Cybersecurity Bootcamp | 42 Madrid

ft_blockchain

Summary: Cryptography and decentralization

Version: 1

Contents

Chapter I

Introduction

The objective of this project is to create a blockchain based on a `(Proof of work)` algorithm. To do this, you will have to implement the logic of the block chain, as well as a server through which you can interact with it.

Chapter II

Prologue

A purely peer-to-peer version of electronic cash would allow onlinepayments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

Satoshi Nakamoto, 2012

`0x000000000019d6689c085ae165831e934ff763ae46a2a6c172b3f1b60a8ce26f`

Chapter III

Instrucciones generales

For this project you can use any programming language. You can use cryptographic libraries like `openssl` or `hashlib` for generating `hashes`, but the blockchain structure must be implemented by you. In the same way, a web framework like `NestJS` or `Flask` can be used for the server implementation.

```
block =
{
    'index': 4,
    'timestamp': 1644045050.00042,
    'transactions': [
        {
            'sender': '4c6e7e2a9f2f7f7ff8e7d3d6c8b7c6e8e23a7',
            'recipient': 'b3c6e7e2a9f2f7f7ff8e7d3d6c8b7c6e8e23a7',
            'amount': 42,
        }
    ],
    'proof': 324984774000,
    'previous_hash':
        '084c799cd551dd1d8d5c5f9a5d593b2e931f5e36122ee5c793c1d08a19839cc0',
}
```

Example of a block. The hash of the above block has been generated using the SHA-256 algorithm.

Chapter IV

Mandatory Part

The workflow is to add different transactions to the current block and mine the block so that the chain is added.

The `proof-of-work algorithm` should be simple, for example, finding the number that, concatenated with the previous proof-of-work, matches the result of the SHA-256 hash ending in `4242`. The chain of blocks will not be persistent, it will be stored in the memory of the server but the server will not be connected to any specific database software. When developing mining, three things must be done:

- Calculate proof of work
- Reward miners (one transaction)
- Creation of the new block and add it to the chain

Once the blockchain is created, you can interact with it through different HTTP requests on a text-based API:

- `[POST] /transactions/new` : Post a new transaction to add to the next block.
- `[GET] /mine` : Execute the proof of work and create a new block.
- `[GET] /chain` : Returns information about the full blockchain (blocks, transactions, etc).

Chapter V

Bonus Part

The evaluation of the bonuses will be done `IF AND ONLY IF` the mandatory part is `PERFECT`. Otherwise, the bonuses will be totally `IGNORED`.

You can enhance your project with the following features:

- Difficulty of the dynamic PoW algorithm, ascending depending on the number of mined blocks or the elapsed time.
- Implementation of communication with other network nodes through a decentralized network and a consensus algorithm to verify the correct chain.
- Proof of Stake in addition to Proof of Work, as an alternative, eco-friendly consensus algorithm.
- Everything that comes to your mind... you will be able to justify everything during the defense.

Chapter VI

Peer evaluation

This project will be corrected by other students. Turn in the files inside the Git repository and make sure everything works as expected.