CSD Blockchain COIN

RUBEN VAZ, FCT, PT

1 INTRODUÇÃO

Nos dias de hoje os termos blockchain e criptomoedas são dos termos mais falados por todo o mundo. Atualmente estas moedas podem ser utilizadas como bem de troca. Mas como surgem estas moedas, como é garantida a segurança numa transferência, quanto tempo demora uma transferência até que o utilizador final possa utilizar esse valor.

Este projeto tem como objetivo a criação de uma blockchain para o desenvolvimento de uma criptomoeda, tendo como base outras já criadas e assim perceber o funcionamento das mesmas e que garantias podem ser obtidas através da implementação da blockchain.

Para que objetivo seja alcançado é necessário ter em conta quais são as necessidades da nossa moeda, de forma a garantir os principais pilares da segurança (Integridade, Confiabilidade e disponibilidade).

2 TRABALHOS RELACIONADOS

2.1 BFT-SMART

É uma biblioteca que implementa um protocolo semelhante ao PBFT, o objetivo desta biblioteca é a capacidade de manter a alta performance e o correto funcionamento do sistema no caso de existirem réplicas com falhas.

O protocolo é caraterizados por 3 fases, "Total Order Multicast", "State Transfer" e "Reconfiguration". Este modelo mantém o correto funcionamento no caso de ter 3f+1 nós, caso não seja verificada a condição, não é obtido um consenso e o cliente não aceita operar perante esta situação.

2.2 Bitcoin

A primeira criptomoeda a ter grande impacto no mundo e aquela que tem mais valor neste momento.

A bitcoin foi a principal referencia para o desenvolvimento da nossa blockchain, pois foi a partir da bitcoin que surgiu a estrutura para a nossa moeda e para a estrutura da nossa blockchain.

A blockchain da bitcoin guarda as transações confirmadas pelos nós, esta confirmação é feita através de mineradores (PoW) e que recompensa o minerador que consiga fechar um bloco válido primeiro. É importante também realçar o termo UTXO que é uma lista que contém todas as transações que ainda não foram gastas, ou seja, o saldo de cada utilizador.

3 ARQUITETURA

Pode-se dizer que o nosso programa está composto em três secções a API Rest, a interface que estabelece comunicação entre os clientes finais e a nossa aplicação, para a APIRest foi utilizada a framework spring. A comunicação Cliente RestApi é feita através de HTTPS utilizando como default a versão 1.2 pois a imagem docker não suporta TLS1.3, porém caso o sistema (versão do java) suporte TLS1.3 este pode ser alterado facilmente nas configurações do Spring.

Author's address: Ruben Vaz, ra.vaz@campus.fct.unl.pt, FCT, PT.

Em seguida a nossa RestApi comunica com um proxy (BFTSmart-Client), este proxy estabelece uma comunicação segura com todas as réplicas e verifica se as mesmas não estão a atuar de forma bizantina. Este canal usa também protocolo TLS1.2. Este cliente codifica os pedidos numa Message, (classe serializavel). Todas as mensagens têm uma chave, correspondente com o pedido, e um mapa com os tipos e o conteúdo de cada parâmetro.

Por fim temos as réplicas onde são analisados os pedidos provenientes do proxy e são feitas as verificações necessárias e o processamento dos dados. É também feita uma prova de hash para cada operação de forma que o proxy verifique que todas as réplicas escreveram a mesma operação.

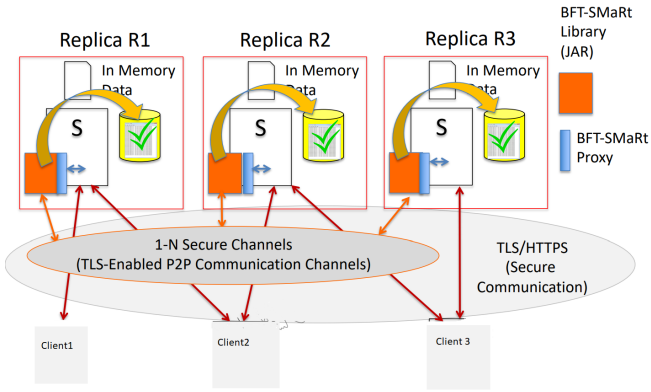

Fig. 1. Interfaces/Arquitetura da aplicação

Quanto á estrutura das classes o programa divide-se fundamentalmente nos utilizadores, que contém a informação sobre os mesmos, contém a chave publica do utilizador e adiciona os utilizadores ao sistema.

Temos a classe CoinBase que é a classe que tem os registos dos depósitos e das recompensas por mineração.

A classe TokenTransfer que contém a informação relativa a uma transação.

E por fim uma classe bloco que está dividida em duas classes uma para escrita na base de dados e envio do bloco na rede, que contém apenas um byte[] correspondente ao bloco(estrutura apresentada na próxima secção), e o javaBlock que contém o objeto bloco (estrutura do bloco, segundo programação orientada por objetos, simplifica as operações de validação).

4 MECANISMOS

Apesar da estrutura anteriormente apresentada existem certas características que não podem ser inseridas na blockchain pois, para cada uma das replicas este valor é diferente e no caso do cliente pedir um conjunto de dados (Transações e CoinBase) o consenso nunca seria obtido, então em relação a estrutura real destes objetos foram feitas algumas alterações. Estas alterações ajudam a evitar a serialização dos dados.

4.1 BlockChain

Cada bloco da cadeia contém um Id (4 bytes), um nonce (4 bytes, valor obtido através da mineração), um Hash dos dados, contém o hash final do bloco anterior, assim pode garantir que o bloco anterior não foi alterado, no caso deste hash ser diferente do anterior este bloco deixa de ser válido. Contém também um Hash do header deste bloco (32 bytes) e por fim contem os dados com as transações e com as CoinBase.

Todos os hashes são obtidos através de uma função SHA-256.

	Structure	Bytes
Header	Id	4
	Nonce	4
	DataHash	32
	PreviousHash	32
	HeaderHash	32
	Data	Variable

Fig. 2. Estrutura de um bloco na cadeia

4.2 Dados

Os dados estão organizados primeiro pelos CoinBase e depois pelas transações, antes de cada existe um int que revela quantos objetos de cada tipo existem. Uma vez que os tamanhos são variáveis para obter uma transação com um certo Id é necessário percorrer todo o bloco até chegar á transação pretendida.

Data	Bytes
Number of Coinbase	4
Coinbases	Variable
Number of Transaction	4
Transactions	Variable

Fig. 3. Estrutura dos dados num bloco

4.3 CoinBase

Um CoinBase é definido pelo seu Id(4 bytes), pelo numero de carateres do utilizador(4 bytes), o nome do utilizador(nº de caracteres obtidos anteriormente), e a quantia (4 bytes).

4.4 Transaction

Uma transação é também caraterizada pelo seu Id (4 bytes), a assinatura da operação pelo utilizador (32 bytes, inicialmente estava no final da estrutura, mas devido a alguns problemas foi colocada no início). Aqui da mesma maneira que no CoinBase são inseridos os números de caracteres e o nome do emissor (4 bytes + nº de caracteres), posteriormente o mesmo é feito para o destinatário.

Em seguida existe um byte que refere se a transação foi cifrada e por fim temos a quantia da transação (4bytes).

CoinBase	Bytes
Id	4
Number os Char's	4
Username	Variable
Amount	4

Fig. 4. Estrutura de um CoinBase nos dados

Transaction	Bytes
Id	4
Signature	32
Nº Char's From	4
From Username	Variable
Nº Char's To	4
To Username	Variable
IsEncrypted	1
Amount	4

Fig. 5. Estrutura de uma Transação nos dados

4.5 Mineração

Para que seja feita a mineração o utilizador deverá pedir um número de transferências maior que 0 e menor que 50. Na operação de pedir transferências o CoinBase prevalece sobre as transações, para que exista sempre a possibilidade do saldo ser o máximo, antes de qualquer transferência.

Em seguida o utilizador deverá obter informações sobre o último bloco a ser adicionado á cadeia, para que possa obter o Id do novo bloco e o Hash do anterior.

Para mineração os dados são dispostos segundo a estrutura do bloco, e vão sendo tentados diferentes nonces até que seja cumprida a condição dos três primeiros bytes todos a zero.

O bloco é enviado para o servidor e lá são verificados os requisitos e a validade do mesmo. No caso de ser um bloco válido o utilizador é recompensado com moedas. Estas moedas são atribuídas ao utilizador sob a forma de CoinBase.O numero de moedas da recompensa depende do número Transações, e ou, CoinBase que foram inseridas no bloco.

5 QUESTÕES DE IMPLEMENTAÇÃO

Existem alguns conflitos nomeadamente relativamente as transações pois, quando uma transação é criada, esta é criada com um timestamp.

Porém uma vez que os relógios não estão sincronizados, estas timestamps seriam diferentes de réplica para réplica. Ao devolver o objeto ao utilizador com a timestamp, o proxy não iria aceitar pois não conseguiria chegar a um consenso, os mesmo aconteceria quando o utilizador pedisse transações não confirmadas.

Outra implementação possível seria o utilizador passar o seu próprio timestamp, porém ao ser validado pelas réplicas estas poderiam não aceitar, devido à não sincronia dos relógios, no caso de criar um offset, este offset poderia aumenta a vulnerabilidade a ataques. Devido a esta situação os timestamps não foram implementados nos dados dos blocos.

Para o cálculo dos Hashes dos dados não foi utilizada uma MerkleRoot como deveria ser, de modo aumentar a segurança.

Para a realização de transações é necessário colocar o utilizador o que é mais user friedly do que ter de inserir a chave publica de outro utilizador, porém acrescenta mais complexidade á estrutura do bloco uma vez que as Strings podem ser variáveis em tamanho, ao contrário da chave publica utilizada (RSA).

6 ANÁLISE E OBSERVAÇÕES

Foi criado um teste onde era criado o sistema do zero, eram criados dois utilizadores e eram realizadas operações de transferência durante dois minutos, este teste foi realizado para um, dois, três e quatro clientes em simultâneo. Este teste foi realizado em máquina virtual o que pode também afetar os resultados.

Com o aumento do número de clientes em simultâneo, observa-se um aumento do número de operações, porém ao duplicar o numero de clientes não duplica o numero de operações. Observa-se também que após 4 clientes em simultâneo pode ocorrer uma diminuição do número de operações.

Também se pode observar que o tempo para executar uma operação também aumenta, o que nos leva a afirmar que quantos mais clientes estiverem conectados mais latência vão ter as operações.

Quantos aos valores mínimo e máximo não se pode tirar grandes conclusões pois podem ser falsos positivos, mas era esperado que estes aumentassem com o aumento do número de clientes em simultâneo.

Quanto á mineração de blocos pode se observar que os blocos que contém apenas CoinBase são mais rápidos. Quando comparados blocos que contém apenas o mesmo tipo (ou só transações ou só

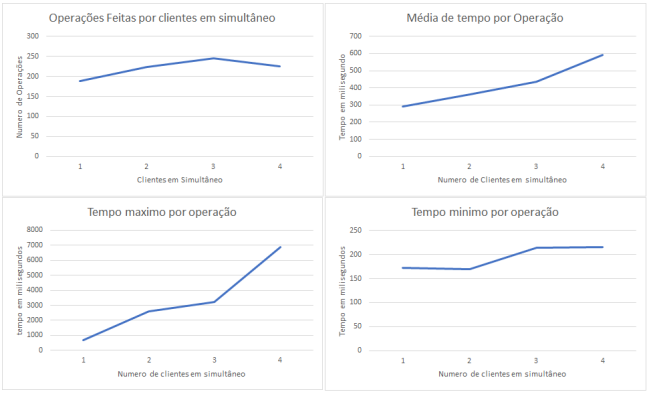

Fig. 6. Comparação de operações/tempos com o aumento do numero de clientes

CoinBase) verifica-se que quando o tamanho aumenta também o tempo de mineração aumenta.

Para este teste o hash do bloco anterior assim como o ID do bloco também podem influenciar estes resultados

Não esquecer que podem ser minerados blocos que contêm ambos os tipos de dados.

.

CoinBase		Transações	
25Bytes	750Bytes	180Bytes	7KBytes
28 s	120 s	137s	173

Fig. 7. Tempos de mineração dependendo do tipo e tamanho

7 CUMPRIMENTO DE REQUISITOS

Uma parte dos requisitos obrigatórios foram concluídos ficando por implementar as transferências com privacidade, onde o valor da transferência deveria ser cifrado, e a implementação de uma solução compatível com Smart Contracts. Quanto aos requisitos cumpridos, condições de verificação não estão a ser verificadas, por exemplo um cliente pode fazer transferências mesmo sem ter moedas. A implementação de uma UTXO ficou abaixo do pretendido, apenas existe um registo das transferências onde se podem encontrar aquelas que foram validadas e as que estão por validar. Para obter o saldo é necessário percorrer todas as transferências.

8 CONCLUSÃO

O projeto tem um grau de complexidade bastante interessante, pois existem muitos tradeoffs, e muitos detalhes. A realização deste projeto foi bastante gratificante pois conseguiu-se desenvolver um protótipo de uma cripto-moeda, algo que tem vindo a ganhar valor e é bastante falado nos dias de hoje. Apesar de não terem sido atingidos todos os objetivos, pode-se fazer um balanço positivo do trabalho realizado.

REFERENCES

[1] Nakamoto, S. (n.d.). Bitcoin: A Peer-to-Peer Electronic Cash System.

[2] Alysson Bessani, J. S., amp; E. P. Alchieri, E. (n.d.). State Machine Replication for the Masses with BFT-SMART.