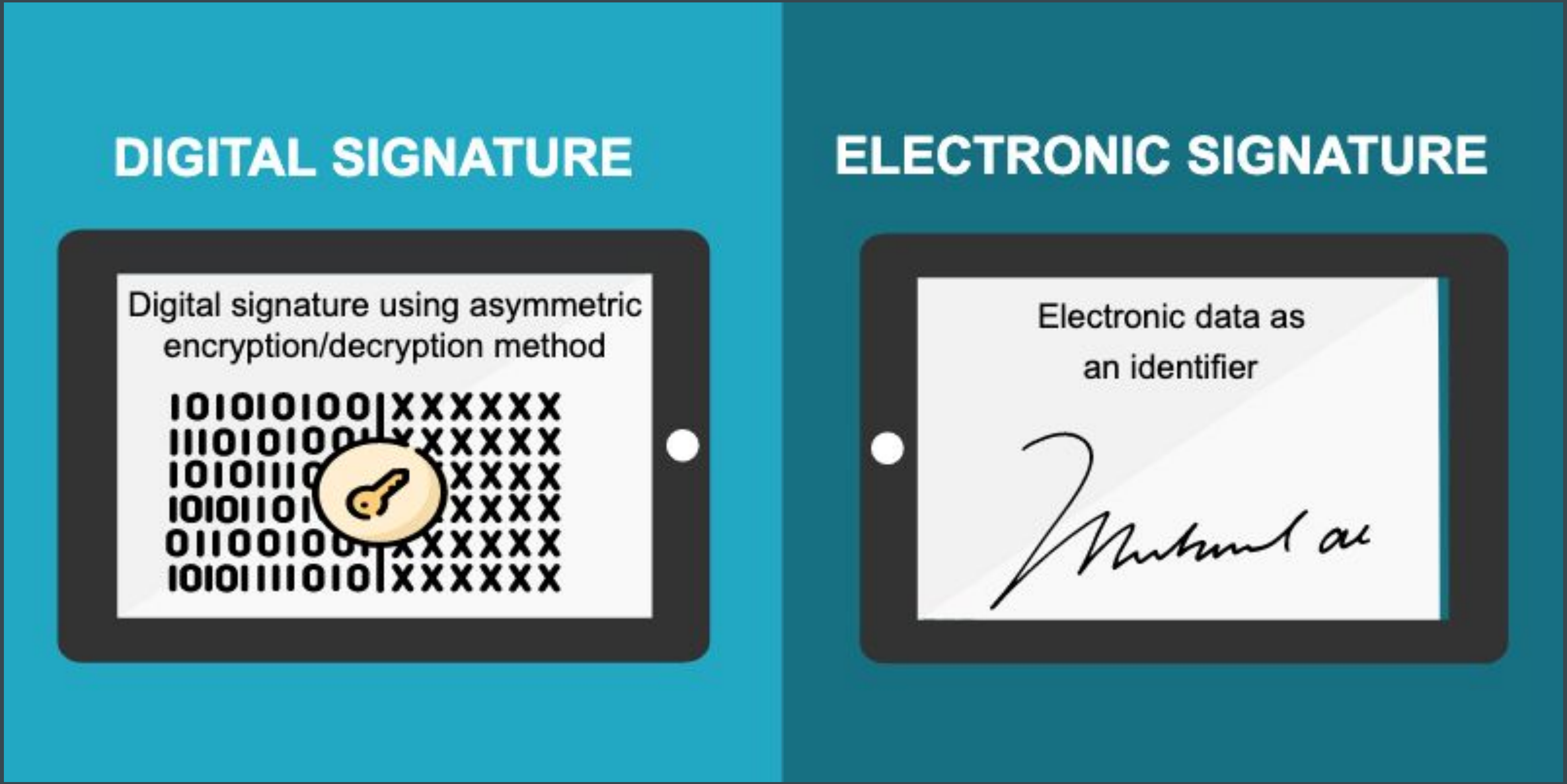
5. A sent C 10 UMassCoin for fruits
6. B sent C 20 UMassCoin for fruits



Problems encountered so far

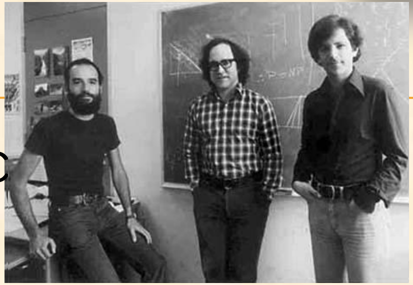
1. Transaction can be easily falsified
2. The tracking person may be dishonest
3. Previous transactions can be easily modified

Digital Signature



The Asymmetric Algorithm: RSA

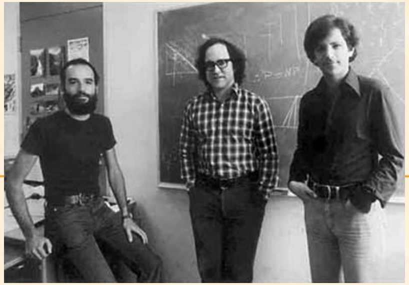
- The most widely accepted and implemented public encryption
- Invented by Ron Rivest, Adi Shamir, and Len Adleman in 1977



[Shamir, Rivest, Adleman 1977]

Passage of time…!

RSA Cryptosystem



[Rivest, Shamir, Adleman 1977]

- Key generation:
  - Generate large primes p, q
  - Compute $n=pq$
    - Note that $\phi(n)=(p-1)(q-1)$
  - Choose small e, relatively prime to $\phi(n)$
  - Compute unique d such that $ed \equiv 1 \bmod \phi(n)$
  - Public key = (e,n); private key = d
- **Encryption** of m: $c = m^e \bmod n$
- **Decryption** of c: $c^d \bmod n = (m^e)^d \bmod n = m$

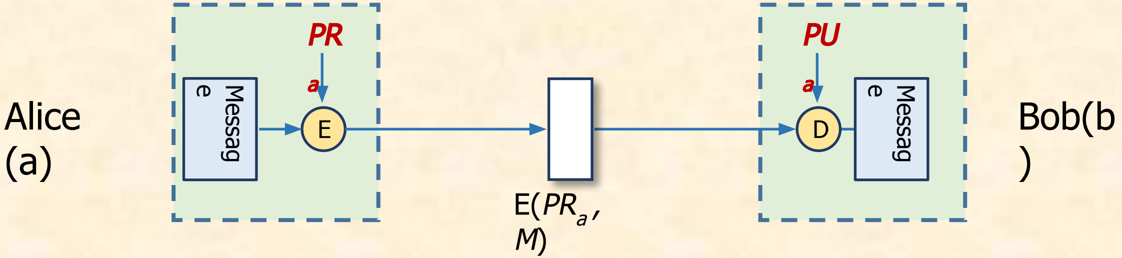
Let’s create a RSA key pairs!

Why Is RSA Secure?

- Factoring problem: given positive integer n, find primes $p_1, \ldots, p_k$ such that $n = p_1^{e1} p_2^{e2} \ldots p_k^{ek}$
- If factoring is easy, then RSA problem is easy (why?)
- You have to try 1.88×10^{302} this many numbers
- You have to try for 5.95×10^{211} this many years

Digital Signature Basic Idea

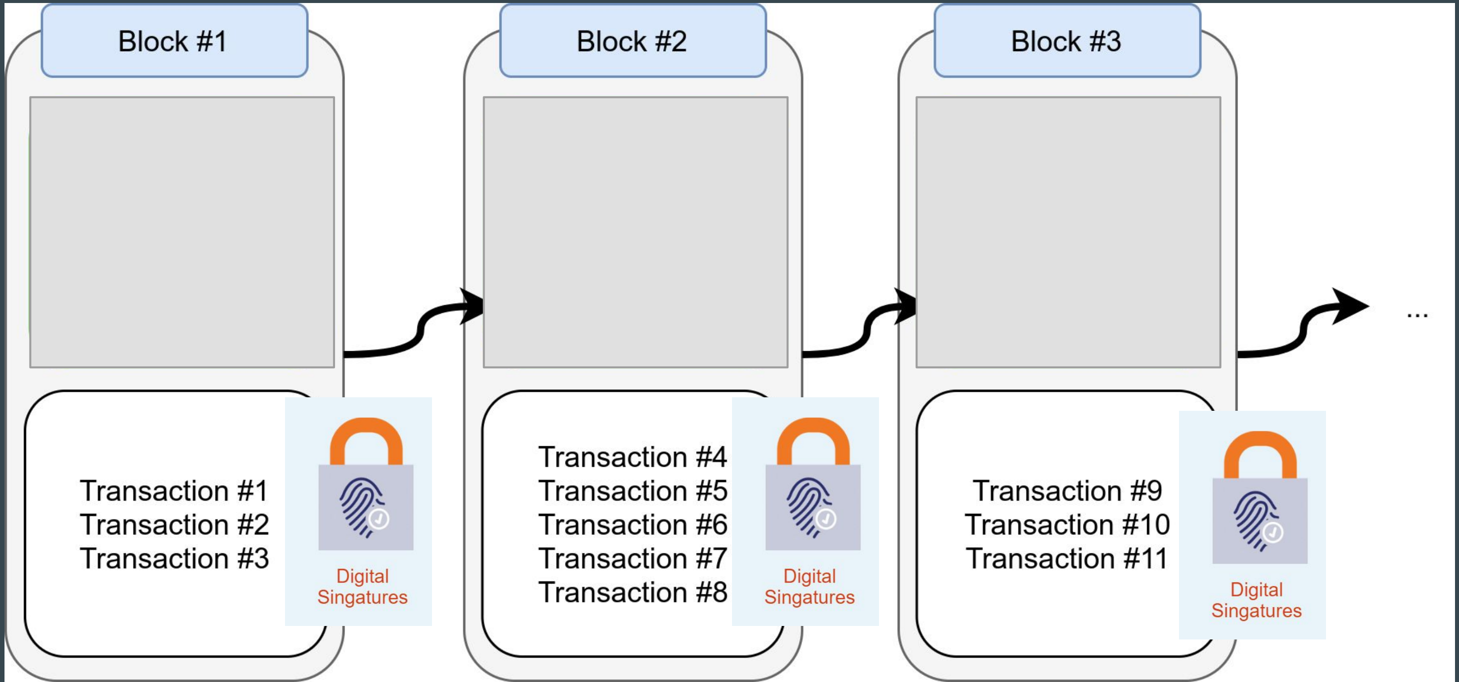
- Given:
  - Everybody knows Alice's public key
  - Only Alice knows the corresponding private key



Let’s try using RSA!

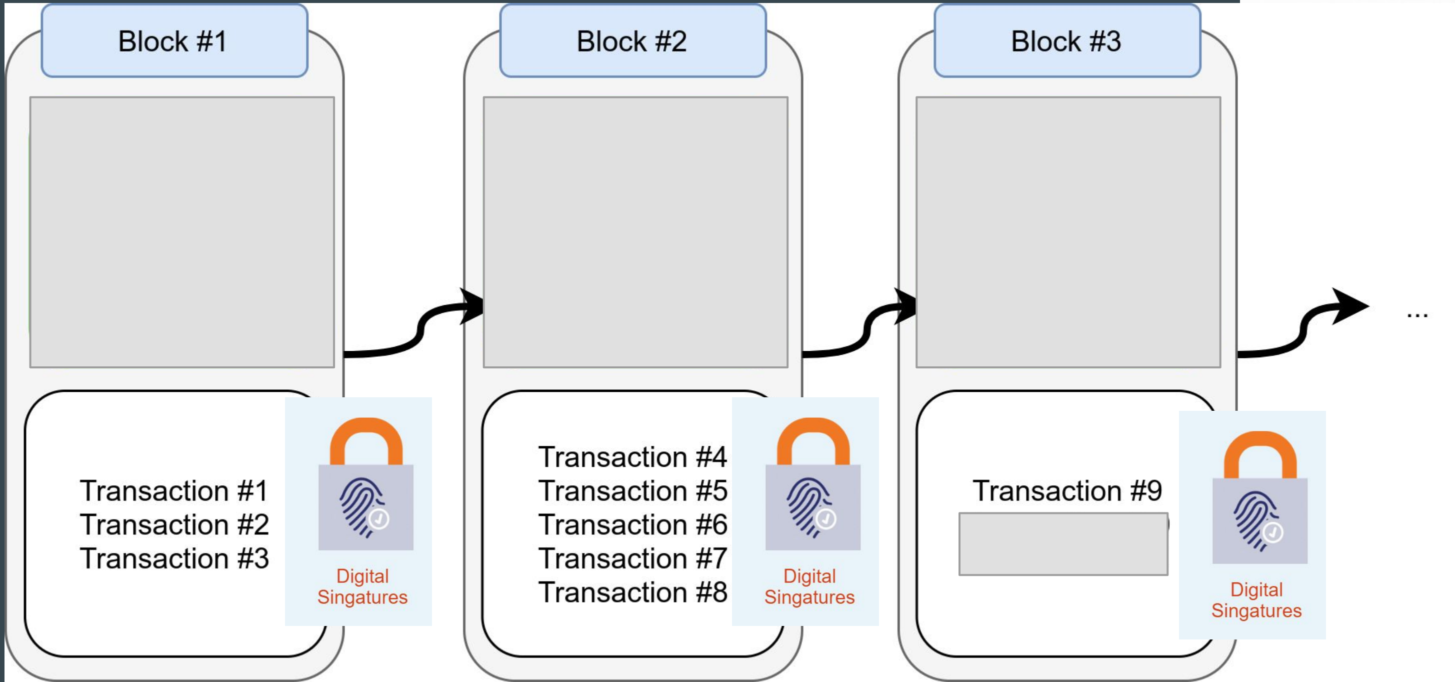
A better solution:

1. Each turn will create a new block for the ledger with
2. Digital signatures at the end of each transaction



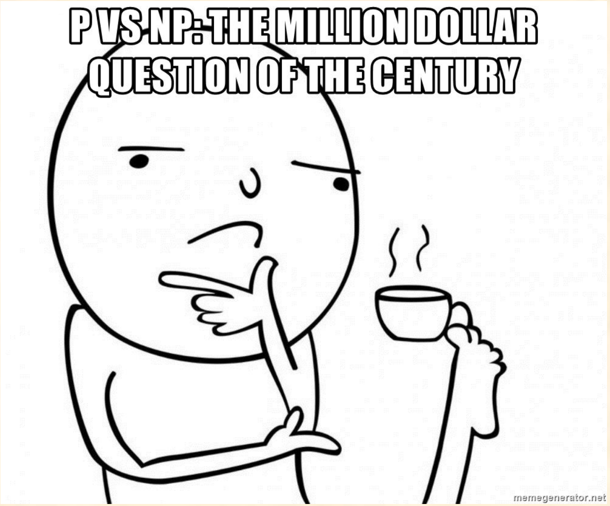
If I know I will be incharge of block #3......

Wiping out my own transaction!



One-way Functions

- An One-way functions is easy to compute the output given the input, but hard to compute the input if only given the output
  - 2^x mod 7 = Y or f(x) = 2^x mod 7
  - If I tell you x = 4, you can quickly compute Y = 2^4 mod 7 = 16 mod 7 = 2
  - If I tell you Y = 6, can you quickly compute X=?



Hash function

1. Hash function is a one-way function.
2. Hash function takes any-length input and return fixed-length output.
3. If one bit changed in the input, the output will change drastically.

SHA-256 example

SHA-256 stands for Secure Hash Algorithm with 256-bit output

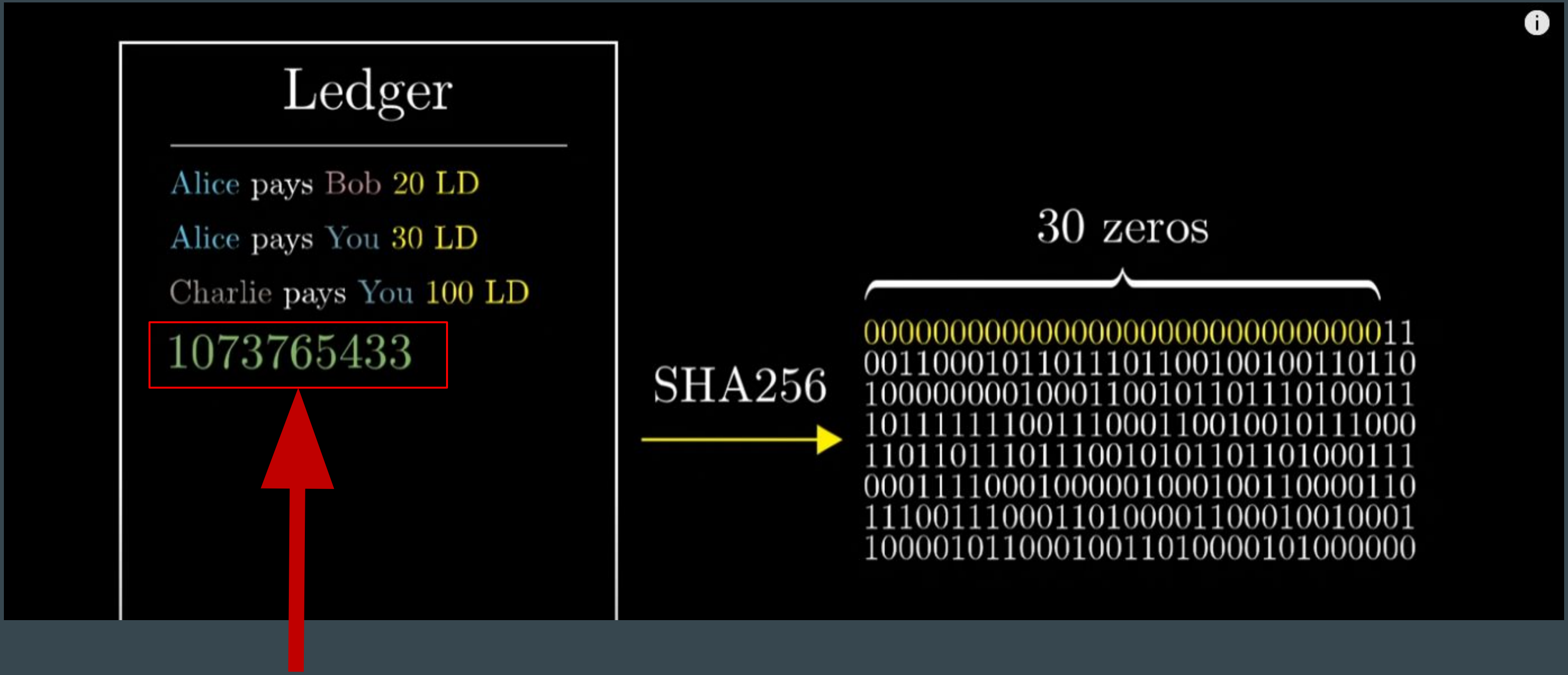
INPUT DATA	HASH OUTPUT (SHA-256)
My name is Toby	cacb5418163039b016be9746818a2926f68fd1e4bad1b04f6791 f6aabb5e8c52
My name is Tony	9cd2444dc56929bdb97123add1f007643effa88bf1ed061eee1e ead4e15ac7f9
My name is Toby and this is my project	9abbaa0c54fcd028ac51bede2608d06e8d3a026784e34adfac14 fadd143d212c

Proof of work using Hash function

- Miner needs to find a “magical number”
- The “magical number” will result a unique hash for the transaction block
- The unique hash value starts with 30 zeros

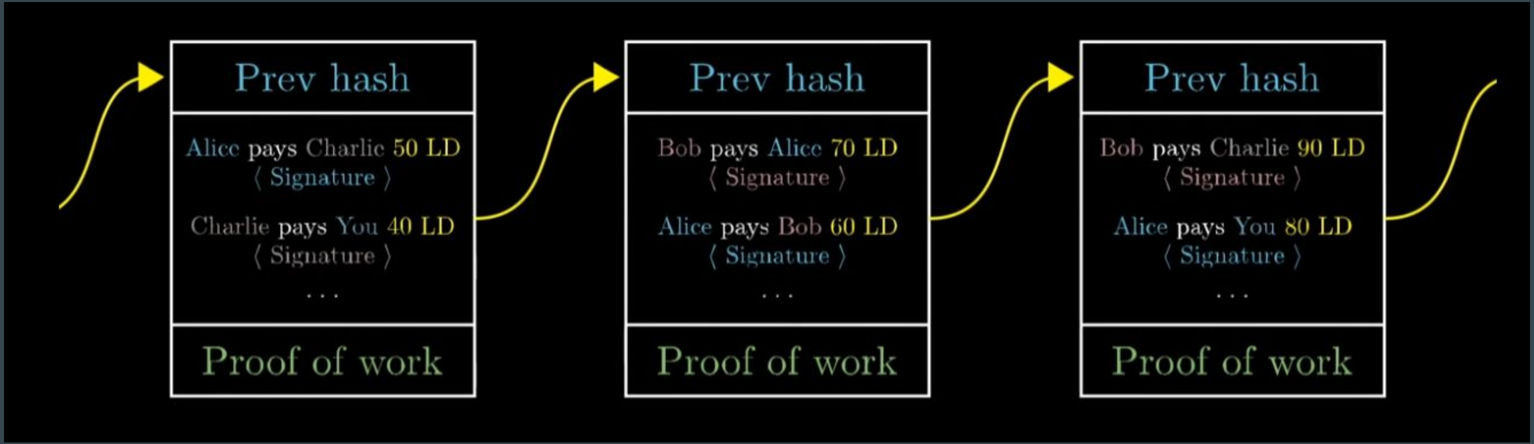
The magical number

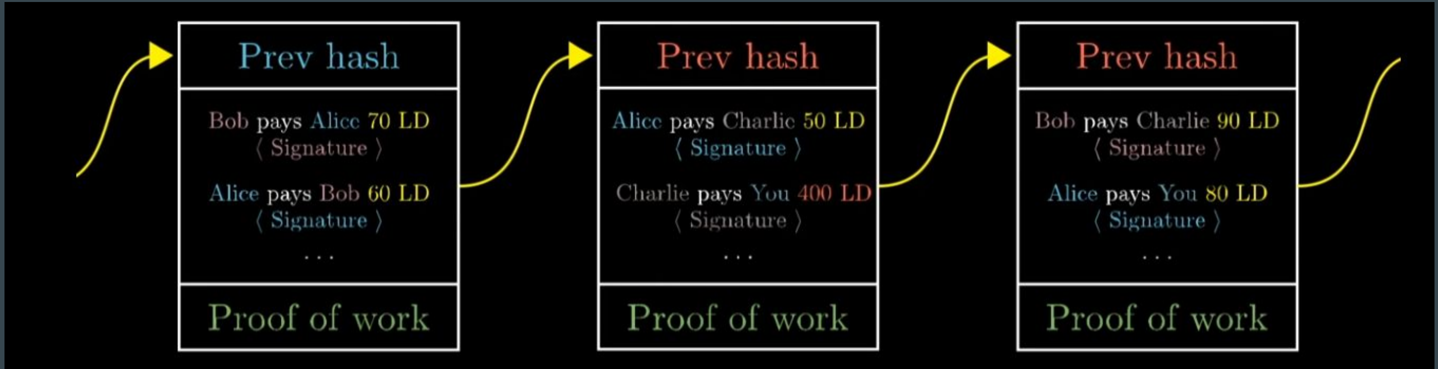
Proof of work using Hash function



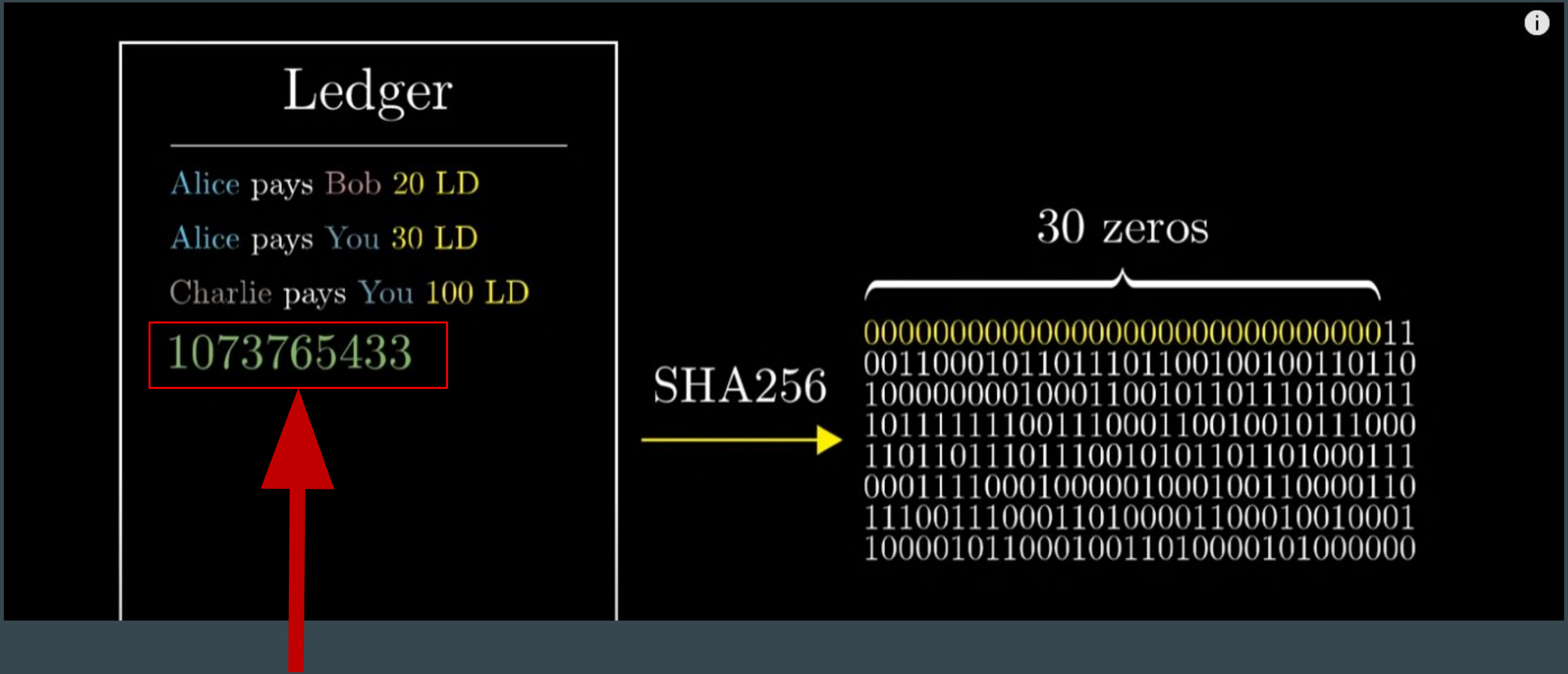
The magical number

Proof of work using Hash function





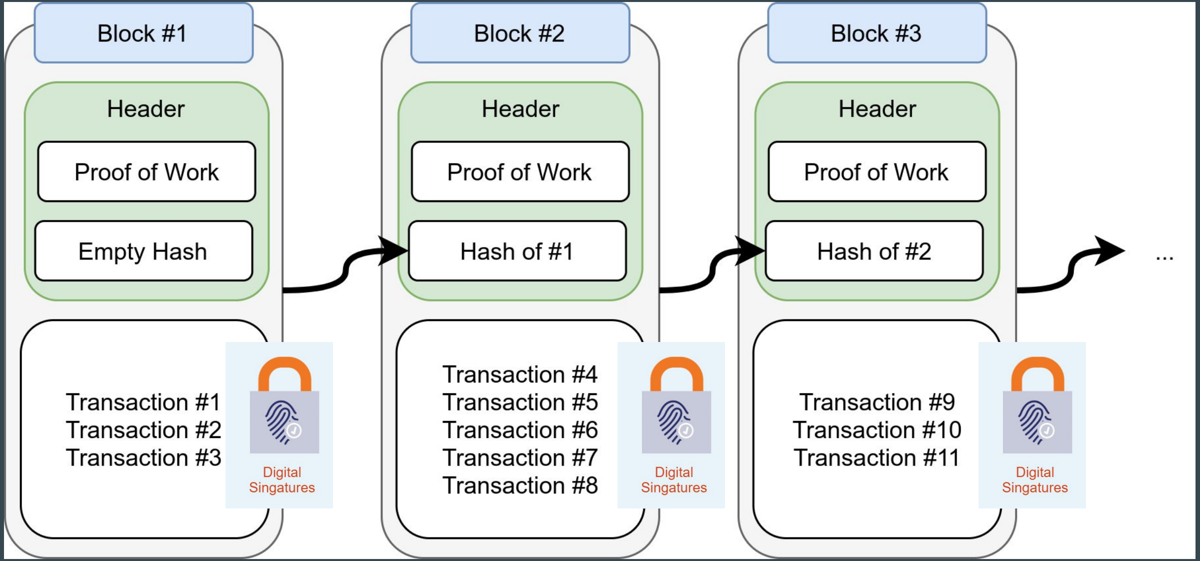
Proof of work using Hash function



The magical number

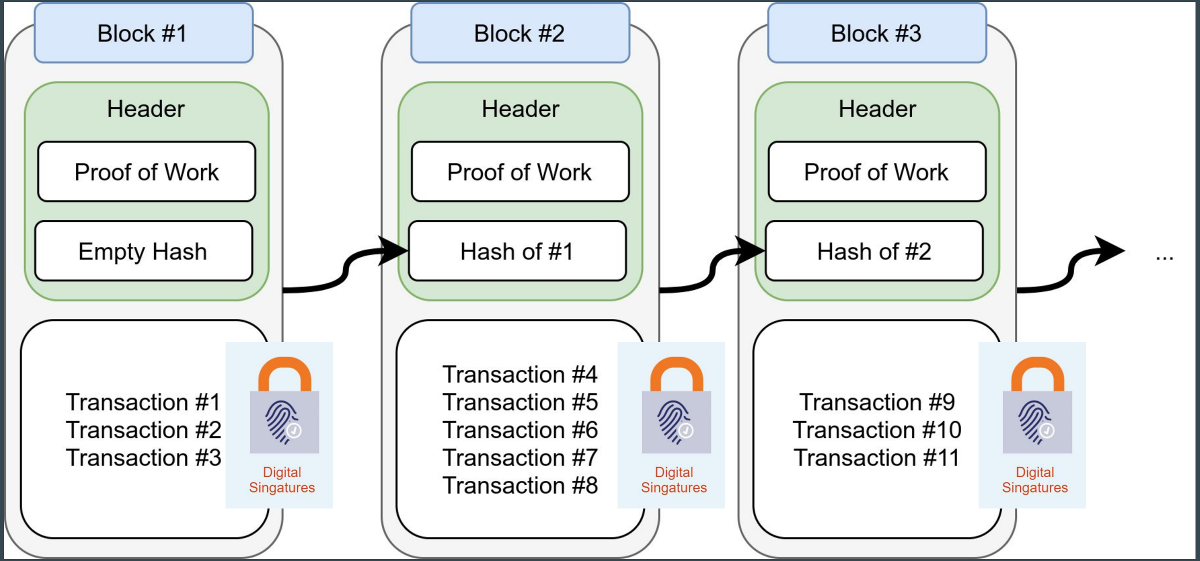
The BitCoin solution:

1. Each turn will create a new block for the ledger
2. Digital signatures at the end of each transaction
3. Using proof of work to keep authenticity
4. Chaining hash together against tampering history



Let's dig some UMassCoin!

1. Use RSA to produce your Public/Private key pairs
2. Use SHA256 to produce proof of work
3. Use this google sheet to log your public key
4. Search the magic number



Recap

- RSA gives the digital signature of transaction
- Hash function gives the proof of work
- A reward coin goes to the first miner who get the magical number
- Chaining Hash makes it impossible to alter the previous history

Further reading

- What is the upper limit for all the BitCoin combining together?
- How does the system decide how many zeros is needed for the hash value?

Sources:

1. https://medium.com/coinmonks/the-blockchain-473aac352e5
2. https://www.youtube.com/watch?v=bBC-nXj3Ng4&t=1318s
3. https://www.jinse.com/blockchain/65436.html
4. https://www.coingogo.com/article/92306