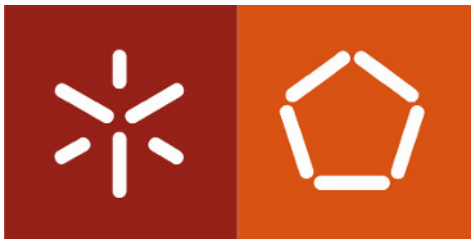

Mestrado em Engenharia Informática
Universidade do Minho

Engenharia de Segurança

Aula 05 TP - 15/03/2020

João Miranda - PG41845
Sandro Cruz - PG41906

16 de Março de 2020

Conteúdo

Capítulo 1

Blockchain

1.1 Pergunta 1.1

Quando o objecto "Block"for criado vai utilizar o *new Date()* para obter a data actual, essa data depois vai ser formatada de modo a só obter o ano, mês e dia.

```
class Blockchain{
    constructor(){
        this.chain = [this.createGenesisBlock()];
    }

    createGenesisBlock(){
        return new Block(0, new Date().toISOString().slice(0, 10) , "Bloco inicial da koreCoin", "0");
    }

    getlatestBlock(){
        return this.chain[this.chain.length - 1];
    }
```

Figura 1.1: Porção do código alterado

1.2 Pergunta 1.2

```
user@CSI:~/Desktop/EngSeg/TPraticas/Aula5/koreCoin$ node main.experiencia1.1.js Is Blockchain valid?
true
tampering with data...
Is Blockchain valid? false
user@CSI:~/Desktop/EngSeg/TPraticas/Aula5/koreCoin$
```

Figura 1.2: Execução do código "main.experiencia1.1.js"

Capítulo 2

Proof of Work Consensus Model

2.1 Pergunta 1.1

```
user@CSI:~/Desktop/EngSeg/TPraticas/Aula5/koreCoin$ time node main.experiencia2.1.js
Mining block 1...
Block mined: 007abd29e589b6a35f6107095cbc8a3628b7a8ff9ac5f7a203f79b262fff077e
Mining block 2...
Block mined: 009d3e5e004957e04604a4032cc58a97f33adb6711f93bc7514de8fe4a531a37
Mining block 3...
Block mined: 004bb82a658a99fd8c9f0d40ec9e69d3a441dacb6383242659b1ff8f7b42195b

real    0m0.096s
user    0m0.092s
sys     0m0.008s
```

Figura 2.1: Execução do script "main.experiencia2.1.js"com a dificuldade de 2

```
user@CSI:~/Desktop/EngSeg/TPraticas/Aula5/koreCoin$ time node main.experiencia2.1.js
Mining block 1...
Block mined: 0007c25169f5256ebb80fcbb423b582a8e699a759e41abb28906295973f9de4c
Mining block 2...
Block mined: 0004af57b70136e9f4b5309f92d518b3f297168d4cf47d4f9354808daadce457
Mining block 3...
Block mined: 00075796ab77bb5289bb9a952b91047482b3348d54d915ff5e5bce28256d01fe

real    0m0.475s
user    0m0.480s
sys     0m0.032s
```

Figura 2.2: Execução do script "main.experiencia2.1.js"com a dificuldade de 3

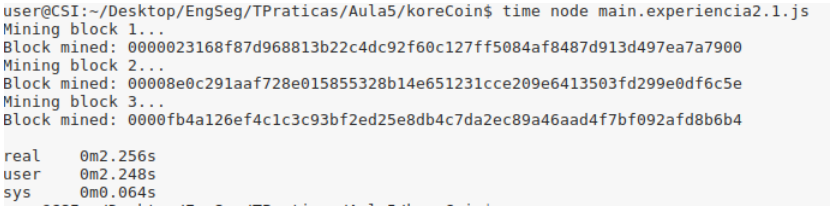

Figura 2.3: Execução do script "main.experiencia2.1.js"com a dificuldade de 4

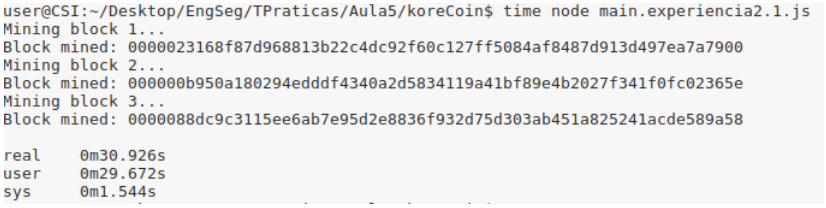

Figura 2.4: Execução do script "main.experiencia2.1.js"com a dificuldade de 5

Como dá para se reparar quanto maior for a dificuldade quanto mais tempo vai demora a minerar as *coins*. Dado que a dificuldade influencia o número de 0's com que a *hash* que é gerada necessita de ter no inicio de forma a um bloco ser minerado da *coin*, quanto maior for a dificuldade o mais difícil será de gerar uma *hash* que comece com esse número de 0's. A velocidade de geração das *hashs* é também influenciada pelo poder computacional do computador, pois o computador tem que tentar fazer os diversos cálculos de forma a chegar a *hash* com o número de 0's necessários.
Este pequeno exemplo demonstra como uma "proof of work"funciona, dado que demonstra como é possível implementar uma dificuldade que force a que seja necessário utilizar muito poder computacional para conseguir criar uma *coin*

2.2 Experiência 2.2

2.2.1 Na experiência anterior, qual é o algoritmo de 'proof of work' ?

O algoritmo implementado utiliza o cumprimento da *blockchain* como o numero inicial a ser incrementado e vai incrementado +1 de forma a quando chegar a um número divisível por 9 o utilizador minera uma nova *coin* e o processo de mineração termina, dado que o utilizador conseguiu obter a coin.

2.2.2 Parece-lhe um algoritmo adequado para minerar? Porquê?

Pelos teste que efectuei é muito fácil minerar a *coin* o que desvaloriza o valor da moeda, dado que é facilmente obtido um número múltiplo de 9, pois depois de criar uma nova *coin* ele só tem de fazer outras 9 incrementações de forma a criar um outra nova *coin*.
Também o facto de que os cálculos são efectuados todos pelo servidor poderá não ser o mais correcto dado que é possível fazer um ataque acidental de *denial of service* ao servidor se existirem muitos *miners* da *coin* pode dar um pouco de *overload* do servidor.