Blockchain CheatSheet - Technical Use

Read Time: 5 m

Table of Contents

§ Addresses

- Use Cases
- The Steps

§ Cryptotransactions

- Analogy
- Mechanics of Transactions
- Validation of the Proposal
- Cryptotransactions in Depth

§ Scalability

- Layers
- Layer 2 Lightning Network

---

§ Addresses

Use Cases

- Sign transaction with public key to Identify and Validation of data.
- Anyone that possesses the public key can Identify and Validate data.

The Steps

1. Generating Key Pairs:
   - Create ***Private Key*** :: `256 bit or 64 hexadecimal Chars`
     Randomly.
   - Derive ***Public Key Base*** :: `512 bits or 128 Hexadecimal Chars`
     We use the ***Private Key*** with the *Elliptic Curve Digital Signature Algorithm* (Algorithm => x_coordinate-256bits + y_coordinate-256bits = 512 bits ***Public Key Base***).
2. Hashing (Ethereum):
   - Hash ***Public Key*** :: `From 512 bits to 256 bits or 64 Hexadecimal Chars`
     Hash the ***Public Key Base*** with *Kekkak-256* or *Sha-3*.
3. Generating Public Address (Ethereum):
   - Create ***Public Address*** :: `From 64 Hexadecimal Chars to 42 Hexadecimal Chars`
     Take last 40 Hexadecimal Chars (20 bytes) and prefix with 0x to 42 Hexadecimal Chars.

---

§ Cryptotransactions

Analogy

Suppose that the parties **A**, **B**, and **C** each have a *lockbox* which contains content that travels through the Blockchain Protocol System, which enforces the rules of how everything works. These *lockboxes* have a slot that only accepts inward content, and the only way to retrieve the content is with the private key of the owner.

Mechanics of Transactions

A Sends -> to B Data or Cryptocurrency

1. **B** Creates ***Public Address*** and ***Public Key*** from ***Private Key***:
    - **B *Private Keys*** :: **B *Public Address*** and ***Public Key***
2. **B** Sends the ***Public Address*** -> to **A** (*Public address can change for every transaction*).
    - **B *Public Address*** -> to **A**
3. **A** will add the ***Public Address*** of **B** and the data or amount to a "Transaction" Message:
    - **A** Initialize Transaction :: **B** Public Address and Content
4. **A** will Sign the transaction with the ***Digital Signature***:
    - ***Digital Signature*** :: Derive from **A**'s own Private Key
      With the *Elliptic Curve Digital Signature Algorithm* (x_coordinate-256bits + y_coordinate-256bits).
5. **A** Transaction is <u>*Proposed*</u> by the blockchain protocol in the *Memory Pool*:
    - ***Validation*** :: Miners attempt to validate the transaction by including it in a block from the memory pool.

Validation of the Proposal

B then Sends -> to C

- We need to check before imprinting the transaction in the Blockchain that B has effectively the content necessary to be sent again: **B**'s Transaction is Sent -> to **Blockchain Memory Pool** then the Protocol Sends -> to **C**

Bitcoin Cryptotransactions in Depth

Bitcoin vs Ethereum

- Bitcoin: We need to think Every transaction as a container of an unique unspent cryptocurrency which is not mixed with the others.
- Ethereum: Different from Bitcoin has an accounting system which keeps track of the total balance.

Transaction Management

Cryptocurrency as it is linked to transaction containers
that we will name *X*-Trsct-C*n* (*X* = ID, Trsct = Transaction , C*n* = Container Number)
then need to be accessed manipulating the container to handle it.

A Sends 10 Bitcoin -> to B from a Transaction container that has 20 Bitcoin

1. **B** Creates ***Public Address*** and ***Public Key*** from ***Private Key***
2. **B** Sends the ***Public Address*** -> to **A** (*Public address can change for every transaction*)

3. **A** will add the ***Public Address*** of **B** and the amount to a "Transaction" Message
4. The *New empty Transaction container* (**A**-Trsct-C4) will Take an input and will send one or two outputs, the import and the eventual change:
   - The input is based on the Transaction containers that has the unspent Cryptocurrency or UTXO (Unspent Transaction Output) which covers the import of The New Transaction

   ```
   A-Trsct-C1 = 10 Bitcoin
   A-Trsct-C2 = 30 Bitcoin  -> Input
   A-Trsct-C3 = 5 Bitcoin
   ```

   - The first output will be the import of The New Transaction

   ```
   A-Trsct-C4  = 20 Bitcoin -> Output to B-Trsct-C1
   ```

   - The Optional output will be the change which is sent back to the sender A

   ```
   A-Trsct-C4 = 10 Bitcoin -> Output to A-Trsct-C4
   !!!
   A-Trsct-C2 = 30 Bitcoin is then Destroyed
   ```

5. **A** will Sign the transaction with the ***Digital Signature***
6. **A** Transaction is *Proposed* by the blockchain protocol in the *Memory Pool*
7. Validation of the *Proposal*

Miners Proof of Work Validation

Diagram

```
          Transactions/data -> 80 byte Group of Transactions
                                                                        v
                                                               Block Header
                                                                        |
                                                                        v
                                                             * Nonce Research
                                                                        |
                                                                        v
                                                            Hash algorithm
calulus
                                                                        |
                                                         ----------------------------
```

```
-----
                                                |
|
                                   Valif Hash                    Non-valid
Hash
                                                |
|
                   # Block validated and added           * Nonce increment
                           to the Blockchain                   Repetition
of
                          & Miner's rewarded                      Process
```

Blockchain Block Structure

In the context of blockchain, miners create blocks with a specific structure. A typical Bitcoin block header is 80 bytes and includes the following:

- 4 bytes: version number
- 32 bytes: previous block hash
- 32 bytes: Merkle root (hash of the transactions in the block)
- 4 bytes: timestamp
- 4 bytes: difficulty target
- 4 bytes: nonce

> Usually, the only differences between miner's hashing attempts are:
>
> - The hash of the data (the first part of which is the reward for the miner).
> - The timestamp (which can vary not only by location but also by the number of attempts to find the nonce).
> - The nonce itself.
> - Additionally, the order in which the data is grouped can vary between miners.

---

§ Scalability

Layers

- **Layer 0** : Internet as we know it.
- **Layer 1** : Blockchain Layer 1 transactions are slow more than the traditional methos up to 10 minutes for a settlement.
- **Layer 2** : Wallet for small transactions faster.

Layer 2 Lightning Network

Description: The Lightning Network is an off-chain solution that functions as a payment channel, built on a network structure that connects users. It enables transactions to be processed without recording every transaction on the Bitcoin blockchain.

What it Solves: It significantly increases transaction speed by using a double signature system as an agreement of exchange.

How it Works: Whenever customers need to make a payment, both parties send a transaction as described in the Transaction Management section. The sender sets the amount due, and the receiver sets a transaction with a near-zero value. The payment request travels through the network, seeking the shortest path of connected channels to reach the recipient. Each channel holds a balance that can be transferred between the parties involved, and only the opening and closing of channels are recorded on the blockchain.

Suggested Follow-up

Blockchain CheatSheet - Consensus

Author: Kenneth Boldrini