udp UNIVERSIDAD DIEGO PORTALES

Facultad de Ingeniería y Ciencias

White Paper

Proyecto de Blockchain y algoritmos descentralizados

Integrantes	:	*Enrique Acosta, Benjamín Alonso, Christian Bastías y Diego Vergara*
Profesor Cátedra	:	*Marcello Tavano*
Fecha de Entrega	:	*29/11/2023*

Introducción

El presente documento describe el proyecto realizado para la cátedra de Blockchain y Algoritmos descentralizados, que tiene por objetivo el diseño e implementación de una red P2P que permite el funcionamiento de un servicio de transacciones sobre una Blockchain.

Motivación de Proyecto

La principal motivación de este proyecto recae en la oportunidad de aprendizaje de los principios fundamentales de la tecnología blockchain. En particular, comprender cómo estas innovaciones y nuevas tecnologías pueden transformar la forma en que interactuamos con servicios web supone un área de alto interés y potencial.

Además, la creación de una red P2P para facilitar transacciones en una blockchain no solo es un ejercicio académico, sino una aplicación práctica de los conceptos teóricos aprendidos en la cátedra. Este proyecto brinda la oportunidad de poner en práctica los conocimientos adquiridos y enfrentar desafíos en el diseño y la implementación de sistemas descentralizados.

Problemas a atacar

1. **Seguridad de las transacciones**: Desarrollar medidas para garantizar la seguridad de las transacciones en la red P2P.
2. **Persistencia de Datos:** Asegurar la persistencia y consistencia de los datos en la red, incluso en situaciones donde la red se detenga por completo.
3. **Prevención de Doble Gasto**: Implementar mecanismos que verifiquen y aseguren que una misma unidad de valor no se gaste más de una vez, garantizando la integridad y confiabilidad de las transacciones en la red blockchain.

Enfoque

El enfoque del proyecto, orientado en particular a un sistema monetario, se centra en el diseño y la implementación de una red que permita realizar transacciones de manera eficiente, segura y transparente. La visión principal para este proyecto es crear una infraestructura que pueda ser utilizada como un sistema financiero descentralizado, donde las transacciones se ejecuten de manera confiable sin depender de intermediarios centralizados.

Descripción de la arquitectura del software

Estructuración de capas lógicas

Se definen 4 capas principales, que comprenden las operaciones esenciales para la puesta en marcha de un red P2P para Blockchain:

1. **Capa de Infraestructura**: Engloba los componentes fundamentales para el funcionamiento de la red P2P, incluyendo la implementación de la comunicación libp2p, la gestión de nodos y la persistencia de datos.
2. **Capa de Blockchain**: Contiene la lógica central relacionada con la cadena de bloques, como la creación, escritura y lectura de bloques. Aquí se implementa la lógica de validación de transacciones y la solución de doble gasto.

3. **Capa de Transacciones**: Se enfoca en el procesamiento de transacciones, incluyendo la creación de nuevas transacciones y la sincronización de las transacciones entre nodos.
4. **Capa de API REST**: Proporciona una interfaz externa para interactuar con la red, exponiendo endpoints que permiten la creación de transacciones, consultas de bloques y transacciones a través de una API REST.

Detalles de implementación

Descripción de implementación del nodo y su protocolo de comunicación

La implementación de los nodos y su comunicación está basada en libp2p . En esta biblioteca, un nodo representa un participante en una red peer-to-peer (P2P). Asimismo, se comprende mejor como un componente central que encapsula la funcionalidad esencial para la comunicación descentralizada entre nodos en una red distribuida. Cada nodo libp2p tiene un identificador único (ID) y puede estar conectado a otros nodos en la red. En particular, la transmisión de información se realiza mediante el protocolo TCP. En este, esta capa de transporte proporciona la base para la comunicación fiable y orientada a la conexión entre nodos en una red P2P mediante el proceso de *handshake*.

Descripción de métodos de lectura/escritura implementados

Se implementan un método de lectura y otro de escritura, que permite a los nodos procesar la data en tránsito. Ambas funciones trabajan juntas para lograr una comunicación constante de escritura y lectura entre los extremos asociados al método bufio.ReadWriter(). La función *WriteData()* envía un saludo cada 5 segundos, mientras que *ReadData()* está continuamente leyendo las cadenas que llegan y que luego registra.

Solución de doble gasto

Se implementa un sistema pub-sub donde los nodos actúan como suscriptores y están suscritos a eventos relacionados con nuevas transacciones. El publicador es responsable de transmitir información sobre transacciones válidas a todos los suscriptores. Antes de incluir una transacción en un bloque, los nodos están atentos a los estados de la red ejecutados por el publicador a través de la red. Si una transacción ya ha sido confirmada, los nodos evitan incluirla nuevamente, lo que previene el doble gasto.

Descripción los modelos de datos utilizados

Se utiliza una serie de modelos de datos, implementados mediante *structs*, que se comprenden como un tipo de dato compuesto que agrupa campos de diferentes tipos bajo un mismo nombre. Se definen los siguientes structs:

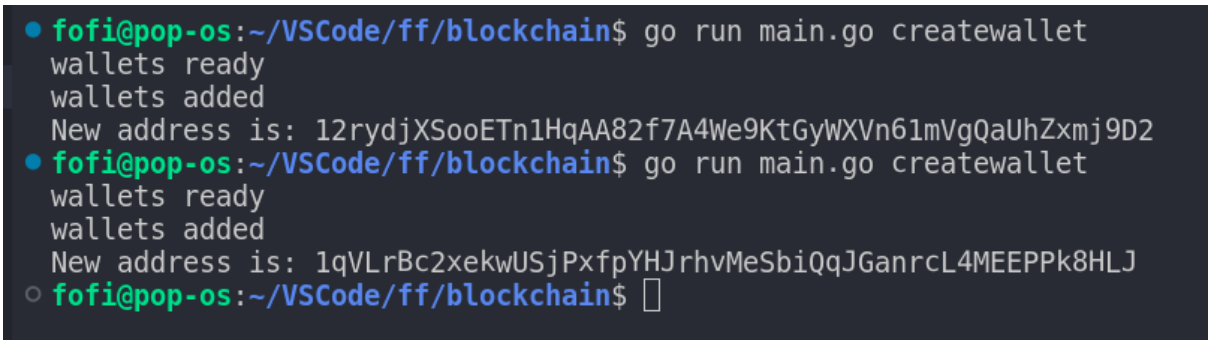
```
type ProofOfWork struct {
      Block  *Block
      Target *big.Int
}
type Blockchain struct {
      LastHash []byte
      Database database.DB
}
type Block struct {
      TimeStamp    string
      Hash         []byte
      Transactions []*Transaction
      PreviousHash []byte
      Nonce        int
}
type Transaction struct {
      ID      []byte
      Inputs  []TxInput
      Outputs []TxOutput
}
```

Estos structs forman la base de un sistema blockchain en el proyecto. **ProofOfWork** encapsula la prueba de trabajo en la minería de bloques, incluyendo el bloque actual y el objetivo para la dificultad de la prueba. **Blockchain** representa la cadena de bloques, con el último hash y una base de datos para almacenar los bloques.

El struct **Block** define la estructura de cada bloque en la cadena, con detalles como la marca de tiempo, hash, transacciones, hash anterior y un *nonce* para la prueba de trabajo. Por último, **Transaction** modela las transacciones en la blockchain, con identificadores, inputs y outputs, proporcionando la estructura fundamental para la transferencia de valor en el sistema descentralizado.

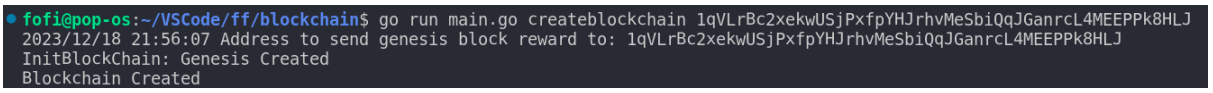
Pruebas de funcionamiento

Createwallet:

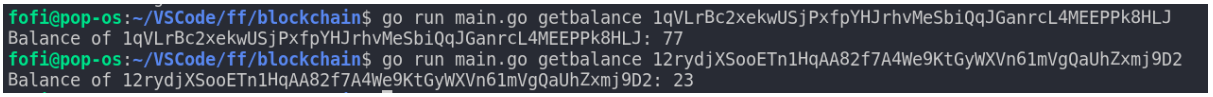


Listaddresses:

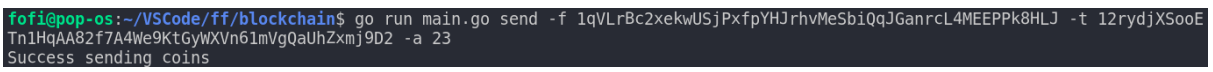
Createblockchain:



Getbalance:



Send:



Printchain:

```
fofi@pop-os:~/VSCode/ff/blockchain$ go run main.go printchain
All addresses in wallet:
12rydjXSooETn1HqAA82f7A4We9KtGyWXVn61mVgQaUhZxmj9D2
1qVLrBc2xekwUSjPxfpYHJrhvMeSbiQqJGanrcL4MEEPPk8HLJ
-------------------------
--------------
Block Hash: 00034c015e2e1e94af02f2fd6cc5d1500a62689f73500e7b5c5559a39e0cd96b
Previous Hash: 000a2d816e410fd8831bc0c8fb76fb049c17464370948915561c0d377983730c
Timestamp: "2023-12-18 21:56:53.890"

Transaction:
        TXID: f68d57879695fbd22ff78386fd41e7cc0fe0b6b420fc6206ca476455a2285a14

        Inputs:
        TXID: 0d876369a45f6fea39475ff76522bbaa0b8771df0ec469877ce9c92e740845c8
        Out: 0
        Signature: 7ee63457f8fd74e9aff7d523ddd68fdd9e6c867c5d06cf33e75882e5bd69c169ed05c5dfa4bf689bba179fd8dfb4c844bd5870409
e7adcd01072764468918507

        Outputs:
        Value: 23
        Address: 12rydjXSooETn1HqAA82f7A4We9KtGyWXVn61mVgQaUhZxmj9D2
        Value: 77
        Address: 1qVLrBc2xekwUSjPxfpYHJrhvMeSbiQqJGanrcL4MEEPPk8HLJ
--------------
Block Hash: 000a2d816e410fd8831bc0c8fb76fb049c17464370948915561c0d377983730c
Previous Hash:
Timestamp: "2023-12-18 21:56:07.074"

Transaction:
        TXID: 0d876369a45f6fea39475ff76522bbaa0b8771df0ec469877ce9c92e740845c8

        Inputs:
        TXID:
        Out: 100
        Signature:

        Outputs:
        Value: 100
        Address: 1qVLrBc2xekwUSjPxfpYHJrhvMeSbiQqJGanrcL4MEEPPk8HLJ
```