Build your own blockchain from scratch

Sanket Shanbhag

Intro

What you will learn:

- How to think and reason about Blockchains.
- How to design a Blockchain.
- How to implement a cryptocurrency in python.

What you won’t learn:

- Advanced implementation details and optimizations.
- How to get rich trading crypto.

Prerequisites

- Literally nothing!
- Familiarity with basic cryptography and hashing concepts.
- Basic familiarity with python is assumed.
- Python3 with the Flask and requests libraries installed.
- Postman for testing API calls.

How would you design a digital currency?

How would you design a currency?

Money

- Medium of exchange.
- Measure of value.
- Standard of deferred payment.
- Store of value.

Money - properties

- Fungibility: individual units must be interchangeable.
- Durability: able to withstand repeated use.
- Portability: easily carried and transported.
- Cognizability: its value must be easily identified.
- Stability of value: its value should not fluctuate.

The ledger

---

1. We can keep track of all spending on a single document called the ledger.
2. Every person writes down how much they want to pay to any other person.
3. Eg. “Alice gives Bob 10 coins”
4. This **intent itself is the information** which represents the currency.

Initial Design

Suppose some person - let’s call her Alice - has some digital money which she wants to spend. If we use some digital data to represent coins, we run into a few problems:

1. Identity - Was it really Alice who created this message?
2. Forgery and Infinite money - Anyone can duplicate the bits.

Cryptography

Cryptography

---

1. Digital Signature:
   - Generate secret and public keys.
   - sign(Message, secret_key) => signature
   - verify(Message, signature, public_key) => True/False

The security comes from the fact that there is no good way to break the sign function than just random guessing (There is no known way to calculate the inverse except brute force).

Solving identity problems

- Alice can sign her message with her secret key.
- “Alice gives Bob 10 coins” + “Alice_signature”
- Anyone in the world can verify it was Alice who created the message. This solves the identity problem.

Solving the forgery problem

- Attach each coin with a unique serial number.
- “Alice gives Bob 10 coins srno. 8234” + “Alice_signature”

However this breaks our assumptions about intent. We now need a way to uniquely identify and create coins.

The Bank

Introducing the bank

We need a way to make each coin be uniquely representable by a serial number.

The conventional way to do this would be to give control of this to a bank, which issues the coins and would be a trusted source of serial numbers.

Guarantees that the money will have value.

But this is too conventional, we can do better.

Introducing the “collective” bank

- Key Idea: Make everyone the bank.
- Everyone using our coins keeps a complete ledger of which coins belong to which person.
- Now, suppose Alice wants to transfer a coin to Bob. She signs the message “Alice gives Bob 1 coin srno 1234567”, and gives the signed message to Bob.
- Bob can use his copy of the ledger to verify the transaction and then broadcasts it further if it is valid

From people to computers

- Do everything we just described by algorithms
- Every person is now a “node”
- Broadcasts and updates happen over the network and on every node’s internal storage

More problems

- Where do the serial numbers come from?
- Double spending - Alice gives the same coins to Bob and Charlie and broadcasts this to different nodes together.

Double Spending

- Made worse due to network delays.
- Dummy identities can automated by bots.

Solving double spending

---

1. Bob shouldn’t try to verify the transaction alone. Rather, he should broadcast the possible transaction to the entire network of nodes, and ask them to help determine whether the transaction is legitimate.
2. If collectively the network decides the transaction is OK, Bob can accept the transaction.

Proof of Work

Proof of Work

---

1. Artificially make it computationally costly for network users to validate and add new transactions; and
2. Reward them for trying to help validate transactions. The reward is used so that people on the network will try to help validate transactions, even though that’s now been made a computationally costly process.

Proof of Work

---

1. Every node listens for transactions and adds it to a list of transactions.
2. A node checks their copy of the Blockchain, and can see that each transaction is valid for their copy.
3. As part of the validation protocol the node is required to solve a hard computational puzzle - the proof-of-work.

Hashing

Hashing

---

1. Way to convert any digital information to a fixed length string.
2. The ideal cryptographic hash function has the following main properties:
    a. Deterministic, same message always results in the same hash
    b. Quick to compute the hash value for any given message
    c. Infeasible to generate a message that yields a given hash value
    d. Infeasible to find two different messages with the same hash value
    e. Avalanche effect -> small changes in message will lead to large changes in hash value

Using hashing to create computational puzzles

---

1. h("Hello, world!0") =
   1312af178c253f84028d480a6adc1e25e81caa44c749ec8197619...
2. h("Hello, world!1") =
   e9afc424b79e4f6ab42d99c81156d3a17228d6e1eef4139be78e9...
3. h("Hello, world!4250") =
   0000c3af42fc31103f1fdc0151fa747ff87349a4714df7cc52ea9...

Incentives

---

1. Nobody would do this just for the sake of the network.
2. We add an incentive to make nodes want to validate the transactions.
3. Every node who finds a solution to the problem gets to add another transaction to the ledger, the “coinbase transaction”
4. In this transaction, the node is allowed to give itself coins as a reward for performing the validation.
5. In bitcoin, this is called “mining”

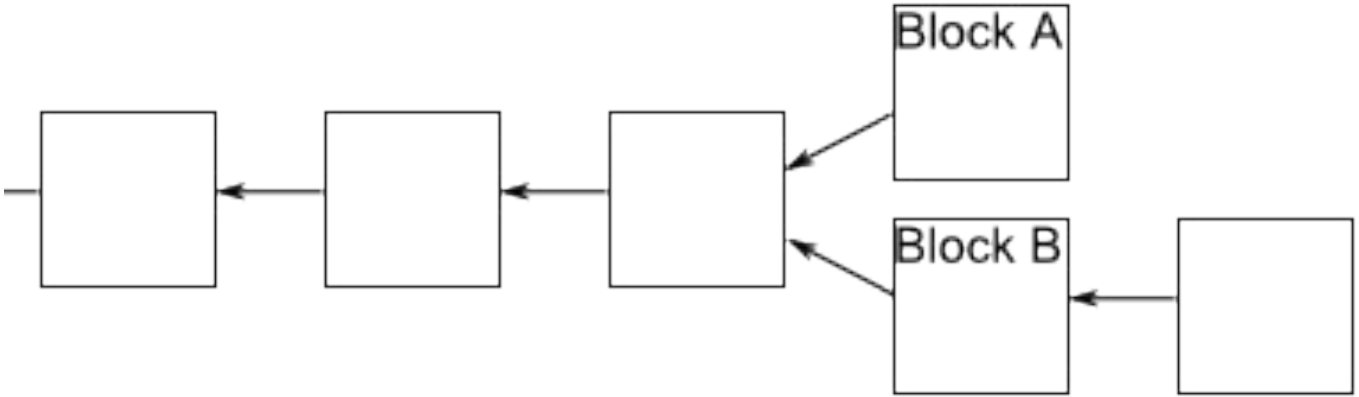
From Ledgers to Blocks

- Instead of repeatedly updating a single ledger, we break it down into chunks of transactions called “blocks”.
- Each block holds the list of transactions as well as some metadata. This metadata includes the nonce.
- To enforce ordering of transactions, each block holds the hash of its previous block, effectively linking it to its predecessor and creating a “chain”.

Proof of Work revisited

- Proof of work can be thought of as a competition.
- Each miner owns resources proportional to his computing power in relation to the entire network.
- So, in theory a dishonest miner can only manipulate the network if he controls 51% of the total network’s resources.

Forks



From forks to confirmations

In Bitcoin, a transaction is not considered confirmed until:

(1) it is part of a block in the longest fork, and

(2) at least 5 blocks follow it in the longest fork.

We have solved Double Spending?!

A Bitcoin transaction and serial numbers

Field	Description	Size
Version no	currently 1	4 bytes
Flag	If present, always 0001, and indicates the presence of witness data	optional 2 byte array
In-counter	positive integer VI = VarInt	1 - 9 bytes
list of inputs	the first input of the first transaction is also called "coinbase" (its content was ignored in earlier versions)	<in-counter>-many inputs
Out-counter	positive integer VI = VarInt	1 - 9 bytes
list of outputs	the outputs of the first transaction spend the mined bitcoins for the block	<out-counter>-many outputs
Witnesses	A list of witnesses, 1 for each input, omitted if flag above is missing	variable, see Segregated_Witness
lock_time	if non-zero and sequence numbers are < 0xFFFFFFFF: block height or timestamp when transaction is final	4 bytes

Tying up a few loose ends

- What is an address?
- It’s the public key you generate when creating your digital signature
- Intended to be used as a single use token.

Block reward and transaction fees

- Initially, the block reward was set to be a 50 bitcoin reward. But for every 210,000 validated blocks (roughly, once every four years) the reward halves.
- Bitcoin also makes it possible to add a transaction fee, which goes to the miner who helps validate it.
- “In a few decades when the reward gets too small, the transaction fee will become the main compensation for [mining] nodes. I’m sure that in 20 years there will either be very large transaction volume or no volume.”

“Talk is cheap. Show me the code.”

- Linus Torvalds

Sources and Credits

- Original Bitcoin paper: https://bitcoin.org/bitcoin.pdf
- Blog post by Michael Nielsen: https://goo.gl/BW1RV3
- Blog post by Daniel van Flymen: https://1n.pm/L5C8V
- Ethereum white paper: https://goo.gl/XXZddT
- Video by 3Blue1Brown: https://youtu.be/bBC-nXj3Ng4

Thanks!

Contact me:

sanketshanbhag@gmail.com