UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

THIAGO FONSECA MARTINS

Prova de existência de arquivos digitais utilizando a tecnologia blockchain do protocolo Bitcoin

Monografia apresentada como requisito parcial para a obtenção do grau de Bacharel em Engenharia da Computação

Orientador: Prof. Dr. Raul Fernando Weber

Porto Alegre
2018

“Intellectual growth should commence at birth
and cease only at death.”
— ALBERT EINSTEIN

AGRADECIMENTOS

Agradeço aos meus pais, Roberto Martins e Valquiria Martins, ao meu irmão, Gabriel Martins, e a todos os demais familiares, por todo o apoio e incentivo que sempre tive, não só ao longo deste curso, mas durante toda a minha vida.

A minha namorada, Júlia Ribeiro, pelo amor, pelo companheirismo, pela compreensão, e pelo suporte nos momentos mais difíceis.

Aos amigos e colegas de curso, Vicente Guedes, Iury Barros, Luiz Gustavo Gomes, Kleber Porto, e Pablo Bodmann, por estarem presentes tanto nos momentos mais difíceis desta jornada, quanto nos momentos de descontração, e por terem contribuído muito com a minha formação acadêmica.

Aos amigos mais antigos, Felipe Fontoura, Felipe Frosi, Daniel Balz, Maurício Badke, Felipe Kern, e Yuri Lopes, por todos os momentos de diversão, por todas as conversas construtivas, e por terem contribuído com a formação do meu caráter.

Ao meu professor orientador, Raul Fernando Weber, e a todos os professores e funcionários desta universidade, pela competência, pela dedicação, e pelos conhecimentos que continuamente transmitem a todos os alunos.

RESUMO

Blockchain é a tecnologia desenvolvida no protocolo Bitcoin para manter o registro de transações distribuído entre diversos usuários em uma rede *peer-to-peer*, dispensando a necessidade de centralização ou hierarquia de controle. Os mecanismos utilizados pela *blockchain* exploram o uso da criptografia para garantir a imutabilidade de seus registros. Como a adulteração dos dados armazenados é computacionalmente inviável, a tecnologia tornou-se alvo de estudos que destinam-se a expandir seu uso para outras áreas, que não se restringem ao setor financeiro. Neste trabalho foi desenvolvida uma aplicação que explora o uso da *blockchain* do protocolo Bitcoin para registrar arquivos digitais, sendo capaz de fornecer uma prova inegável de sua existência na data de registro. A aplicação funciona como um cartório digital descentralizado, podendo registrar arquivos em qualquer formato. Visando a compreensão das técnicas utilizadas para garantir a confiabilidade das informações armazenadas, foi realizado um estudo detalhado acerca do protocolo Bitcoin e da tecnologia *blockchain*, assim como da viabilidade de utilizá-la para tal propósito. As informações a respeito dos registros são inseridas na *blockchain* através de transações da moeda, requerendo o pagamento de uma taxa de transação. Ao longo do desenvolvimento deste trabalho, houve um aumento expressivo na taxa média necessária para que uma transação seja processada, inviabilizando movimentações de pequenos valores, e comprometendo a utilização do protocolo Bitcoin para o registro de arquivos. Para contornar o problema, este trabalho propõe o uso de *blockchains* alternativas, como a da plataforma Ethereum, que atualmente permite transações com taxas mais atrativas.

Palavras-chave: Blockchain. Bitcoin. Prova de existência. Registro distribuído. Aplicação descentralizada.

Proof of existence of digital files using the blockchain technology of the Bitcoin protocol

ABSTRACT

Blockchain is the technology developed on the Bitcoin protocol in order to maintain the transactions ledger distributed among users in a peer-to-peer network, dispensing the need for centralization or hierarchy control. The mechanisms used by the blockchain explore the use of cryptography to ensure immutability of its ledgers. Since the adulteration of the stored data is computationally impracticable, the technology has become the target of studies that are intended to expand its use to other areas, which are not restricted to the financial sector. In this work, an application to explore the blockchain of the Bitcoin protocol has been developed to register digital files, being capable to provide an undeniable proof of its existence on its date of registry. The application works as a decentralized digital notary, being able to register files in any format. Aiming the comprehension of the techniques used to ensure the reliability of the stored information, a detailed study has been carried out on the Bitcoin protocol and on the blockchain technology, as well as on the viability to use it for such purpose. The information regarding registries is inserted into the blockchain through coin transactions, requiring the payment of a transaction fee. Throughout the development of this work, there has been an expressive increase on the average fee needed for a transaction to be processed, making small value transactions impracticable and compromising the use of the Bitcoin protocol for registering files. In order to avoid this issue, this work proposes the use of alternative blockchains, such as the Ethereum platform one, which currently allows transactions with more attractive fees.

Keywords: Blockchain. Bitcoin. Proof of existence. Distributed ledger. Decentralized application.

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
BTC	Bitcoin
LIFO	Last In, First Out
P2PKH	Pay-to-Public-Key-Hash
RIPEMD	RACE Integrity Primitives Evaluation Message Digest
RSA	Rivest–Shamir–Adleman
SHA	Secure Hash Algorithm
UTXO	Unspent Transaction Outputs

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Neste capítulo será apresentada uma visão geral acerca da *blockchain*, mostrando algumas propostas alternativas de utilização da tecnologia. Após, serão apontados os objetivos deste trabalho, com exemplos de uso da aplicação a ser desenvolvida. O capítulo encerra apresentando os trabalhos relacionados, e descrevendo a forma como o texto foi estruturado.

1.1 Contextualização

A tecnologia *blockchain* funciona como um registro público e distribuído, construído e mantido por um número expressivo de participantes que recebem recompensas por sua contribuição no sistema. A ideia resultou na criação do Bitcoin, uma moeda digital descentralizada que propõe uma alternativa ao sistema de pagamentos atual por eliminar a dependência de terceiros em uma negociação. As transações são agrupadas em blocos, e a *blockchain* representa o encadeamento entre eles, explorando o uso da criptografia para garantir a segurança, a transparência e a imutabilidade dos registros.

Mais tarde, percebeu-se que a tecnologia poderia trazer vantagens em diversas outras aplicações que não se restringem ao setor financeiro. A empresa Everledger (2017), por exemplo, utiliza a *blockchain* para manter um registro rastreável de toda a cadeia de produção de bens, gerando um certificado de procedência ao final do processo. Outra aplicação é dada pela empresa Storj (2017), que faz o uso da *blockchain* para oferecer um serviço de armazenamento de arquivos em nuvem de forma descentralizada e segura.

1.2 Objetivos

O objetivo deste trabalho é explorar o uso da tecnologia empregada no protocolo Bitcoin, criando uma aplicação capaz de registrar arquivos digitais na *blockchain* e posteriormente fornecer uma prova de existência através do seu registro. É importante ressaltar que o registro de um arquivo não consiste no armazenamento do próprio arquivo na *blockchain*, mas sim no armazenamento de uma comprovação de que o arquivo existia na data do registro. Dado o caráter imutável do histórico armazenado pela *blockchain*, um usuário pode provar a data e a hora em que o arquivo foi registrado de maneira incontestável e verificável por qualquer um que possua uma cópia do arquivo original.

É possível registrar arquivos digitais de qualquer natureza. Um exemplo prático de uso da aplicação se dá no registro de um documento digital que representa um contrato, assinado digitalmente pelas partes contratantes. Com uma cópia do original, qualquer usuário pode comprovar que o registro de fato pertence ao contrato em questão, podendo comprovar também o momento em que foi registrado e incorporado à *blockchain*. Como outro exemplo, a aplicação poderia ser utilizada por artistas independentes que desejam registrar suas obras, como músicas, textos, pinturas, fotografias ou vídeos que estejam digitalizados em qualquer formato de arquivo. Em casos de plágio, pode ser possível provar a autoria exibindo uma comprovação da data de registro.

De maneira geral, a aplicação funciona como uma espécie de cartório digital descentralizado. Como os registros são distribuídos através da rede utilizando a cadeia de blocos do protocolo Bitcoin, as informações registradas possuem a mesma garantia de integridade e imutabilidade das transações da moeda. A dificuldade em fraudar um registro, como apagá-lo ou adulterá-lo, é equivalente à dificuldade em fraudar uma transação qualquer inserida na *blockchain*. Conforme será estudado ao longo deste trabalho, é computacionalmente impraticável realizar modificações na cadeia de blocos, fazendo com que ela se torne um local seguro para o armazenamento de pequenas informações.

1.3 Trabalhos relacionados

O protocolo Bitmessage[1] utiliza uma rede *peer-to-peer* inspirada no protocolo Bitcoin para enviar e receber mensagens privadas. Por se tratar de uma rede independente, adaptações foram realizadas a fim de permitir a troca de mensagens de qualquer tamanho, ultrapassando as limitações existentes no protocolo Bitcoin – que foi criado para efetuar transações monetárias. Entretanto, a utilização de uma rede independente traz à tona preocupações acerca da segurança do sistema.

Como solução, Sleiman, Lauf and Yampolskiy (2015) propuseram a codificação de mensagens em transações da própria rede Bitcoin, utilizando o campo que especifica a quantidade de moedas a serem transferidas. A implementação tem como foco o envio de mensagens secretas, baseando-se na natureza parcialmente anônima dos endereços Bitcoin. O método de codificação utilizado, porém, força o envio de transações com valores monetários excessivamente altos, dada a forte valorização que a moeda sofreu após a publicação da proposta.

A OriginalMy (2017), uma das primeiras empresas brasileiras a utilizar a tecnologia

[1] Disponível em: <https://bitmessage.org/bitmessage.pdf>. Acesso em: 27 de dezembro de 2017.

blockchain para fins não monetários, oferece entre seus serviços o registro de documentos digitais de maneira similar à estudada neste trabalho. Além da possibilidade de efetuar registros na *blockchain* do Bitcoin, é possível efetuá-los em outras três *blockchains* públicas: Ethereum[2], Ethereum Classic[3], e Decred[4], contribuindo com a flexibilização do custo de registro. De acordo com a OriginalMy, os contratos assinados através da plataforma possuem validade jurídica em território brasileiro, desde que as partes envolvidas demonstrem concordância com o modelo de certificação digital utilizado.

1.4 Estrutura do trabalho

No Capítulo 2 serão estudados os tópicos de criptografia fundamentais para a compreensão do protocolo Bitcoin, como funções *hash* criptográficas, criptografia de chave pública, criptografia de curvas elípticas, e assinaturas digitais. A forma como o protocolo faz o uso da criptografia será estudada no Capítulo 3, que tem como principal objetivo proporcionar a compreensão a respeito das transações da rede. No Capítulo 4, os conceitos estudados anteriormente serão utilizados para explicar o funcionamento da cadeia de blocos de transações, que serve como infraestrutura para a aplicação proposta. As técnicas utilizadas para inserir dados na *blockchain* serão estudadas no Capítulo 5, onde será apresentado também um estudo acerca das limitações da tecnologia. Detalhes a respeito da implementação da aplicação proposta, bem como os resultados obtidos, serão exibidos no Capítulo 6. Ao final deste trabalho, serão apresentadas as conclusões, assim como possíveis melhorias a serem realizadas em trabalhos futuros.

[2]Disponível em: <https://www.ethereum.org/>. Acesso em: 26 de outubro de 2017.
[3]Disponível em: <https://ethereumclassic.github.io/>. Acesso em: 26 de outubro de 2017.
[4]Disponível em: <https://www.decred.org/>. Acesso em: 26 de outubro de 2017.

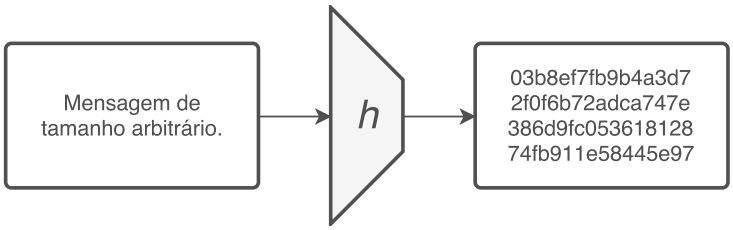
2 CRIPTOGRAFIA

Para que o funcionamento do protocolo Bitcoin seja devidamente compreendido, uma explanação dos conceitos criptográficos envolvidos se faz necessária. Neste capítulo serão estudados os fundamentos de funções *hash* criptográficas, criptografia assimétrica, criptografia de curvas elípticas, e assinaturas digitais.

2.1 Funções hash criptográficas

Uma função *hash* criptográfica consiste em um algoritmo matemático que produz uma saída de tamanho fixo dada uma entrada qualquer de tamanho arbitrário (THOMSEN; KNUDSEN, 2005). A entrada é usualmente chamada de mensagem, enquanto a saída é chamada de resumo. Para que uma função *hash* seja considerada ideal para o uso em criptografia, ela deve ser considerada unidirecional. Isso significa que dada uma mensagem qualquer, o cálculo do seu resumo deve ser computacionalmente simples, ao passo que reconstruir a mensagem a partir do seu resumo seja computacionalmente impraticável. A única maneira de obter a mensagem a partir do resumo deve ser por força bruta, testando todas as possíveis entradas buscando encontrar a mesma saída. Além disso, uma pequena modificação na mensagem deve produzir um resumo tão diferente do primeiro que os dois não aparentem correlação qualquer. A Figura 2.1 exemplifica o funcionamento de uma função *hash* criptográfica.

Figura 2.1: Exemplo de entrada e saída da função SHA-256



Fonte: Adaptado de (PRENEEL, 1994)

O nível de segurança de uma função *hash* é definido com base em sua capacidade de lidar com os diversos tipos de ataques criptoanalíticos, e para isso ela deve obedecer três propriedades importantes. A primeira propriedade está conectada ao conceito de unidirecionalidade, e é chamada resistência à pré-imagem. Refere-se à dificuldade de encontrar uma mensagem que produza um determinado resumo conhecido. Em outras palavras, dado um resumo h, deve ser difícil encontrar qualquer mensagem m tal que $hash(m) = h$. A segunda propriedade é chamada resistência à segunda pré-imagem, e diz que para uma dada mensagem, deve ser difícil

encontrar outra mensagem qualquer que produza o mesmo resumo da primeira. A terceira propriedade é chamada resistência à colisão, referente à dificuldade de encontrar duas mensagens quaisquer, diferentes entre si, que produzam o mesmo resumo (PRENEEL, 1994).

O uso de funções *hash* em segurança da informação é bastante amplo. Uma de suas principais aplicações se dá no uso de assinaturas digitais, mas também são utilizadas para verificar a integridade de arquivos ou mensagens, detectar dados duplicados, armazenar e verificar senhas de maneira mais segura, entre diversas outras aplicações. São também utilizadas no conceito de prova-de-trabalho, que será abordado em mais detalhes no Capítulo 4. O protocolo Bitcoin faz o uso dos algoritmos de *hash* SHA-256 e RIPEMD-160, que produzem resumos de 32 bytes e 20 bytes, respectivamente.

2.2 Criptografia assimétrica

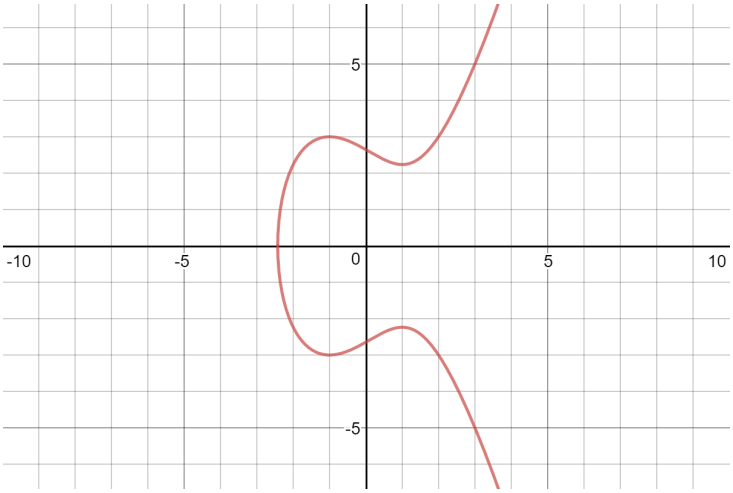
Os protocolos que utilizam criptografia assimétrica, também chamada de criptografia de chave pública, se valem de um par de chaves matematicamente conectadas: uma pública, que deve ser distribuída publicamente, e outra privada, que deve ser conhecida somente pelo proprietário (GIRAULT, 1991). O processo de geração de um par de chaves é computacionalmente simples, sendo a chave pública gerada a partir da chave privada. Apesar da ligação matemática, a obtenção da chave privada a partir da chave pública é um processo computacionalmente impraticável, pois é baseado em problemas matemáticos que atualmente não admitem solução eficiente, como o problema do logaritmo discreto (KO et al., 2000).

Considerando um par de chaves, uma mensagem criptografada com a chave pública só pode ser descriptografada com a chave privada, e vice-versa. Em uma comunicação entre Alice e Bob através da *internet*, por exemplo, Alice poderia enviar uma mensagem criptografada com a própria chave pública de Bob. Um atacante, mesmo tendo acesso à mensagem criptografada e à chave pública de Bob, não conseguiria descriptografar seu conteúdo – o que só é possível com a chave privada de Bob. Como a chave pública pode ser exibida sem que a segurança seja comprometida, algoritmos de chave pública não exigem um canal seguro para a troca inicial da chave secreta, como acontece na criptografia simétrica. Em realidade, a segurança de um sistema de criptografia assimétrica consiste em manter secreta a chave privada.

2.3 Criptografia de curvas elípticas

A teoria das curvas elípticas tem aplicações nas mais diversas áreas, como em geometria diferencial, teoria dos números e geometria algébrica sobre corpos finitos, tendo sua importância também em criptografia (OLIVEIRA, 2009). Uma curva elíptica é uma curva algébrica não-singular – o que significa que não possui cúspides ou auto-intersecções – definida pela equação $y^2 = x^3 + ax + b$, como demonstrada na Figura 2.2. As técnicas de utilização de curvas elípticas na criptografia foram propostas independentemente por Miller (1985) e Koblitz (1987), servindo como uma eficiente forma de implementação de um sistema de chave pública. De acordo com seus desenvolvedores, a criptografia de curvas elípticas pode ser mais rápida e utilizar chaves mais curtas para proporcionar o mesmo nível de segurança de métodos mais tradicionais, como o RSA[1].

Figura 2.2: Demonstração de uma curva elíptica

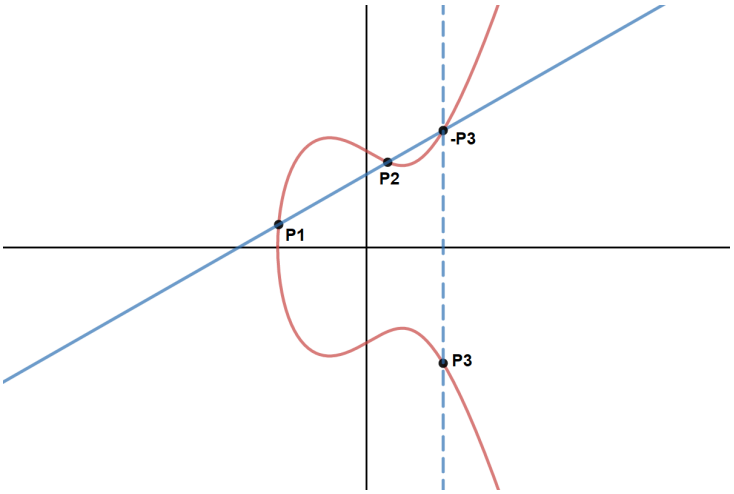


Fonte: Elaborado pelo autor

No campo da criptografia, as curvas elípticas utilizadas são definidas sobre corpos finitos. Um corpo é um conjunto com duas operações, normalmente soma e multiplicação, que satisfazem propriedades usuais das mesmas operações com números reais. A soma deve ser comutativa, associativa, possuir elemento neutro e elemento simétrico. Já a multiplicação deve ser comutativa, associativa, distributiva, possuir elemento neutro, e todo elemento não-nulo deve possuir um inverso. Conforme o exemplo da Figura 2.3, para obter o resultado da soma de dois pontos P_1 e P_2 pertencentes à curva, uma reta é traçada entre eles, fazendo intersecção em um ponto chamado $-P_3$. Refletindo o ponto sobre o eixo x, obtém-se $P_3 = P_1 + P_2$. No caso

[1] Método de criptografia de chave pública criado pelos professores Ronald Rivest, Adi Shamir e Leonard Adleman, do Instituto de Tecnologia de Massachusetts (RIVEST; SHAMIR; ADLEMAN, 1978).

Figura 2.3: Operação de soma em uma curva elíptica



Fonte: Elaborado pelo autor

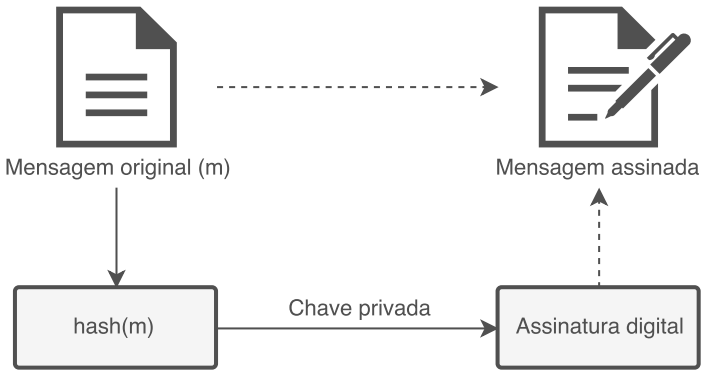
especial em que P_1 é exatamente igual a P_2, a reta deve ser traçada de forma a tangenciar o ponto na curva. Caso P_1 e P_2 tenham os mesmos valores em x, a linha traçada será vertical, e nesse caso, P_3 é considerado um "ponto no infinito". A multiplicação em curvas elípticas é definida como uma sequência de somas. Sendo P um ponto qualquer na curva e k um número inteiro, então $kP = P + P + ... + P$ (k vezes).

2.4 Assinaturas digitais

Assinatura digital é um método de autenticação de conteúdo digital desenvolvido para garantir determinadas propriedades. Uma assinatura digital deve prover autenticidade, permitindo ao receptor confirmar que a assinatura foi, de fato, produzida pelo emissor. Além disso, deve garantir a integridade do conteúdo assinado, fazendo com que a assinatura não corresponda mais ao documento caso ele venha a sofrer qualquer tipo de alteração. Por fim, deve ser irretratável, impedindo que o emissor negue a autenticidade do conteúdo (RIVEST; SHAMIR; ADLEMAN, 1978). A assinatura digital é normalmente tratada como substituta à assinatura física autenticada, servindo como prova de autenticidade de documentos digitais da mesma forma que a assinatura física serve como prova de autenticidade de documentos físicos.

Para assinar um documento, o autor deve encontrar o resumo da mensagem a ser assinada utilizando uma função *hash* criptográfica, e após, criptografá-lo utilizando sua chave privada, gerando assim uma assinatura digital, que é anexada ao documento original e disponibilizada para as partes interessadas. A Figura 2.4 exibe um esquemático demonstrando o processo de obtenção de uma assinatura digital.

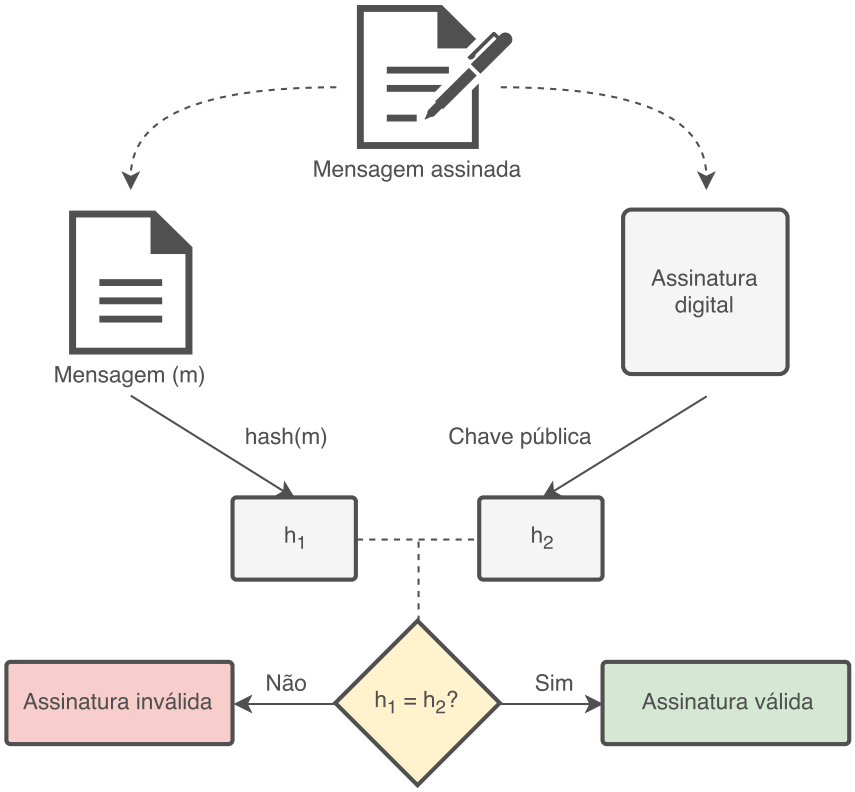
Figura 2.4: Processo de obtenção de uma assinatura digital



Fonte: Elaborado pelo autor

O receptor do documento assinado pode facilmente verificar sua autenticidade. Como exibido no esquemático da Figura 2.5, basta encontrar o resumo h_1 da mensagem recebida utilizando o mesmo algoritmo de *hash* utilizado na assinatura do documento, e em seguida, descriptografar a assinatura digital utilizando a chave pública do emissor, obtendo o resumo h_2 original da mensagem. Caso h_1 seja exatamente igual a h_2, a assinatura é considerada válida para o documento, e a autenticidade, a integridade e a irretratabilidade são garantidas.

Figura 2.5: Processo de verificação de uma assinatura digital



Fonte: Elaborado pelo autor

3 BITCOIN

Na primeira seção deste capítulo serão apresentadas as motivações que serviram como base para a criação das moedas digitais, bem como as primeiras propostas relevantes que deram inspiração para a implementação do Bitcoin. Nas seções seguintes, serão apresentados detalhes a respeito do protocolo, tendo como foco o funcionamento das transações.

3.1 História

Historicamente, transações monetárias *online* sempre requereram um intermediário de confiança. Transferências bancárias, por exemplo, exigem que a subtração do saldo em uma conta e a adição em outra seja controlada pelo banco. Pagamentos realizados com cartão de crédito exigem a aprovação da bandeira, e o controle de cobranças é feito pelo emissor do cartão. Existem ainda serviços que auxiliam a transferência *online* entre as partes, como o PayPal (2017). De toda forma, um intermediário sempre fez-se necessário para validar as transferências de recursos através de meios eletrônicos, pois diferentemente de moedas físicas, moedas digitais podem ser duplicadas, permitindo que um usuário faça cópias de suas moedas e envie para diversos outros simultaneamente – técnica conhecida como gasto duplo (KARAME; ANDROULAKI; CAPKUN, 2012).

Criptomoedas são moedas digitais que se valem da criptografia para assegurar a validade das transações. A principal proposta é eliminar a necessidade de um intermediário, e a dificuldade consiste em encontrar uma forma viável de fazer com que os participantes do sistema atuem em conjunto para garantir seu correto funcionamento, evitando fraudes e ataques por parte de usuários maliciosos. As criptomoedas buscam a descentralização do sistema de pagamentos, fazendo com que o registro das transações seja distribuído entre todos os usuários participantes, sendo todos eles responsáveis pela consistência dos dados, e eliminando a necessidade de confiança em um único intermediário. Quando uma tentativa de gasto duplo é detectada por um participante, ele deve notificar todos os demais; e uma transação é considerada válida quando os usuários atingirem um consenso a seu favor.

O conjunto de usuários participantes pode ser compreendido como uma rede, onde cada um representa um nó. Um sistema cujo consenso seja obtido através do voto da maioria dos nós sofre desvantagem semelhante ao problema dos generais bizantinos[1], sendo vulnerável ao

[1] Abstração utilizada para descrever situações onde o consenso é desejado em um sistema distribuído, mesmo na presença de componentes que tentam confundir os demais (LAMPORT; SHOSTAK; PEASE, 1982).

chamado *Sybil Attack*, que ocorre quando um usuário malicioso simula milhares de nós com o objetivo de tomar o controle das decisões (TSCHORSCH; SCHEUERMANN, 2016). No caso das moedas digitais, um atacante poderia simular tantos nós quantos fossem necessários para obter o controle das decisões, permitindo validar transações de gasto duplo de suas próprias moedas, por exemplo.

As primeiras pesquisas sobre o uso da criptografia para levar às transações eletrônicas a privacidade existente na troca do dinheiro em espécie apareceram ainda na década de oitenta. Em 1983, David Chaum publicou um artigo propondo um método chamado "assinaturas cegas", onde o banco seria responsável por assinar digitalmente as transações sem conhecer o seu conteúdo, provendo assim alto nível de privacidade aos seus usuários. No artigo, foi sugerida a criação de um sistema de pagamentos com as seguintes propriedades: incapacidade do banco determinar o beneficiário, a hora e a quantidade do pagamento feito por alguém; capacidade do pagador fornecer a prova do pagamento, ou determinar a identidade do beneficiário em circunstâncias excepcionais; e capacidade de interromper o uso de moedas que foram reportadas como roubadas (CHAUM, 1983). No início da década de noventa, foi especificada uma moeda eletrônica em princípio irrastreável, mas com a propriedade de permitir o rastreamento por parte do banco em casos de gasto duplo (CHAUM; FIAT; NAOR, 1990). No mesmo ano, David Chaum fundou a empresa DigiCash, onde trabalhou na implementação de suas ideias a respeito de dinheiro eletrônico.

Em 1998, preocupado com a interferência do governo no sistema monetário, Wei Dai propôs a criação de uma nova moeda digital, a qual chamou de *b-money*. O algoritmo proposto utiliza os conceitos de criptografia assimétrica, onde a chave pública é o endereço da carteira onde serão depositadas as moedas, e a chave privada é a senha pessoal para a utilização delas (DAI, 1998). As moedas digitais poderiam ser geradas por qualquer pessoa, que receberia um valor equivalente ao esforço computacional empregado em sua criação. Nick Szabo, em 2005, publicou um artigo expondo suas ideias acerca de uma moeda digital nomeada *bit gold*, propondo soluções para evitar o gasto duplo de moedas e garantir a sustentabilidade do sistema utilizando o conceito de prova-de-trabalho (SZABO, 2008). O autor introduziu também a ideia de "mineradores", isto é, nós da rede que receberiam recompensas em troca do poder computacional empregado na validação das transações.

Diversas outras propostas de moedas digitais surgiram ao longo da história. Sob o pseudônimo de Satoshi Nakamoto, foi publicado em 2008 através da *internet* um artigo especificando uma moeda digital chamada Bitcoin, que ganhou destaque por ter sido a primeira a solucionar de maneira eficiente os problemas anteriormente mencionados. A principal inova-

ção foi a solução proposta para resolver o problema do gasto duplo utilizando uma rede *peer-to-peer*, formando um registro distribuído de todas as transações construído através de enorme esforço computacional, e que não pode ser modificado sem que toda a prova-de-trabalho seja refeita (NAKAMOTO, 2008). O registro distribuído consiste em uma cadeia de blocos de transações chamada *blockchain*, e se vale da criptografia para garantir a segurança, a transparência e a imutabilidade do seu conteúdo. Na rede Bitcoin, o consenso não é atingido com base no voto da maioria dos nós, mas sim com base na solução que teve maior poder de processamento empregado. Como cada nó precisa resolver desafios computacionais para participar do processo, um atacante não poderia tomar o controle do sistema simplesmente simulando milhares deles. Seria preciso deter um poder computacional superior ao somatório do poder de processamento de todos os demais nós da rede.

A ideia que permite o funcionamento do protocolo foi tão inovadora que diversos estudos posteriores surgiram com o objetivo de expandir o uso da tecnologia para outras áreas, inclusive além do setor financeiro. Existe uma grande variedade de aplicações que podem se beneficiar das propriedades da *blockchain*, incluindo sistemas de votação, contratos inteligentes, ou até mesmo armazenamento distribuído de arquivos na nuvem. Os detalhes a respeito do funcionamento do protocolo Bitcoin serão estudados nas seções seguintes.

3.2 Rede

O protocolo Bitcoin está estruturado sobre uma arquitetura de rede do tipo *peer-to-peer*, também conhecida como P2P (NAKAMOTO, 2008). Nessa arquitetura, os participantes, chamados de nós, se comunicam diretamente entre si, atuando simultaneamente como cliente e servidor, e eliminando a dependência de servidores centralizados ou de um sistema de hierarquia entre os componentes (LUA et al., 2005). A escolha de implementar o protocolo sobre uma rede *peer-to-peer* garantiu ao Bitcoin um dos seus principais objetivos, que é a descentralização. Todos os nós são iguais entre si, no sentido de que um não possui prioridade sobre outro.

Alguns clientes da rede são conhecidos como nós completos (*full nodes*), e armazenam todo o histórico de transações que já ocorreram no sistema. O cliente referência da rede, chamado *Bitcoin Core*, é um software de código aberto cujo desenvolvimento foi iniciado por Satoshi Nakamoto, e que hoje conta com uma comunidade ativa de voluntários trabalhando em seu favor (WANG; PUSTOGAROV, 2017). O *Bitcoin Core* é considerado um nó completo, e possui as seguintes funções básicas: roteamento, armazenamento de dados, mineração, e gerenciamento de carteiras. Existem outras implementações de nós que utilizam um subconjunto das

quatro funções principais.

A função de roteamento se faz presente em todos os nós, ou seja, todos eles estabelecem conexões com outros nós e propagam transações e blocos através da rede. A função de armazenamento de dados se refere ao armazenamento completo da *blockchain*. Os nós que optam por armazenar uma cópia completa de todas as transações que já ocorreram podem verificar novas transações de forma autônoma. Outros nós armazenam apenas um subconjunto da *blockchain*, mas para verificar um pagamento devem solicitar aos nós completos um conjunto de dados adicionais. A verificação feita pelos nós que optam por não armazenar uma cópia completa da *blockchain* é estudada em mais detalhes no Capítulo 4.

Apesar de nem todos os nós armazenarem uma cópia local de todo o histórico de transações, a maioria mantém uma lista de transações que ainda não foram incluídas na *blockchain*, usualmente chamada de *mempool*. Assim que uma transação é recebida e verificada por um nó, ela é adicionada à *mempool* e transmitida aos nós adjacentes. Alguns nós armazenam também uma lista de transações órfãs – cujas entradas referenciam transações ainda desconhecidas. Outras implementações de nós mantêm ainda uma lista contendo todas as saídas não gastas de transações passadas, que estão disponíveis para transferências futuras.

3.3 Chaves, endereços e carteiras

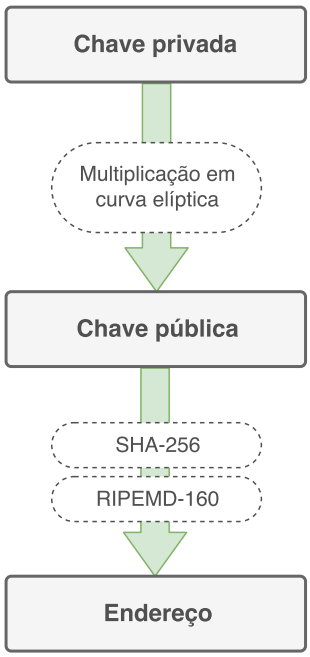
A posse de bitcoins é determinada através de chaves assimétricas e assinaturas digitais. Um endereço Bitcoin, utilizado como destinatário em transferências de posse, é criado através de uma chave pública, que por sua vez, é gerada a partir de uma chave privada. As chaves são normalmente criadas e gerenciadas por *softwares* específicos, genericamente chamados de carteiras (*wallets*), que independem do protocolo Bitcoin ou de acesso à *internet* para isso. Portanto, não é preciso que as chaves sejam armazenadas na rede, e a segurança sobre a posse dos bitcoins depende da forma com que o usuário mantém suas chaves privadas protegidas. Existem diversos tipos de carteiras, que são divididas entre dois grupos: carteiras quentes (*hot wallets*), e carteiras frias (*cold wallets*). As carteiras quentes são conectadas à internet, e normalmente contam com diversos recursos adicionais, como realizar transferências ou acompanhar o estado das transações. Já as carteiras frias armazenam as chaves de maneira mais segura, pois não são conectadas à rede, dificultando o acesso a elas. Entre os diversos tipos de carteiras, existem implementações em *software*, em *hardware*, ou até mesmo em papel (BONNEAU et al., 2015). É importante ressaltar que carteiras não armazenam moedas digitais, mas sim as chaves que permitem o seu uso. Portanto, mesmo que a carteira não esteja conectada à internet, um usuário

que tenha posse de suas chaves privadas poderá gastar seus bitcoins.

Para produzir um endereço Bitcoin, a primeira etapa a ser seguida deve ser a criação de uma chave privada k de 256 *bits*, que é simplesmente um número escolhido ao acaso entre 1 e 2^{256}. O método para a escolha do número é livre, mas não deve ser previsível ou repetível. Para a geração da chave pública K, o protocolo Bitcoin utiliza o padrão secp256k1[2], que estabelece uma curva elíptica definida sobre um campo finito de números primos. Além da curva, o padrão estabelece um ponto gerador G, que deve ser multiplicado pela chave privada, resultando em outro ponto na curva, que é então chamado de chave pública $K = k \cdot G$. Apesar da relação matemática entre as chaves k e K, o par somente pode ser calculado a partir da chave privada, obtendo-se então a chave pública. Atualmente, não existem métodos mais eficientes que força bruta para calcular a chave privada a partir da chave pública. A maioria dos endereços Bitcoin são gerados a partir de chaves públicas, mas existem ainda endereços gerados através de *scripts*, que serão estudados em seção posterior.

Os endereços produzidos através de chaves públicas são calculados utilizando-se duas funções *hash* criptográficas em sequência. Primeiro, o resumo da chave pública é obtido através do algoritmo SHA-256, gerando uma combinação de 32 bytes. Por fim, um novo resumo do resultado é computado utilizando o algoritmo RIPEMD-160, produzindo um endereço de 20 bytes (ATENIESE et al., 2014), como exemplificado na Figura 3.1.

Figura 3.1: Geração de um endereço Bitcoin



Fonte: Elaborado pelo autor

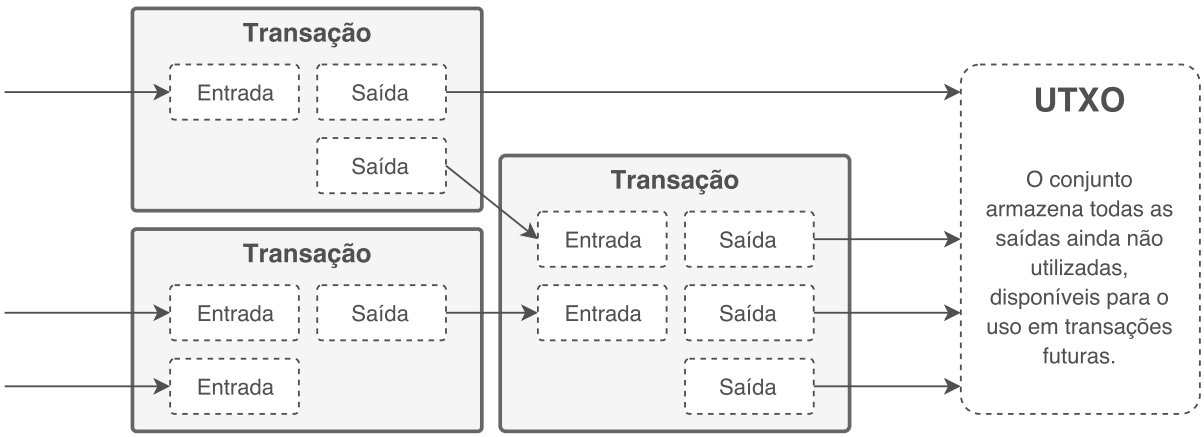
[2]Disponível em: <http://www.secg.org/sec2-v2.pdf>. Acesso em: 11 de novembro de 2017.

A fim de representar números grandes de maneira mais compacta, é comum a utilização de sistemas de numeração com bases maiores do que a decimal – como a chamada Base64, que utiliza 26 letras em caixa baixa, 26 letras em caixa alta, 10 numerais, e os caracteres "+" e "/". Para auxiliar a legibilidade, os endereços Bitcoin são representados na chamada Base58, que consiste em um subconjunto da Base64, porém omitindo caracteres que podem ser frequentemente confundidos ou parecer idênticos quando exibidos em determinadas fontes. Mais especificamente, a Base58 consiste no subconjunto resultante da eliminação dos caracteres "IOl0+/" (letra "i" maiúscula, letra "o" maiúscula, letra "l" minúscula, número "0", e caracteres "+" e "/"). Em realidade, a codificação utilizada para exibir endereços Bitcoin é chamada "Base58Check", por utilizar, além da Base58, um *checksum* ao final, criando uma proteção adicional contra erros de digitação (ANTONOPOULOS, 2014).

3.4 Transações

No protocolo Bitcoin, transações são estruturas de dados que codificam a transferência de valores entre participantes do sistema. Cada transação possui uma ou mais saídas, que especificam o destino do valor a ser transferido, e uma ou mais entradas, que descrevem sua origem. Sempre que uma nova transação é propagada, suas saídas são inseridas em um conjunto chamado UTXO (*unspent transaction outputs*), que armazena todas as saídas disponíveis da rede, que ainda não foram gastas por seus novos proprietários (DELGADO-SEGURA et al., 2017). Cada transação implica em uma modificação no conjunto UTXO, que cresce à medida que novas saídas são criadas, e diminui quando são consumidas. A Figura 3.2 exibe um exemplo de formação do conjunto UTXO.

Figura 3.2: Formação do conjunto UTXO



Fonte: Elaborado pelo autor

O conceito de saldo em uma carteira Bitcoin é criado na camada de aplicação, visando simplificar a utilização do sistema por parte de seus usuários. A informação de saldo não faz parte do protocolo e não é armazenada em registros públicos (BONNEAU et al., 2015). Em realidade, quando uma carteira detecta o "recebimento" de fundos, significa que detectou uma UTXO na rede que pode ser gasta utilizando uma das chaves controladas por ela. Sendo assim, o saldo de um endereço nada mais é do que o somatório dos valores do conjunto UTXO que podem ser gastos pelo proprietário de sua chave privada. Na prática, para informar a quantidade de bitcoins que um usuário tem à sua disposição, a carteira deve varrer todo o conjunto de transações não gastas da rede, buscando por UTXO compatíveis com suas chaves privadas.

O valor de saída pode ser um número de tamanho arbitrário. Porém, uma vez criada, a saída torna-se indivisível e só pode ser consumida por inteiro em uma nova transação (CURRAN; GEIST, 2016). Um usuário nem sempre conseguirá utilizar como entrada o valor exato que deseja transferir a outro proprietário. Caso tenha recebido diversas transferências de pequenas quantidades e deseje realizar uma transferência de maior quantidade, o usuário deverá agrupar múltiplas entradas para compor a transação. Frequentemente, o somatório total de bitcoins na entrada ultrapassa o valor que se deseja transferir. Nesses casos, o usuário deverá inserir seu próprio endereço na saída, informando a quantidade de bitcoins da entrada que permanecerá em sua posse, formando seu próprio troco (NAKAMOTO, 2008).

A diferença entre o somatório de moedas na entrada e o somatório de moedas na saída é chamada de taxa de transação, e será entregue como incentivo ao cliente responsável por sua validação. A taxa de transação é livre, porém, taxas maiores fazem com que a transação tenha maior prioridade no processo de validação. Se o número de transações não validadas for elevado, taxas pequenas podem fazer com que a inclusão da transação em um novo bloco seja postergada. Eventualmente, caso a transação não tenha sido processada, ela será eliminada, tornando a saída livre para que seja utilizada novamente.

A Tabela 3.1 exemplifica uma transação típica, onde Alice deseja transferir 4 BTC (abreviação da unidade monetária bitcoin) para Bob. Suponha que a carteira de Alice estava vazia antes de receber uma transferência de 3 BTC de Carlos, e outra de 5 BTC de Carol. Como a saída da transação que se deseja criar é superior a cada um dos valores que tem à sua disposição, Alice deve referenciar na entrada ambas as transações de Carlos e de Carol, fazendo com que a transação criada tenha em sua entrada um total de 8 BTC. Alice deve inserir o endereço da carteira de Bob na saída, especificando os 4 BTC que deseja transferir, e além disso, inserir o seu próprio endereço informando o troco desejado. No exemplo, Alice optou por pagar uma taxa de 1 BTC pela transação. Como a taxa é dada pela diferença entre o somatório de moe-

Tabela 3.1: Exemplo de entradas e saídas de uma transação

Entradas		**Saídas**	
Origem	**Quantidade**	**Destino**	**Quantidade**
Carlos	3 BTC	Alice	3 BTC
Carol	5 BTC	Bob	4 BTC
Total	**8 BTC**	**Total**	**7 BTC**

Fonte: Elaborado pelo autor

das na entrada e o somatório de moedas na saída, Alice deve especificar um troco de 3 BTC, totalizando 7 BTC na saída da transação.

De maneira geral, uma transação consome saídas não gastas de transações passadas, criando novas saídas que podem ser consumidas em transações futuras. A exceção fica por conta da transação chamada *coinbase*, que gera novos bitcoins a cada bloco de transações recém formado (TSCHORSCH; SCHEUERMANN, 2016). Os clientes responsáveis por construir novos blocos são chamados mineradores, e iniciam a formação do bloco inserindo a transação *coinbase*, que possui uma entrada especial que cria novas moedas, e tem como saída um endereço escolhido pelo próprio minerador. O processo de mineração funciona como uma competição, e apesar de cada minerador inserir seu endereço na saída da transação, ela somente se tornará válida para o primeiro que conseguir finalizar o bloco com sucesso. Mais detalhes a respeito do processo de mineração serão analisados no Capítulo 4.

3.5 Scripts

As entradas e saídas das transações são codificadas em uma linguagem de programação procedural, chamada simplesmente de *Script*. A linguagem é propositalmente simples, permitindo sua execução em uma grande variedade de *hardwares*, até mesmo em dispositivos embarcados. Apesar de conter diversos operadores, não permite laços ou controle de fluxos complexos, somente controle de fluxos condicionais. Isso garante que o tempo de execução seja previsível, impedindo que os *scripts* sejam usados para criar laços infinitos ou outras formas de sobrecarregar o sistema (ANTONOPOULOS, 2014). A linguagem utiliza uma estrutura de dados do tipo pilha, também conhecida como LIFO. Isso significa que existem duas operações básicas: adição e remoção de itens da pilha, sendo que a remoção somente pode retirar o último item inserido. A execução do *script* é realizada processando os itens da esquerda para a direita, e os operadores existentes atuam sobre um ou mais itens retirados da pilha, podendo

o resultado ser inserido nela novamente. Por não possuírem estados, sua execução produz o mesmo resultado em qualquer ambiente.

Os *scripts* são responsáveis por determinar as condições necessárias para que uma moeda possa ser gasta por um proprietário (DECKER; WATTENHOFER, 2014). A saída de uma transação comum, em realidade, é um *script* que fornece a chave pública ou endereço do destinatário, exigindo uma comprovação de posse da chave privada correspondente para liberar o uso das moedas. Sendo assim, o *script* de saída é comumente chamado de *script* de bloqueio, pois trava as moedas da saída, impondo uma condição para destravá-las. Por consequência, o *script* de entrada é chamado de *script* de desbloqueio, pois fornece os recursos exigidos pela saída de uma transação, permitindo o uso das moedas (GIECHASKIEL; CREMERS; RASMUSSEN, 2016). Para validar uma entrada, o *software* de validação deve executar o *script* de desbloqueio, presente na própria entrada, concatenado ao seu *script* de bloqueio, presente na saída da transação referenciada por ela. A entrada é considerada válida caso o *script* de desbloqueio satisfaça as exigências impostas pelo seu *script* de bloqueio.

O *script* de saída mais utilizado nas transações comuns é conhecido como *Pay-to-Public-Key-Hash*, ou P2PKH. Esse *script* fornece um endereço Bitcoin como destino, travando as moedas e dando permissão de uso somente para o cliente que possuir a chave privada relacionada ao endereço especificado (TSCHORSCH; SCHEUERMANN, 2016). Para que a saída possa ser gasta pelo destinatário, ele deve ser capaz de fornecer uma prova autêntica de que possui a chave privada do endereço. Uma vez que exibir a chave privada comprometeria a segurança e permitiria o uso dos bitcoins por qualquer um que a obtivesse, o proprietário deve fornecer como prova sua chave pública, juntamente de uma assinatura digital criada com sua chave privada. Em uma transação típica de Alice para Bob, a saída poderá conter um *script* P2PKH no seguinte formato:

```
OP_DUP OP_HASH160 <Endereço de Bob> OP_EQUALVERIFY OP_CHECKSIG
```

A função de cada um dos operadores utilizados no P2PKH é detalhada na Tabela 3.2. Sendo assim, caso Bob queira utilizar futuramente as moedas recebidas por Alice enviando uma certa quantidade para Carlos, ele deverá compor uma nova transação utilizando o seguinte *script* de entrada:

```
<Assinatura digital de Bob> <Chave pública de Bob>
```

Para verificar se a transação feita por Bob é validada, um cliente poderá concatenar o *script* de desbloqueio contido nessa transação ao *script* de bloqueio contido na transação

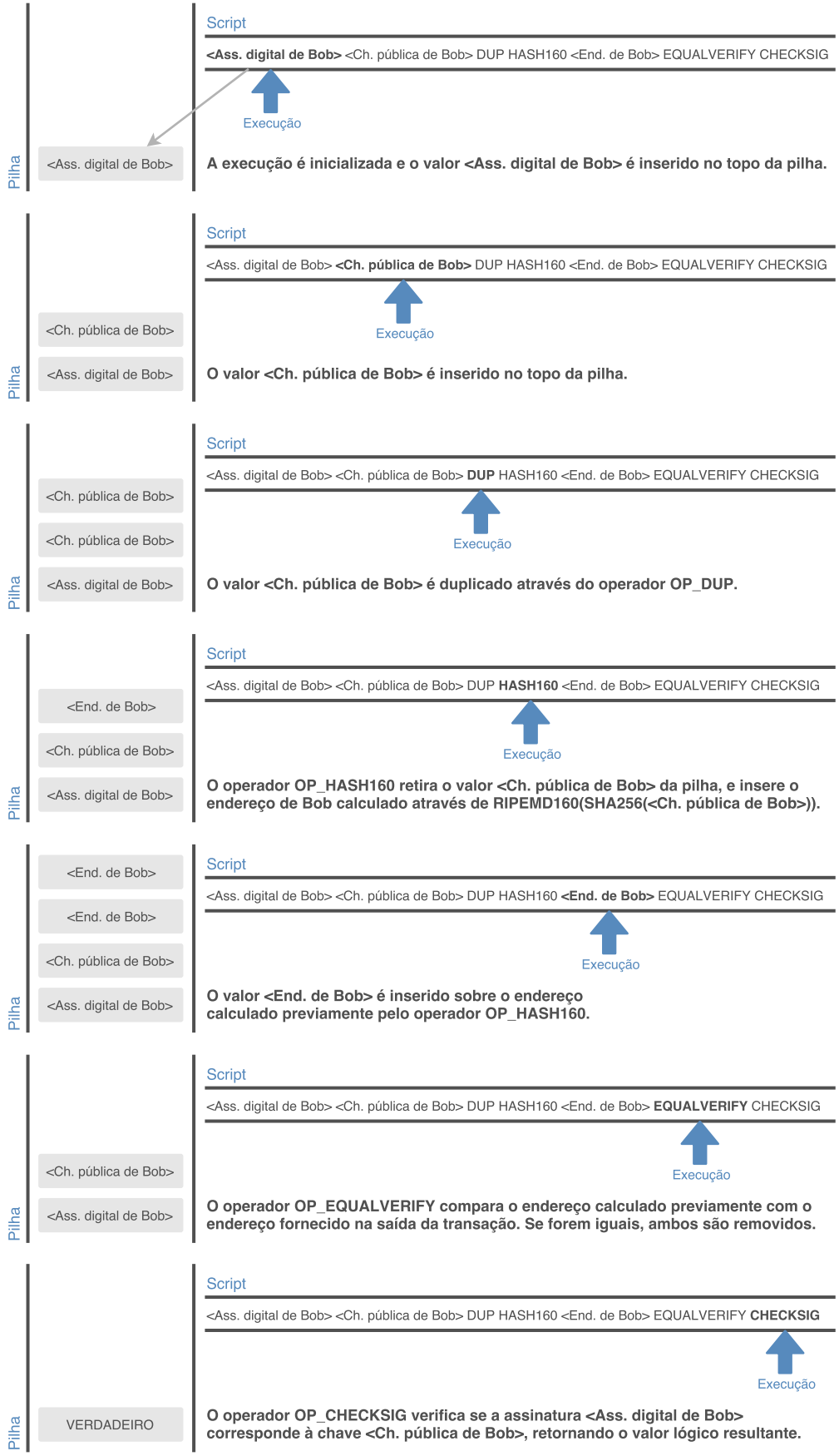
Tabela 3.2: Descrição dos operadores utilizados no P2PKH

Operador	**Descrição**
OP_CHECKSIG	Retira da pilha dois itens denominados *PubK* e *Sig*, e verifica se *Sig* é uma assinatura digital válida e correspondente à chave pública *PubK*. Insere na pilha o valor lógico resultante.
OP_DUP	Insere na pilha uma cópia do último valor inserido previamente.
OP_EQUALVERIFY	Retira e compara dois itens da pilha. Se forem iguais, a execução segue normalmente com os itens removidos da pilha. Se forem diferentes, a execução é interrompida.
OP_HASH160	Retira da pilha um item denominado PubK, e calcula RIPEMD160(SHA256(PubK)), obtendo o endereço Bitcoin correspondente à chave pública. Insere na pilha o resultado.

Fonte: Elaborado pelo autor

que Alice fez para Bob, formando o *script* de validação. Ao ser executado, o valor lógico verdadeiro será retornado somente se a assinatura digital e a chave pública fornecidas por Bob sejam compatíveis com o endereço de Bob, fornecido por Alice. Os passos de execução do *script* de validação do exemplo anterior são detalhados na Figura 3.3.

Figura 3.3: Validação do script P2PKH



Fonte: Adaptado de (ANTONOPOULOS, 2014)

4 BLOCKCHAIN

Neste capítulo será feita uma explanação dos mecanismos utilizados pela *blockchain* para garantir a segurança de seus registros. Será apresentada a estrutura dos blocos de transações, bem como a forma de conexão entre eles. Após, será estudado o processo de validação de um bloco, concluindo o capítulo com um detalhamento a respeito das técnicas adotadas para que o registro distribuído mantenha-se consistente entre os participantes.

4.1 Blocos

A *blockchain* é uma estrutura de dados distribuída que armazena blocos de transações em uma lista encadeada, onde cada um referencia seu bloco imediatamente anterior. A cadeia de blocos foi projetada para que seja computacionalmente impraticável realizar modificações em informações que já foram inseridas no sistema. Um bloco pode ser entendido como um conjunto de transações, juntamente com um conjunto de informações que compõem seu cabeçalho. O identificador de um bloco é chamado de *block hash*, e é definido pela dupla aplicação do algoritmo SHA-256 em seu cabeçalho (ANTONOPOULOS, 2014). Como o identificador pode ser facilmente calculado por qualquer nó da rede, ele não é armazenado no próprio bloco. Em vez disso, cada bloco armazena o identificador de seu bloco anterior, conhecido como bloco pai, formando uma cadeia de blocos que invariavelmente levará ao bloco gênesis – bloco inicial implementado diretamente no *software* de cada cliente (KROLL; DAVEY; FELTEN, 2013). Embora cada bloco possua apenas um único pai, ele pode temporariamente possuir múltiplos filhos. O evento é conhecido como *fork*, e será abordado na seção 4.5.

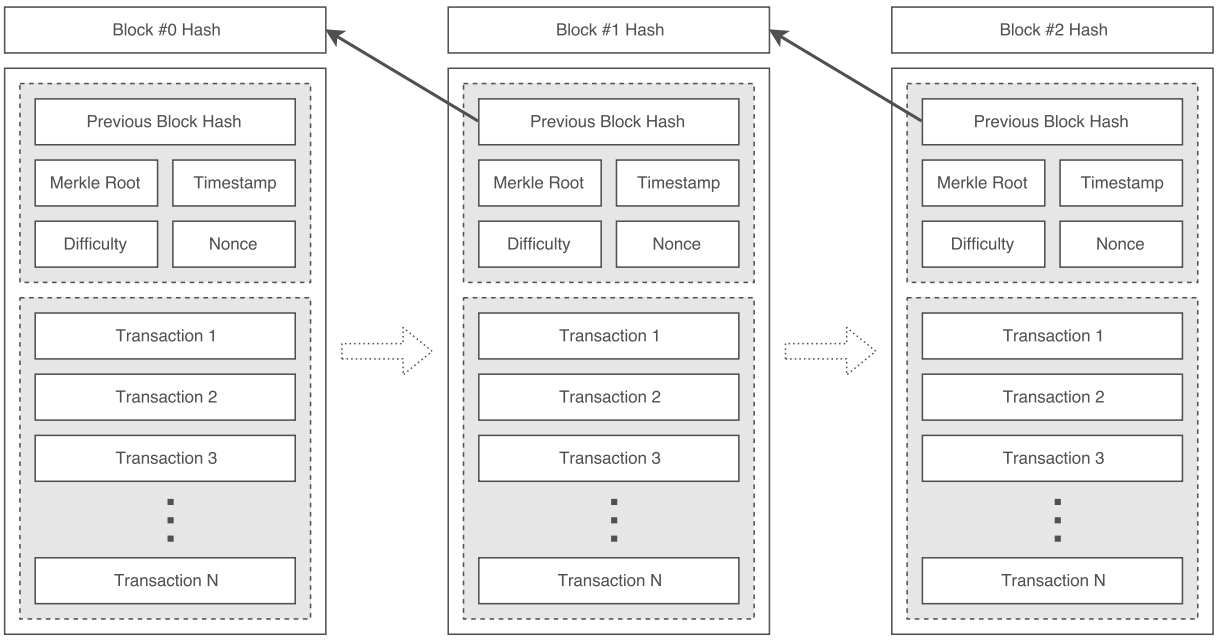
A estrutura do cabeçalho, representada na Tabela 4.1, é dividida em três conjuntos de dados com propósitos diferentes. O campo *previous block hash* garante a conexão entre todos os blocos da *blockchain*. O campo *Merkle Root* é usado para resumir de maneira eficiente o conjunto de transações do bloco, e será estudado na seção 4.2. Por fim, o conjunto dos campos *timestamp*, *difficulty target*, e *nonce* são referentes ao processo de mineração, e serão detalhados na seção 4.4. A Figura 4.1 mostra um exemplo de como se dá o encadeamento entre os blocos. Além da identificação padrão de um bloco dada pelo *hash* duplo de seu cabeçalho, é comum que se refira a um bloco pela sua posição na *blockchain*, onde o bloco gênesis é definido como bloco zero. Porém, ao contrário do *block hash*, a posição do bloco não pode ser considerada como um identificador único. Apesar da posição de um bloco ser invariável na *blockchain*, uma posição pode ser disputada simultaneamente por dois ou mais blocos.

Tabela 4.1: Estrutura do cabeçalho de um bloco

Campo	**Descrição**
Version	Número de versão utilizado para rastrear atualizações do *software*/protocolo.
Previous Block Hash	Referência ao identificador do bloco anterior na *blockchain*.
Merkle Root	Resumo do conjunto de transações pertencentes ao bloco.
Timestamp	Hora aproximada da criação do bloco (em formato Unix Epoch).
Difficulty Target	Dificuldade alvo do algoritmo de prova-de-trabalho utilizada no bloco.
Nonce	Contador utilizado pelo algoritmo de prova-de-trabalho.

Fonte: Adaptado de (ANTONOPOULOS, 2014)

Figura 4.1: Encadeamento de blocos



Fonte: Elaborado pelo autor

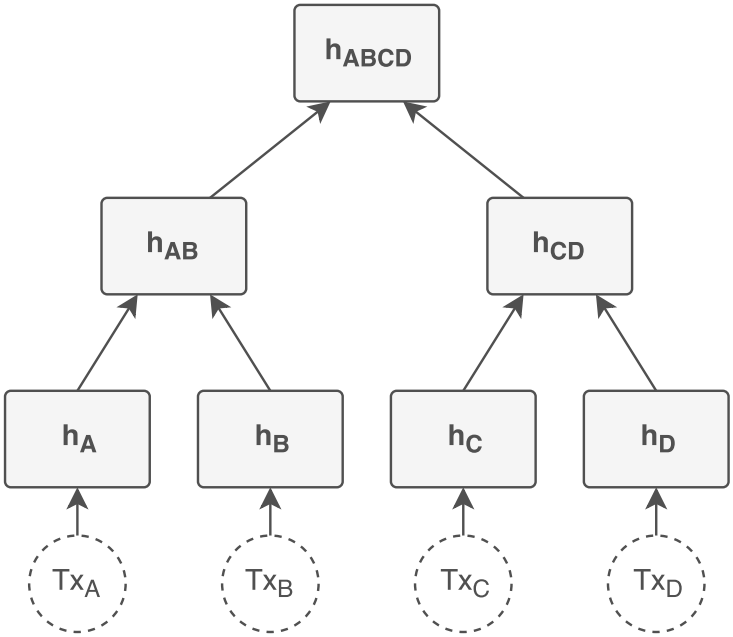
O bloco gênesis é conhecido por todos os nós da rede, uma vez que está codificado diretamente no *software* de cada cliente, e é o ponto de partida da *blockchain*. Contém uma única transação do tipo *coinbase*, que gera novas moedas e não faz referência a transações anteriores. Nessa transação foram geradas as primeiras 50 unidades da moeda. Em seus parâmetros foi codificada a mensagem "*The Times 03/Jan/2009 Chancellor on brink of second bailout for banks*", referente à matéria publicada pelo jornal The Times no mesmo dia em que o bloco gênesis foi gerado, marcando a data inicial da *blockchain*. Partindo do bloco gênesis, os nós completos armazenam uma cópia local de toda a cadeia de blocos e são responsáveis por mantê-la constantemente atualizada. Sempre que um novo bloco é recebido ele deve ser verificado, e, se for válido, adicionado ao fim da *blockchain*.

4.2 Árvores de Merkle

Também conhecidas como Árvores de Dispersão, as Árvores de Merkle são árvores binárias utilizadas para resumir e verificar de maneira eficiente a integridade de grandes conjuntos de dados. No protocolo Bitcoin, são utilizadas para gerar um *hash* único, chamado Raiz de Merkle, que serve como impressão digital do conjunto completo de todas as transações pertencentes a um bloco (TSCHORSCH; SCHEUERMANN, 2016). Através da Raiz de Merkle e de determinada seleção de nodos da árvore, um cliente é capaz de provar que uma transação específica pertence a um bloco, mesmo sem possuir seu conjunto completo de transações. O método é chamado de Verificação de Pagamento Simplificada, e será abordado na Seção 4.3.

O processo de obtenção da Raiz de Merkle a partir de um conjunto de transações é exemplificado na Figura 4.2, onde Tx_A, Tx_B, Tx_C e Tx_D representam as transações presentes no bloco. Para construir a Árvore de Merkle, o primeiro passo consiste em calcular o resumo de cada transação utilizando o algoritmo SHA-256 duplamente, formando os nodos que são chamados de folhas. Como exemplo, h_A é formado através da função SHA256(SHA256(Tx_A)). Cada par de nodos da árvore dará origem a um nodo pai, e para tanto, são concatenados e processados pelo mesmo algoritmo de *hash*, produzindo um novo nível da árvore. Assim, seguindo o exemplo, h_{AB} é formado por SHA256(SHA256($h_A + h_B$)). O processo se repete até que se encontre a Raiz de Merkle, que é armazenada no cabeçalho do bloco (GOBEL; KRZESINSKI, 2017). A árvore deve ser balanceada, e portanto é necessário um número par de transações. Nos casos em que o bloco contém um número ímpar de transações, o resumo da última transação deve ser duplicado para que a raiz seja calculada.

Figura 4.2: Representação da Árvore de Merkle de um bloco de transações



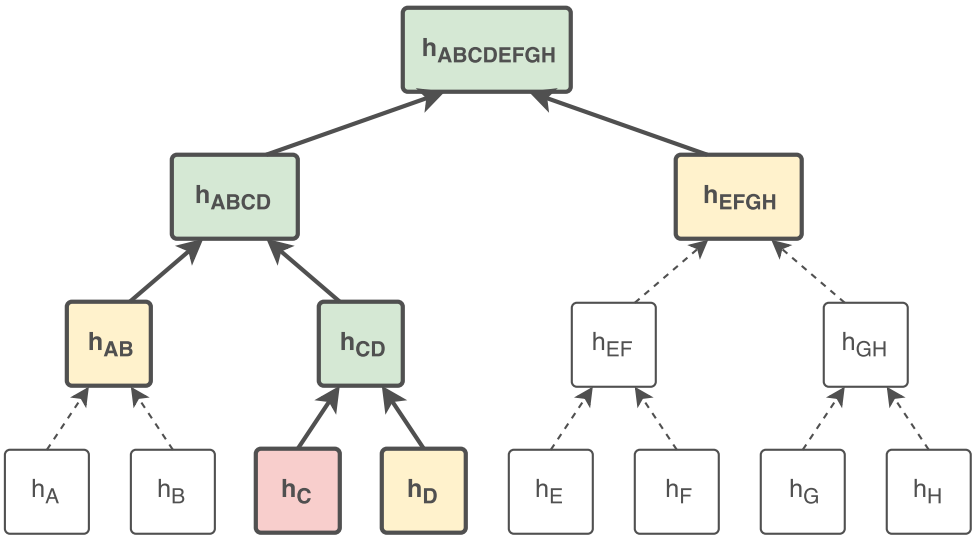
Fonte: Elaborado pelo autor

4.3 Verificação de Pagamento Simplificada

Alguns clientes optam por não armazenar o registro completo de toda a *blockchain*, como clientes executados em dispositivos móveis ou em máquinas com pouco espaço de armazenamento disponível. Nesses casos, para validar uma nova transação recebida, é feito o uso de um sistema de verificação de pagamento simplificada – *Simplified Payment Verification*, ou SPV – onde o cliente utiliza apenas os cabeçalhos dos blocos, e um número reduzido de nodos da Árvore de Merkle, que são solicitados sob demanda aos clientes que possuem uma cópia local de toda a cadeia de blocos (NAKAMOTO, 2008). Os clientes que utilizam esse mecanismo são chamados clientes SPV, e necessitam de um conjunto de dados de tamanho muito inferior ao da *blockchain* completa. Ao todo, considerando os cabeçalhos e o caminho de Merkle, cada bloco de um cliente SPV ocupa aproximadamente um milésimo do tamanho de um bloco completo (ANTONOPOULOS, 2014).

Conforme o exemplo exibido na Figura 4.3, suponha que um cliente SPV esteja interessado em verificar se a transação C pertence a um determinado bloco. O cliente possui apenas a Raiz de Merkle, e deve solicitar os nodos essenciais para formar o caminho que conecta a transação C a ela. Assim, o cliente recebe os nodos h_D, h_{AB} e h_{EFGH}, através dos quais é possível calcular todos os demais resumos necessários para alcançar a raiz. Caso a raiz calculada seja a mesma especificada no cabeçalho, o cliente possui uma prova válida de que a transação

Figura 4.3: Caminho de Merkle utilizado para provar a existência de uma transação



Fonte: Elaborado pelo autor

C pertence ao bloco. É importante observar a eficiência da verificação utilizando Árvores de Merkle. Para um bloco de N transações, é possível provar que uma transação específica está presente no bloco produzindo $2 \log_2 N$ valores de *hash*.

Nós completos verificam as transações varrendo todos os blocos anteriores da cadeia até que o primeiro bloco tenha sido atingido. Enquanto isso, clientes SPV verificam os blocos que foram adicionados depois da transação em questão, tendo uma confirmação de que ela foi aceita pela rede. Para que a verificação de pagamento simplificada seja confiável, é preciso que o cliente esteja conectado a nós honestos. Não é possível fazer um cliente SPV acreditar que uma transação está presente em um bloco ao qual não pertence, pois a existência da transação no bloco é verificada utilizando o Caminho de Merkle. Todavia, é possível ocultar uma transação existente. Em outras palavras, um cliente SPV pode provar que uma determinada transação existe, mas não pode provar a não existência de uma outra transação – que pode ser um gasto duplo da primeira, por exemplo.

4.4 Mineração

O propósito da mineração é frequentemente confundido com a criação de novas moedas. De fato, o processo de mineração gera novas moedas, porém esse artifício é usado como incentivo para que os participantes do sistema contribuam com o principal objetivo da mineração, que é validar as transações da rede sem a necessidade de uma autoridade central. Mesmo depois de uma nova transação ser propagada, o valor de saída somente pode ser gasto pelo

destinatário após a transação ter sido verificada e incluída em um bloco através do processo chamado mineração. Quando a construção do bloco é finalizada, ele é adicionado à cadeia de blocos, e todas as transações nele contidas recebem uma "confirmação". Sempre que um novo bloco é minerado, todas as transações já presentes no registro da *blockchain* recebem uma nova confirmação, de forma que o número de confirmações representa a profundidade da transação na cadeia de blocos (TAYLOR, 2013).

Os mineradores são recompensados pela sua contribuição de duas maneiras. A primeira delas é através da criação de novas moedas que são incluídas em cada bloco recém formado, as quais o minerador recebe o direito de posse. Além disso, o minerador também recebe o somatório das taxas incluídas em todas as transação do bloco. A criação de novas moedas se dá através da transação *coinbase*, sendo a primeira transação inserida em cada bloco. A *coinbase* não referencia transações anteriores em sua entrada, e a quantidade de moedas que podem ser geradas varia de acordo com a quantidade de blocos já produzidos. A partir da criação do bloco gênesis, cada novo bloco passou a injetar no sistema 50 unidades da moeda. De acordo com o protocolo, esse valor é reduzido para a metade a cada 210.000 blocos – evento chamado de "*halving*" (TSCHORSCH; SCHEUERMANN, 2016). A construção de um bloco é feita em aproximadamente 10 minutos, o que significa que um evento de *halving* deve ocorrer, em média, a cada 4 anos. A taxa de criação de moedas decresce exponencialmente, e a expectativa é de que no ano de 2140 todas elas já tenham sido geradas – momento em que existirão aproximadamente 21 milhões de bitcoins em circulação. Atualmente, as taxas de transação representam uma porcentagem pequena da recompensa total do minerador. Porém, conforme a criação de novos bitcoins por bloco for diminuindo, uma porcentagem cada vez maior da recompensa virá através das taxas. Quando o número de moedas em circulação tiver atingido seu limite, os mineradores serão incentivados a continuar cooperando apenas através das taxas de transação.

A mineração de um bloco é baseada no conceito de prova-de-trabalho, que foi desenvolvido na década de noventa com o intuito de combater o abuso de *spam*, embora nunca tenha sido amplamente utilizado para esse fim (BONNEAU et al., 2015). A ideia consiste em fazer com que um usuário seja capaz de provar que realizou alguma tarefa, recebendo assim o direito de executar determinada ação. A tarefa consiste na solução de um desafio criptográfico que deve ser computacionalmente custoso, mas que possa ser facilmente verificado por um avaliador (VALLOIS; GUENANE, 2017). Em outras palavras, o usuário deve ser capaz de provar através da solução do desafio que dedicou tempo suficiente na tarefa proposta.

Todos os nós da rede que contribuem com o processo de mineração estão em uma cons-

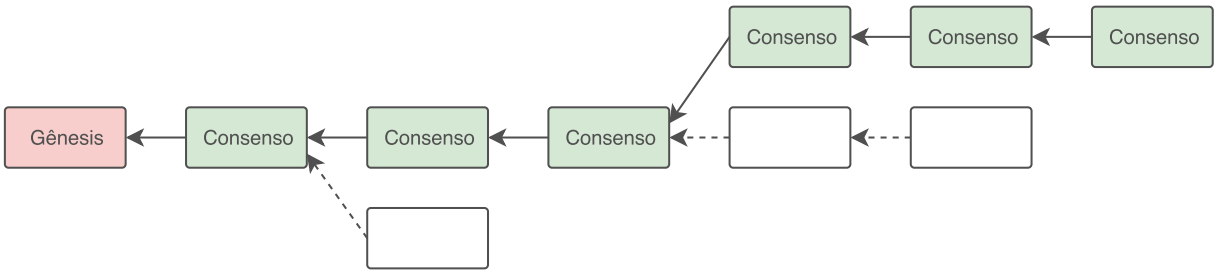
tante disputa global para solucionar o desafio criptográfico do novo bloco de transações que está sendo construído. O desafio consiste em repetidamente calcular o *hash* do cabeçalho variando o número contido no campo *nonce*, até que o resumo resultante satisfaça determinada condição. Mais especificamente, o *hash* do cabeçalho é calculado utilizando duas vezes o algoritmo SHA-256, e a condição necessária é que o resultado seja menor do que o valor alvo especificado no campo *difficulty target* (BONNEAU et al., 2015). De acordo com o conceito de prova-de-trabalho, o desafio é computacionalmente complexo e difícil de ser resolvido, mas uma vez encontrado o valor de *nonce* que satisfaz a condição imposta, qualquer avaliador pode facilmente verificar seu resultado. Assim que um minerador consegue solucionar o desafio, o bloco é distribuído através da rede, e sendo válido, é finalmente inserido na *blockchain*. As novas moedas geradas, bem como as taxas de transação são entregues somente ao minerador que conseguir concluir o desafio criptográfico do bloco antes dos demais. Quando um novo bloco é propagado e aceito pelos demais, todos os mineradores passam a trabalhar imediatamente na construção do bloco seguinte.

A dificuldade de criação de um bloco é proporcional ao valor alvo. A cada 2016 blocos – aproximadamente duas semanas – o valor alvo é ajustado para que a taxa de criação de novos blocos mantenha-se em aproximadamente 10 minutos. Se os últimos 2016 blocos foram gerados em um tempo menor do que duas semanas, significa que o poder de processamento geral dos nodos aumentou, e então a dificuldade deve ser também aumentada (TSCHORSCH; SCHEUERMANN, 2016).

4.5 Consenso

Dado o caráter descentralizado da *blockchain*, é comum que momentaneamente existam diferentes versões suas entre os nós da rede. A situação ocorre quando dois mineradores resolvem quase simultaneamente o algoritmo de prova-de-trabalho do bloco em que estão construindo, propagando sua solução através da rede (VALLOIS; GUENANE, 2017). Os dois blocos são válidos, apesar de diferentes, e passam a competir pela mesma posição da *blockchain*. Dependendo de diversos fatores da rede, cada nó receberá primeiro uma ou outra solução, incorporando o bloco recebido em sua cópia local da *blockchain*. Conforme exibido na Figura 4.4, quando um nó recebe um segundo candidato válido para uma posição que já foi ocupada em sua versão local, ele cria uma ramificação secundária, conectando o bloco a ela – evento conhecido como *fork*. Como resultado, alguns nós da rede terão um dos candidatos como extensão da cadeia principal, enquanto outros nós terão o outro, fazendo com que a rede tenha

Figura 4.4: Exemplo de forks temporários na blockchain



Fonte: Adaptado de (KROLL; DAVEY; FELTEN, 2013)

duas perspectivas a respeito da *blockchain*.

Para resolver a inconsistência, cada nó é programado para dar continuidade à cadeia que possui a maior prova-de-trabalho acumulada, também conhecida como cadeia mais longa (NAKAMOTO, 2008). Como todos os nós selecionam sempre a cadeia mais longa, eventualmente a rede global irá convergir para um estado consistente. Logo após a divisão da rede em duas perspectivas, cada minerador passa a trabalhar imediatamente na solução do bloco subsequente ao bloco que acredita pertencer à cadeia principal. Dessa forma, um grupo de mineradores estará trabalhando para dar continuidade à cadeia que termina por um bloco candidato, enquanto outro grupo estará trabalhando para estender a outra cadeia. Eventualmente a solução para um novo bloco será encontrada, fazendo com que uma das cadeias se torne mais longa do que a outra. Nesse momento, todos os nós da rede passam a reconhecer a cadeia mais longa como cadeia principal, e a *blockchain* voltará ao estado consistente.

Qualquer alteração em uma transação pertencente a um bloco já resolvido exige uma nova solução para o desafio criptográfico do bloco. Isso acontece porque qualquer modificação traz como consequência uma alteração na Raiz de Merkle, fazendo com que a solução do desafio anterior seja inválida para o novo cabeçalho do bloco. A alteração do identificador de um bloco faz com que a referência contida em seu bloco filho aponte para um bloco inexistente. Portanto, para que um usuário modifique uma transação, ele deve refazer a prova-de-trabalho do próprio bloco e de todos os blocos subsequentes a ele. Ainda, o usuário precisa fazer com que sua nova cadeia adulterada seja a mais longa, concluindo também o desafio do último bloco que está sendo construído por toda a rede de mineradores. Para que a solução seja aceita pelos demais nós, todo esse esforço deve ser concluído antes que qualquer minerador encontre a solução para o último bloco. A alteração de uma transação já inserida na *blockchain* exige do usuário, portanto, poder de processamento superior ao somatório do poder de processamento de toda a rede global de mineradores, e por esse motivo, um registro inserido na *blockchain* é considerado imutável.

5 INSERÇÃO DE DADOS NA BLOCKCHAIN

Neste capítulo será feita uma análise a respeito da utilização da *blockchain* do protocolo Bitcoin para a inserção de informações de propósitos diversos. Serão detalhados os métodos existentes, bem como as limitações da tecnologia – tendo como ponto de vista a quantidade de dados que podem ser inseridos por registro, o limite teórico de novos registros por unidade de tempo, e o custo envolvido.

5.1 Métodos

As primeiras aplicações que se propuseram a utilizar a *blockchain* para outros fins utilizavam o *script* de saída das transações, inserindo informações no campo destinado ao preenchimento do endereço do destinatário. A solução é controversa, uma vez que cria transações com saídas inválidas. Quando propagadas, as transações são inseridas no conjunto UTXO (conjunto de todas as saídas não gastas da rede), mas como não possuem um destinatário válido, não há como retirá-las da lista. Isso faz com que a lista de saídas não gastas aumente indefinidamente, sobrecarregando o sistema (ANTONOPOULOS, 2014).

Em 2014, uma atualização no cliente *Bitcoin Core* introduziu um novo operador que poderia ser utilizado como *script* de saída, chamado OP_RETURN, permitindo aos desenvolvedores adicionarem informações diversas na saída de uma transação (BARTOLETTI; POMPIANU, 2017). O operador OP_RETURN cria uma saída que explicitamente não pode ser gasta, fazendo com que ela não seja inserida no conjunto UTXO. É importante ressaltar que não existe um *script* de entrada que corresponda a um *script* de bloqueio criado com o operador OP_RETURN, impossibilitando o uso futuro dos bitcoins utilizados nessa saída (ATENIESE et al., 2017). É possível, porém, criar uma saída com quantidade nula de moedas. Uma transação pode conter apenas uma saída com o operador OP_RETURN, e será rejeitada pelos participantes da rede caso tenha sido construída utilizando o operador múltiplas vezes.

Outro mecanismo para inserir dados na *blockchain* é através da utilização de *scripts* de multiassinatura M-de-N, onde a saída da transação é bloqueada com a inserção de N chaves públicas, sendo necessárias M assinaturas para o desbloqueio dos fundos (BONNEAU et al., 2015). O esquema é limitado a 15 chaves, o que significa que é possível criar qualquer combinação entre 1-de-1 e 15-de-15. A técnica utilizada para inserir dados consiste em criar uma transação de multiassinatura 1-de-N, onde apenas a primeira chave pública representa uma chave válida, e as demais são substituídas por dados quaisquer. A utilização de *scripts* de

multiassinatura para a inserção de dados na *blockchain* deve ser cautelosa, uma vez que o aumento do tamanho da transação, em bytes, exige um aumento da taxa de transação. Algumas aplicações utilizam *scripts* de multiassinatura como método complementar ao uso do operador OP_RETURN, e normalmente possuem um esquema de até 3 chaves.

5.2 Limitações

A quantidade de dados que podem ser inseridos em uma transação através do operador OP_RETURN é limitada a 80 bytes. Utilizando transações de multiassinatura 1-de-N, é possível inserir até 32 bytes de dados para cada uma das $N - 1$ chaves disponíveis. Além do espaço limitado de informações que podem ser inseridas em cada transação, existe uma taxa máxima de novas transações que a *blockchain* é capaz de processar. O protocolo Bitcoin limita o tamanho de cada bloco em 1 MB (GOBEL; KRZESINSKI, 2017). O tamanho de cada transação é variável, e depende do número de entradas e saídas, mas tipicamente possui entre 250 bytes e 500 bytes. Considerando que o intervalo de tempo entre a mineração de dois blocos seja de 10 minutos, é possível estimar que o limite teórico suportado pela *blockchain* é de 3,5 a 7 novas transações por segundo, ou 300 mil a 600 mil transações por dia.

Existem diversas propostas de melhorias para que o limite seja aumentado. Para que uma modificação seja feita no protocolo, é preciso que exista um consenso, sendo ela aceita e adotada pela maioria dos nós da rede. As principais propostas visam aumentar o tamanho máximo de cada bloco, impactando diretamente na taxa com que novas transações podem ser processadas. Outra alternativa é reduzir o intervalo de tempo entre a produção dos blocos, sendo essa normalmente rejeitada pela comunidade por sua alta complexidade. Para reduzir o intervalo é necessário alterar a dificuldade de mineração, o que poderia causar impactos na segurança das transações. Além disso, seria necessário reajustar a recompensa entregue aos mineradores em cada bloco, para preservar a oferta monetária planejada.

5.3 Custo

Sendo os dados inseridos na *blockchain* através de transações da moeda, o custo de inserção representa a taxa que será entregue ao minerador responsável por incluir a transação em um bloco. A taxa pode ser escolhida pelo usuário no momento em que a transação é criada, porém, é importante salientar que taxas muito pequenas ou nulas podem fazer com que a in-

clusão da transação em um bloco seja postergada indefinidamente. Isso ocorre quando existem muitas transações em espera na *mempool*, fazendo com que os mineradores deem preferência às transações que contêm as maiores taxas.

A escolha de uma taxa adequada deve levar em consideração a quantidade momentânea de transações que estão sendo enviadas à rede, e a taxa média das transações que estão sendo processadas. Existem ferramentas, como a Bitcoin Fees[1], que fazem a análise das transações atuais e conseguem estimar o tempo médio necessário para que uma transação seja confirmada para cada taxa específica. Outras ferramentas, como a BitInfoCharts[2], disponibilizam um histórico referente à taxa média paga por transação pelos usuários da rede.

[1]Disponível em: <https://bitcoinfees.earn.com/>. Acesso em: 4 de janeiro de 2018.
[2]Disponível em: <https://bitinfocharts.com/>. Acesso em: 1 de janeiro de 2018.

6 APLICAÇÃO PROPOSTA

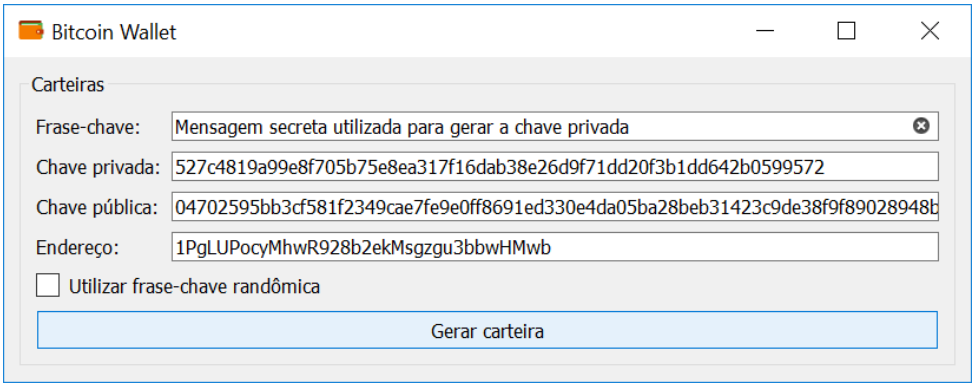
Neste capítulo serão apresentadas as funcionalidades da aplicação implementada em detalhes, e a maneira como foram desenvolvidas. Serão apresentados também os resultados obtidos através do registro de um arquivo de teste, e da posterior verificação de seu registro. Por fim, será apresentada uma análise a respeito da viabilidade de utilizar a *blockchain* do Bitcoin para este propósito.

6.1 Geração de chaves e endereços

Para utilizar o sistema de registros implementado neste trabalho, é necessária a posse de uma chave privada e de seu respectivo endereço Bitcoin, com fundos disponíveis para cobrir o custo de registro. O controle sobre um endereço é essencial, uma vez que os registros são inseridos na *blockchain* através de transações da moeda, que exigem acesso a uma chave privada válida para serem criadas. Para compor o *script* de entrada de uma nova transação, é necessário possuir a chave privada de um endereço que esteja referenciado em alguma UTXO, o que possibilita a utilização dos fundos disponíveis.

A aplicação, desenvolvida através da linguagem de programação Python 3.6.2, permite a criação de chaves privadas através de uma frase-chave inserida pelo usuário, ou através de uma frase-chave aleatória, conforme exibido na Figura 6.1. Quando uma mensagem é inserida manualmente, a chave privada é calculada obtendo-se o resumo do texto utilizando o algoritmo SHA-256. Caso seja escolhida a opção de frase-chave randômica, a chave privada é gerada através da aplicação da mesma função *hash* criptográfica sobre um conjunto de dados pseudo-aleatórios.

Figura 6.1: Geração de chave privada, chave pública e endereço Bitcoin



Fonte: Elaborado pelo autor

Com o auxílio da biblioteca *pybitcointools*, a chave pública é calculada através de multiplicação em curvas elípticas. São utilizadas a curva e o ponto gerador estabelecidos no padrão secp256k1, onde a chave pública é gerada através da multiplicação da chave privada pelo ponto gerador. Por fim, o endereço Bitcoin consiste na aplicação em sequência das funções SHA-256 e RIPEMD-160 na chave pública, sendo o resultado final codificado em Base58Check. Para a efetivação de um registro, o usuário deve transferir fundos para o endereço gerado, de forma a permitir que os registros sejam processados pelos mineradores da rede.

6.2 Registro de arquivos na blockchain

O ato de registrar um arquivo na *blockchain*, para a aplicação proposta, significa inserir na cadeia de blocos um identificador único do arquivo, que possa futuramente provar sua existência na data de registro. O identificador utilizado é o resultado da função *hash* SHA-256 aplicada ao conteúdo do arquivo, que retorna uma sequência de 32 bytes representando seu resumo. Dadas as propriedades das funções *hash* criptográficas, é extremamente difícil encontrar qualquer outro arquivo que produza o mesmo resumo, fazendo com que o identificador possa ser considerado exclusivo.

Como a *blockchain* contém apenas blocos de transações, um registro de arquivo deve ser feito através de uma transação Bitcoin. O método utilizado pela aplicação desenvolvida consiste em inserir o identificador do arquivo após o operador OP_RETURN, no *script* de saída. Para criar uma transação, é necessário fornecer um endereço Bitcoin junto de sua chave privada, e definir uma taxa a ser paga pelo registro, que será entregue ao minerador responsável por inserir a transação em um novo bloco.

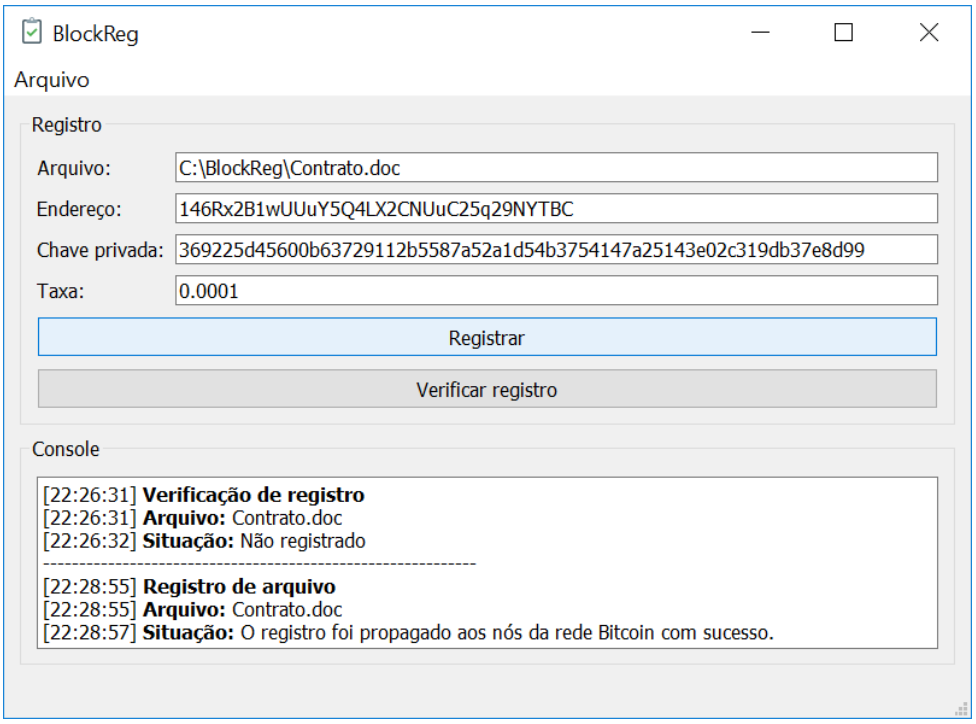
Para compor a entrada da transação, o *software* faz uma busca no conjunto UTXO da rede, verificando se existem saídas não gastas de transações passadas referenciando o endereço fornecido. A aplicação desenvolvida não se conecta diretamente à rede *peer-to-peer* do Bitcoin, e, portanto, efetua a busca utilizando a API disponibilizada por BlockCypher[1]. Caso o somatório dos valores de saída das transações encontradas seja maior ou igual à taxa escolhida, o endereço fornecido possui fundos suficientes, e o registro será produzido. Para compor a saída da transação, são utilizados dois *scripts*. O primeiro consiste no operador OP_RETURN seguido do identificador do arquivo a ser registrado, e o segundo consiste no próprio endereço fornecido, para que receba o troco da transação – que é calculado como a diferença entre os valores de entrada e a taxa escolhida.

[1]Disponível em: <https://www.blockcypher.com/>. Acesso em: 10 de novembro de 2017.

A criação de uma transação não implica em seu envio imediato à *blockchain*. Em realidade, a aplicação gera uma transação serializada, ou seja, um código binário que contém todos os dados referentes à transação. A API de BlockCypher é utilizada novamente para propagar a transação serializada através da rede. Caso a taxa escolhida seja compatível com a taxa média paga por transação no momento do registro, a transação criada será inserida em um bloco por um minerador, efetivando o registro do arquivo.

A Figura 6.2 exibe a interface do *software* de registro. Uma vez especificado o arquivo, é possível efetuar seu registro na *blockchain*, bem como verificar a existência de registro. Como exemplo, foi carregado o arquivo *Contrato.doc*. Conforme exibido no console da aplicação, uma verificação prévia foi realizada, mostrando que o arquivo em questão não possuía registro na *blockchain*. Foram inseridos um endereço e sua chave privada correspondente, gerados através da aplicação desenvolvida para o gerenciamento de carteiras. A taxa escolhida foi de 0,0001 BTC.

Figura 6.2: Verificação e registro de arquivo na blockchain



Fonte: Elaborado pelo autor

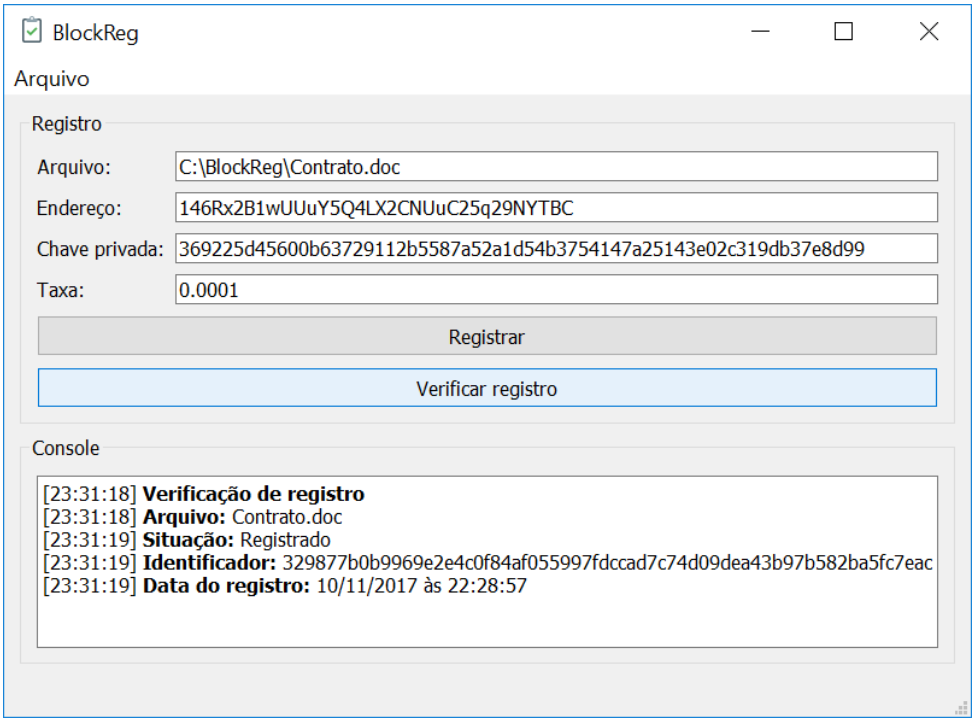
Após ter sido pressionado o botão de registro, uma mensagem foi exibida no console, informando que a transação foi propagada através da rede com sucesso. Apesar de propagada, a transação não é imediatamente inserida em um bloco, evento que depende de diversos fatores como a taxa de transação escolhida e a quantidade de transações que estão sendo propagadas no momento do envio. O registro torna-se válido somente depois de a transação ter sido incluída

em um bloco por um minerador da rede, o que deve ocorrer em até 10 minutos, caso a taxa de transação escolhida seja adequada.

6.3 Prova de existência

A aplicação implementada permite importar um arquivo qualquer a fim de verificar se o seu registro está inserido na *blockchain*, fornecendo uma prova de existência do arquivo na data de registro, em caso positivo. Para isso, o arquivo carregado na aplicação tem seu resumo calculado através a mesma função *hash* criptográfica utilizada no processo de registro. Uma busca é feita por toda a cadeia de blocos, procurando encontrar uma transação que contenha no *script* de saída o operador OP_RETURN seguido do resumo calculado. Para varrer a *blockchain* buscando por uma sequência específica de caracteres inseridos após o operador OP_RETURN, a aplicação faz o uso da API disponibilizada por SmartBit[2]. Se a transação for encontrada, são exibidas a data e a hora de registro – que correspondem ao momento em que o registro foi propagado através da rede – além do identificador da transação que contém as informações registradas. A Figura 6.3 mostra o resultado da verificação de registro do arquivo *Contrato.doc*, que foi anteriormente registrado.

Figura 6.3: Prova de existência de arquivo registrado

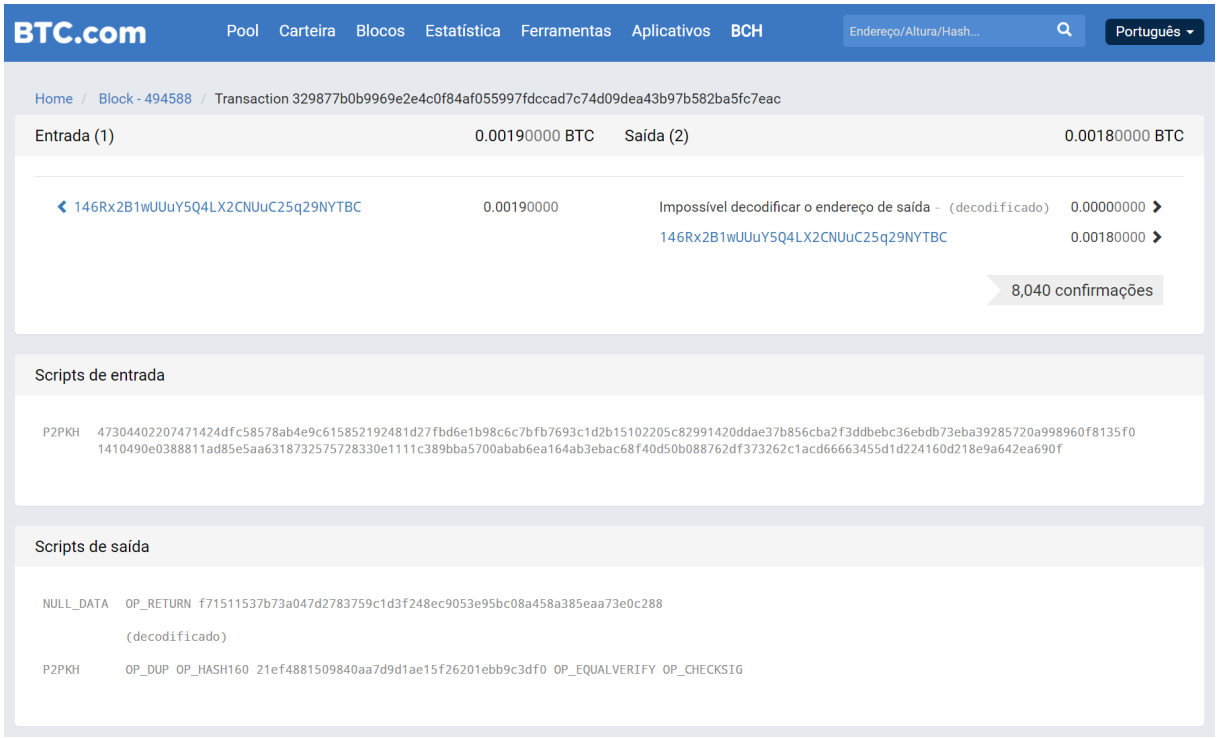


Fonte: Elaborado pelo autor

[2]Disponível em: <http://smartbit.com.au/>. Acesso em: 10 de novembro de 2017.

Existem diversas ferramentas, chamadas de *Block Explorers*, capazes de exibir informações a respeito de blocos, transações ou endereços Bitcoin específicos. A partir do identificador exibido no console para arquivos registrados, é possível verificar a existência da transação inserindo o identificador em qualquer *Block Explorer* desejado. A Figura 6.4 exibe uma consulta referente ao identificador de registro do arquivo *Contrato.doc*, realizada através da ferramenta BTC.com[3]. É possível averiguar que o *script* de saída contém o resumo do arquivo após o operador OP_RETURN.

Figura 6.4: Verificação de registro através de um explorador de blocos



Fonte: Adaptado de BTC (2017)

Tendo sido registrado um arquivo, a confiança na implementação desta aplicação não se faz necessária para sua conferência, uma vez que qualquer usuário pode obter de maneira independente a prova de existência do arquivo na data de registro, bastando para isso seguir os passos realizados pela aplicação. Através de uma cópia do arquivo original, o usuário poderá calcular o resumo do arquivo com o algoritmo SHA-256 e utilizar qualquer ferramenta de sua preferência para varrer a *blockchain*, verificando se existe alguma transação que contenha o mesmo resumo em seu *script* de saída, após o operador OP_RETURN.

De acordo com a propriedade de resistência à segunda pré-imagem da função *hash* criptográfica utilizada, é computacionalmente inviável encontrar um segundo arquivo diferente que produza o mesmo resumo do primeiro, o que garante que o identificador exibido na transação,

[3]Disponível em: <https://btc.com/>. Acesso em: 22 de dezembro de 2017.

de fato, pertence ao arquivo registrado. Como também é computacionalmente inviável adulterar transações inseridas na *blockchain*, a transação encontrada prova a existência do arquivo na data em que foi enviado à rede.

6.4 Viabilidade da aplicação

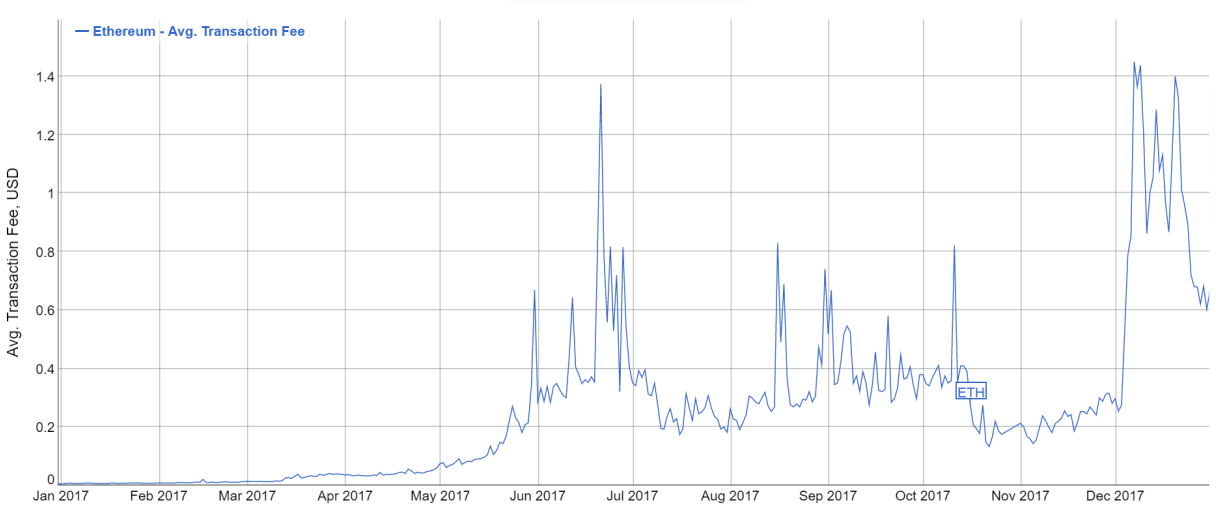
Do ponto de vista econômico, a viabilidade de utilizar a *blockchain* do Bitcoin para registro de arquivos foi comprometida durante o desenvolvimento deste trabalho devido ao aumento excessivo da taxa média de transação, conforme pode ser visualizado na Figura 6.5. O aumento foi causado pela forte demanda pelo mercado de criptomoedas durante o ano de 2017. No primeiro dia do ano, a taxa média paga foi de apenas $ 0,35 por transação, o que permitia o registro de arquivos a um custo relativamente baixo. Ao longo do ano, o aumento excessivo no número de transações, aliado às limitações atuais de escalabilidade da rede Bitcoin, elevaram as taxas a um patamar que inviabilizou transações monetárias de valores pequenos. No último dia do ano, a taxa média paga foi de $ 25,17 por transação, representando um aumento de 7091% no período de 2017. Quando a demanda ultrapassa os limites suportados pelo protocolo, somente as transações com as maiores taxas serão processadas.

Figura 6.5: Taxa média paga por transação na rede Bitcoin no ano de 2017

Fonte: BitInfoCharts (2018)

O problema do alto custo de registro atual pode ser solucionado conforme novas propostas de melhorias no protocolo Bitcoin forem aceitas, aumentando a capacidade de processamento de novas transações na rede. Como alternativa, a aplicação desenvolvida pode ser adaptada para a utilização de outras *blockchains*, mantidas por usuários de outras criptomoedas.

Figura 6.6: Taxa média paga por transação na rede Ethereum no ano de 2017



Fonte: BitInfoCharts (2018)

Existem diversas propostas em funcionamento, sendo a *blockchain* da plataforma Ethereum a mais utilizada atualmente em aplicações externas. Conforme exibido na Figura 6.6, apesar de o custo médio das transações realizadas na rede Ethereum também ter sofrido forte aumento no período de 2017, a utilização da plataforma se mostra uma alternativa ao uso do protocolo Bitcoin. No fechamento do período analisado, enquanto os usuários da rede Bitcoin despenderam uma taxa média de $ 25,17 por transação, os usuários da rede Ethereum gastaram, em média, $ 0,66 por transação.

7 CONCLUSÃO

A tecnologia *blockchain* foi desenvolvida com o objetivo de permitir a transferência de moedas virtuais sem a necessidade da confiança em um intermediador. Para isso, conta com a contribuição dos próprios usuários do sistema para a construção de um registro público e distribuído de todas as transações que já ocorreram na rede, impedindo que um usuário consiga transferir a mesma moeda múltiplas vezes. A *blockchain* utiliza a criptografia para garantir a segurança, a transparência e a imutabilidade dos registros de transações, fazendo o uso de um mecanismo que torna cada vez mais improvável a possibilidade de adulteração de dados armazenados conforme novos registros são inseridos. Dadas suas características, a tecnologia tornou-se alvo de estudos que visam expandir seu uso para diversas áreas, que não se restringem ao setor financeiro.

Neste trabalho foi implementada uma aplicação que utiliza a tecnologia *blockchain* do protocolo Bitcoin para promover o registro de arquivos digitais em uma rede descentralizada, sendo capaz de fornecer uma prova de existência dos arquivos na data em que foram registrados. O registro de um arquivo consiste em inserir no *script* de saída de uma transação um identificador exclusivo do arquivo, que é calculado através da função *hash* SHA-256 aplicada ao seu conteúdo. Para verificar a existência de um registro, é necessário carregar o arquivo na aplicação, que faz uma busca pela *blockchain* visando encontrar uma transação que contenha seu identificador exclusivo. Caso tenha sido encontrado, o identificador da transação e a data de registro são exibidos no console da aplicação. Através do identificador fornecido, é possível verificar a existência do *hash* do arquivo na transação, assim como a data de registro, utilizando qualquer ferramenta capaz de explorar os dados da *blockchain*.

Após a efetivação de um registro, a confiança na aplicação desenvolvida não se faz necessária para sua verificação, uma vez que a prova de existência pode ser obtida de maneira independente por qualquer usuário que possua uma cópia do arquivo original. Para isso, basta que o identificador do arquivo seja calculado através da função SHA-256, e que uma busca pelo identificador seja realizada na *blockchain* utilizando qualquer ferramenta desejada. As técnicas utilizadas pela *blockchain* para garantir a imutabilidade dos registros, somadas à propriedade da resistência à segunda pré-imagem das funções *hash* criptográficas, provam a existência do arquivo na data armazenada no cabeçalho do bloco em que a transação está inserida. A integridade de um arquivo também pode ser confirmada através de seu registro, visto que qualquer modificação no arquivo original provoca uma alteração em seu identificador, que não corresponderá ao identificador contido na transação de registro.

A criação de um registro exige a posse de um endereço Bitcoin e sua correspondente chave privada. Para tanto, foi implementada uma segunda aplicação capaz de gerar chaves e endereços. A chave privada consiste no *hash* de uma mensagem secreta ou de um número pseudo-aleatório, utilizando a função SHA-256. A chave pública é calculada através da multiplicação em curva elíptica da chave privada por um ponto gerador. São utilizadas a curva e o ponto gerador estabelecidos no padrão secp256k1. Por fim, o endereço Bitcoin é calculado através da aplicação em sequência das funções SHA-256 e RIPEMD-160 na chave pública, sendo o resultado codificado em Base58Check. Para a efetivação do registro, é necessário realizar o pagamento de uma taxa, que será destinada ao minerador responsável por inserir na *blockchain* a transação que contém o registro. O usuário deve transferir fundos suficientes para o endereço criado, a fim de permitir o pagamento da taxa de registro.

Para que fosse possível a compreensão das técnicas utilizadas pela *blockchain* para garantir suas características, um discorrimento detalhado a respeito do protocolo Bitcoin foi realizado ao longo deste trabalho, tendo como foco o funcionamento das transações e da cadeia de blocos, e os métodos existentes para inserir dados na *blockchain*. Por fim, foram apresentados os detalhes acerca da aplicação desenvolvida, exibindo um exemplo de arquivo que teve seu registro inserido com êxito na cadeia de blocos.

Do ponto de vista técnico, a implementação da aplicação proposta foi realizada com sucesso, atendendo corretamente aos propósitos de registro e posterior prova de existência de arquivos digitais. Entretanto, durante o desenvolvimento deste trabalho, houve um aumento expressivo na taxa média paga por transação, devido a um forte aumento da demanda pelo mercado de bitcoins no ano de 2017. No período de um ano, a taxa média paga pelos usuários subiu de $ 0,35 para $ 25,17 por transação, inviabilizando movimentações de pequenos valores. Como os registros na *blockchain* são efetivados através de transações, o uso da cadeia de blocos do Bitcoin tornou-se economicamente inviável para este propósito. O problema pode ser contornado utilizando-se *blockchains* alternativas, como a da plataforma Ethereum, que atualmente oferece taxas mais atrativas.

REFERÊNCIAS

ANTONOPOULOS, A. M. **Mastering bitcoin: unlocking digital cryptocurrencies**. [S.l.]: O'Reilly Media, Inc., 2014.

ATENIESE, G. et al. Certified bitcoins. In: SPRINGER. **International Conference on Applied Cryptography and Network Security**. [S.l.], 2014. p. 80–96.

ATENIESE, G. et al. Redactable blockchain–or–rewriting history in bitcoin and friends. In: IEEE. **Security and Privacy (EuroS&P), 2017 IEEE European Symposium on**. [S.l.], 2017. p. 111–126.

BARTOLETTI, M.; POMPIANU, L. An analysis of bitcoin op_return metadata. **Cornell University Library**, 2017. Disponível em: <https://arxiv.org/abs/1702.01024>. Acesso em: 17 de dezembro de 2017.

BITINFOCHARTS. 2018. Disponível em: <https://bitinfocharts.com/comparison/bitcoin-transactionfees.html>. Acesso em: 1 de janeiro de 2018.

BONNEAU, J. et al. Sok: research perspectives and challenges for bitcoin and cryptocurrencies. In: IEEE. **Security and Privacy (SP), 2015 IEEE Symposium on**. [S.l.], 2015. p. 104–121.

BTC. 2017. Disponível em: <https://btc.com/329877b0b9969e2e4c0f84af055997fdccad7c74d09dea43b97b582ba5fc7eac>. Acesso em: 22 de dezembro de 2017.

CHAUM, D. Blind signatures for untraceable payments. In: SPRINGER. **Advances in cryptology**. [S.l.], 1983. p. 199–203.

CHAUM, D.; FIAT, A.; NAOR, M. Untraceable electronic cash. In: SPRINGER-VERLAG NEW YORK, INC. **Proceedings on Advances in cryptology**. [S.l.], 1990. p. 319–327.

CURRAN, T.; GEIST, D. Using the bitcoin blockchain as a botnet resilience mechanism. 2016. Disponível em: <https://www.os3.nl/_media/2016-2017/courses/ot/dana_tom.pdf>. Acesso em: 21 de dezembro de 2017.

DAI, W. **B-money**. 1998. Disponível em: <http://www.weidai.com/bmoney.txt>. Acesso em: 30 de junho de 2017.

DECKER, C.; WATTENHOFER, R. Bitcoin transaction malleability and mtgox. In: SPRINGER. **European Symposium on Research in Computer Security**. [S.l.], 2014. p. 313–326.

DELGADO-SEGURA, S. et al. Analysis of the bitcoin utxo set. 2017. Disponível em: <https://eprint.iacr.org/2017/1095.pdf>. Acesso em: 29 de dezembro de 2017.

EVERLEDGER. 2017. Disponível em: <https://www.everledger.io/>. Acesso em: 18 de novembro de 2017.

GIECHASKIEL, I.; CREMERS, C.; RASMUSSEN, K. B. On bitcoin security in the presence of broken cryptographic primitives. In: SPRINGER. **European Symposium on Research in Computer Security**. [S.l.], 2016. p. 201–222.

GIRAULT, M. Self-certified public keys. In: SPRINGER. **Advances in Cryptology—EUROCRYPT'91**. [S.l.], 1991. p. 490–497.

GOBEL, J.; KRZESINSKI, A. Increased block size and bitcoin blockchain dynamics. In: IEEE. **2017 27th International Telecommunication Networks and Applications Conference (ITNAC)**. [S.l.], 2017. p. 1–6.

KARAME, G. O.; ANDROULAKI, E.; CAPKUN, S. Double-spending fast payments in bitcoin. In: ACM. **Proceedings of the 2012 ACM conference on Computer and communications security**. [S.l.], 2012. p. 906–917.

KO, K. et al. New public-key cryptosystem using braid groups. In: SPRINGER. **Advances in cryptology—CRYPTO 2000**. [S.l.], 2000. p. 166–183.

KOBLITZ, N. Elliptic curve cryptosystems. **Mathematics of computation**, v. 48, n. 177, p. 203–209, 1987.

KROLL, J. A.; DAVEY, I. C.; FELTEN, E. W. The economics of bitcoin mining, or bitcoin in the presence of adversaries. In: **Proceedings of WEIS**. [S.l.: s.n.], 2013. v. 2013.

LAMPORT, L.; SHOSTAK, R.; PEASE, M. The byzantine generals problem. **ACM Transactions on Programming Languages and Systems (TOPLAS)**, 1982.

LUA, E. K. et al. A survey and comparison of peer-to-peer overlay network schemes. **IEEE Communications Surveys & Tutorials**, IEEE, v. 7, n. 2, p. 72–93, 2005.

MILLER, V. S. Use of elliptic curves in cryptography. In: SPRINGER. **Conference on the Theory and Application of Cryptographic Techniques**. [S.l.], 1985. p. 417–426.

NAKAMOTO, S. **Bitcoin: a peer-to-peer electronic cash system**. 2008. Disponível em: <https://bitcoin.org/bitcoin.pdf>. Acesso em: 30 de junho de 2017.

OLIVEIRA, J. G. de. **Curvas Elípticas sobre Corpos Finitos e Criptografia de Chave Pública**. 2009. Disponível em: <https://www.sbm.org.br/docs/coloquios/CO-1-04.pdf>. Acesso em: 14 de dezembro de 2017.

ORIGINALMY. 2017. Disponível em: <https://originalmy.com/>. Acesso em: 24 de outubro de 2017.

PAYPAL. 2017. Disponível em: <https://www.paypal.com/>. Acesso em: 28 de outubro de 2017.

PRENEEL, B. Cryptographic hash functions. **Transactions on Emerging Telecommunications Technologies**, Wiley Online Library, v. 5, n. 4, p. 431–448, 1994.

RIVEST, R. L.; SHAMIR, A.; ADLEMAN, L. A method for obtaining digital signatures and public-key cryptosystems. **Communications of the ACM**, ACM, v. 21, n. 2, p. 120–126, 1978.

SLEIMAN, M. D.; LAUF, A. P.; YAMPOLSKIY, R. Bitcoin message: Data insertion on a proof-of-work cryptocurrency system. In: IEEE. **Cyberworlds (CW), 2015 International Conference on**. [S.l.], 2015. p. 332–336.

STORJ. 2017. Disponível em: <https://storj.io/>. Acesso em: 18 de novembro de 2017.

SZABO, N. **Bit gold**. 2008. Disponível em: <http://unenumerated.blogspot.com.br/2005/12/bit-gold.html>. Acesso em: 30 de junho de 2017.

TAYLOR, M. B. Bitcoin and the age of bespoke silicon. In: IEEE PRESS. **Proceedings of the 2013 International Conference on Compilers, Architectures and Synthesis for Embedded Systems**. [S.l.], 2013. p. 16.

THOMSEN, S. S.; KNUDSEN, L. R. **Cryptographic hash functions**. Thesis (PhD) — Technical University of DenmarkDanmarks Tekniske Universitet, Department of Applied Mathematics and Computer ScienceInstitut for Matematik og Computer Science, 2005.

TSCHORSCH, F.; SCHEUERMANN, B. Bitcoin and beyond: a technical survey on decentralized digital currencies. **IEEE Communications Surveys & Tutorials**, 2016.

VALLOIS, V.; GUENANE, F. A. Bitcoin transaction: From the creation to validation, a protocol overview. In: **2017 1st Cyber Security in Networking Conference (CSNet)**. [S.l.: s.n.], 2017. p. 1–7.

WANG, L.; PUSTOGAROV, I. Towards better understanding of bitcoin unreachable peers. **Cornell University Library**, 2017. Disponível em: <https://arxiv.org/abs/1709.06837>. Acesso em: 18 de dezembro de 2017.

APÊNDICE A — TRABALHO DE GRADUAÇÃO I

Aplicação da tecnologia blockchain para registro de arquivos

Thiago Fonseca Martins[1], **Raul Fernando Weber**[1]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Av. Bento Gonçalves, 9500 – Porto Alegre, RS – Brasil

tfmartins@inf.ufrgs.br, weber@inf.ufrgs.br

Abstract. *The recent popularization of cryptocurrencies has brought a growing demand for decentralized applications that involve the use of blockchains to ensure security and agility to its users. Blockchains work as distributed ledgers in a peer-to-peer network, with intrinsic characteristics that guarantee the immutability of stored information. This article aims to propose the implementation of an application that uses the bitcoin blockchain to register files of any nature, providing an irrefutable proof of its existence at the time it was registered. For a better undestanding of the proposal, the article presents a detailed study on the operation of the bitcoin protocol and on the blockchain mechanisms.*

Resumo. *A recente popularização das criptomoedas trouxe uma demanda crescente por aplicações descentralizadas que envolvem o uso de blockchains para garantir segurança e agilidade aos seus usuários. Blockchains funcionam como registros distribuídos em uma rede peer-to-peer, com características intrínsecas que garantem a imutabilidade das informações armazenadas. O objetivo deste artigo é propor a implementação de uma aplicação que utiliza a blockchain do bitcoin para o registro de arquivos de qualquer natureza, possibilitando o fornecimento de uma prova irrefutável de sua existência na data em que foi registrado. Para a compreensão da proposta, o artigo apresenta um estudo detalhado sobre o funcionamento do protocolo bitcoin e sobre os mecanismos da blockchain.*

1. Introdução

No sistema monetário tradicional, os bancos são responsáveis por manter o registro das transações e garantir a consistência dos dados. A principal proposta das moedas digitais é justamente eliminar a necessidade de uma autoridade central, e a dificuldade consiste em encontrar uma forma viável de fazer com que todos os participantes atuem em conjunto para garantir o correto funcionamento do sistema, evitando fraudes e ataques por parte de usuários maliciosos. Sob o controle de um banco, um usuário jamais conseguiria utilizar a mesma moeda para realizar dois pagamentos simultaneamente. Como o banco mantém um registro sequencial das transações, um dos pagamentos seria processado primeiro, e sendo aceito, invalidaria o segundo. Sem o controle de um banco, nada impede que um usuário faça cópias de suas moedas digitais e envie para diversos outros usuários simultaneamente – técnica conhecida como gasto duplo.

A ideia por trás da descentralização consiste em fazer com que cada participante do sistema armazene uma cópia do registro de transações, de forma que todos os nós da rede sejam responsáveis pela consistência dos dados. Quando uma tentativa de gasto

duplo é detectada por um nó, o mesmo deve notificar todos os demais; e uma transação é válida quando a maioria dos participantes assim a considerarem. Um sistema cujo consenso seja obtido através do voto da maioria dos nós sofre desvantagem semelhante ao problema dos generais bizantinos[1], sendo vulnerável ao chamado *Sybil Attack*, que ocorre quando um usuário malicioso simula milhares de nós com o objetivo de tomar o controle das decisões [Tschorsch and Scheuermann 2016]. No caso das moedas digitais, um atacante poderia simular tantos nós quantos fossem necessários para dominar a rede e permitir o gasto duplo de suas próprias moedas, por exemplo.

Diversas propostas de moedas digitais surgiram ao longo da história. Uma delas – nomeada *bitcoin* – ganhou destaque por ter sido a primeira a solucionar de maneira eficiente os problemas anteriormente mencionados. O desenvolvedor da moeda introduziu um conceito inovador de estrutura de dados pública e distribuída chamada *blockchain*, que armazena todas as transações do sistema, e se vale da criptografia para garantir segurança, transparência e imutabilidade dos registros. A ideia foi tão revolucionária que diversos estudos posteriores surgiram com o objetivo de expandir o uso da tecnologia para outras áreas, inclusive além do setor financeiro. Existe uma grande variedade de aplicações que podem se beneficiar das propriedades da *blockchain*, incluindo sistemas de votação, contratos inteligentes, ou até mesmo armazenamento distribuído de arquivos na nuvem.

Uma aplicação particularmente interessante consiste em permitir o registro de arquivos digitais, funcionando como um cartório descentralizado, que possibilita o fornecimento de uma prova incontestável da existência dos mesmos no momento do registro. É possível registrar arquivos de qualquer natureza, como áudio, vídeo, imagem, documento de texto, ou qualquer outro formato. Algumas empresas já oferecem serviços de registro de arquivos na *blockchain*, como é o caso da OriginalMy, que fornece uma maneira rápida e prática de registrar documentos e posteriormente provar sua autenticidade e data de registro. De acordo com a empresa, contratos podem ter validade jurídica caso as partes tenham comprovado ciência de seu conteúdo no momento do registro [Osório 2016]. Para que seja possível compreender a maneira como diversas aplicações se valem das características da *blockchain*, é necessário entender seu funcionamento. Neste artigo, serão apresentadas as primeiras propostas de moedas digitais, bem como uma explicação detalhada sobre o funcionamento do *bitcoin* e da *blockchain*. Por fim, será apresentada a proposta de aplicação que será desenvolvida na segunda etapa deste trabalho.

2. Primeiras propostas de moedas digitais

As primeiras pesquisas sobre o uso da criptografia para levar às transações eletrônicas a privacidade existente na troca do dinheiro em espécie apareceram ainda na década de oitenta. Em 1983, David Chaum publicou um artigo propondo um método chamado "assinaturas cegas", onde o banco seria responsável por assinar digitalmente as transações sem conhecer o seu conteúdo, provendo assim alto nível de privacidade aos seus usuários. No artigo, foi sugerida a criação de um sistema de pagamentos com as seguintes propriedades: incapacidade do banco determinar o beneficiário, a hora e a quantidade do pagamento feito por alguém; capacidade do pagador fornecer a prova do pagamento, ou determinar

[1]Abstração utilizada para descrever situações onde o consenso é desejado em um sistema distribuído, mesmo na presença de componentes considerados "traidores", que tentam confundir os demais [Lamport et al. 1982].

a identidade do beneficiário em circunstâncias excepcionais; e capacidade de interromper o uso de moedas que foram reportadas como roubadas [Chaum 1983]. No início da década de noventa, foi especificada uma moeda eletrônica em princípio irrastreável, mas com a propriedade de permitir o rastreamento por parte do banco em casos de gasto duplo [Chaum et al. 1990]. No mesmo ano, David Chaum fundou a empresa DigiCash, onde trabalhou na implementação de suas ideias a respeito de dinheiro eletrônico.

Em 1998, preocupado com a interferência do governo no sistema monetário, Wei Dai propôs a criação de uma nova moeda digital, a qual chamou de *b-money*. O algoritmo proposto utiliza os conceitos de criptografia assimétrica, onde a chave pública é o endereço da carteira onde serão depositadas as moedas, e a chave privada é a senha pessoal para a utilização das mesmas [Dai 1998]. As moedas digitais poderiam ser geradas por qualquer pessoa, que receberia um valor equivalente ao esforço computacional empregado na criação das mesmas. Nick Szabo, em 2005, publicou um artigo expondo suas ideias acerca de uma moeda digital nomeada *bit gold*, propondo soluções para evitar o gasto duplo de moedas e garantir a sustentabilidade do sistema utilizando o conceito de prova-de-trabalho [Szabo 2008]. O autor introduziu também a ideia de "mineradores", isto é, nós da rede que receberiam recompensas em troca do poder computacional empregado na validação das transações.

3. Bitcoin

Sob o pseudônimo de Satoshi Nakamoto, foi publicado em 2008 através da *internet* um artigo especificando uma moeda digital chamada *bitcoin*. A principal inovação foi a solução proposta para resolver o problema do gasto duplo utilizando uma rede *peer-to-peer*, formando um registro distribuído de transações construído através de enorme esforço computacional, e que não pode ser modificado sem que toda a prova-de-trabalho seja refeita [Nakamoto 2008]. No *bitcoin*, o consenso não é atingido com base no voto da maioria dos nós, mas sim com base na solução que teve maior poder de processamento empregado. Como cada nó da rede precisa resolver desafios computacionais para participar do processo, um atacante não poderia tomar o controle do sistema simplesmente simulando milhares de nós. Seria preciso deter um poder computacional superior ao somatório do poder de processamento de todos os demais nós da rede. O protocolo *bitcoin* será estudado com mais detalhes nas próximas subseções.

3.1. Transações

Uma transação pode ser encarada como um registro transmitido à rede informando que o proprietário de determinada quantidade de *bitcoins* autorizou a transferência dos mesmos a outro usuário. O novo proprietário poderá então gastar seus *bitcoins* criando uma nova transação autorizando sua transferência para outro, e assim por diante, criando uma cadeia de proprietários. De um lado, cada transação possui uma ou mais entradas, e de outro, uma ou mais saídas. As entradas especificam carteiras de onde determinadas quantidades de *bitcoins* serão debitadas, enquanto as saídas informam carteiras onde os *bitcoins* serão creditados. Frequentemente, a quantidade total descrita na saída de uma transação é ligeiramente inferior à quantidade total especificada na entrada. A diferença entre esses valores representa uma pequena taxa que será entregue como recompensa ao usuário responsável por validar a transação.

Cada entrada referencia uma saída de transação anterior, e portanto, nem sempre um usuário conseguirá utilizar como entrada o valor exato que deseja transferir a outro proprietário. Caso tenha recebido diversas transferências de pequenas quantidades, e deseja realizar uma transferência de maior quantidade, o usuário deverá agrupar múltiplas entradas para compor a transação. Frequentemente, o somatório total de *bitcoins* na entrada ultrapassa o valor que se deseja transferir. Nesses casos, o usuário deverá inserir o endereço de sua própria carteira na saída, informando a quantidade de *bitcoins* da entrada que permanecerá em sua posse.

A Figura 1 exemplifica uma transação típica, onde Alice deseja transferir 0,7 BTC (abreviação da unidade monetária *bitcoin*) para Bob. Suponha que a carteira de Alice estava vazia antes de receber uma transferência de 0,5 BTC de Carol, e outra de 0,4 BTC de Carlos. Como a saída da transação que se deseja criar é superior aos valores individuais que tem à sua disposição, Alice deve referenciar na entrada ambas as transações de Carol e de Carlos. Dessa forma, a transação criada terá como entrada um total de 0,9 BTC. Alice deve inserir o endereço da carteira de Bob como saída, especificando os 0,7 BTC que deseja transferir, e além disso, inserir o seu próprio endereço informando a quantidade da entrada que deseja permanecer como proprietária. Caso não especifique seu próprio endereço na saída, os 0,2 BTC restantes serão integralmente utilizados como recompensa ao usuário responsável por validar a transação.

Entradas		**Saídas**	
Carol	0,5 BTC	Bob	0,7 BTC
Carlos	0,4 BTC	Alice	0,1999 BTC
Total	**0,9 BTC**	Total	**0,8999 BTC**
Taxa: 0,9 BTC - 0,8999 BTC = **0,0001 BTC**			

Figura 1. Transação criada por Alice para transferir 0,7 BTC para Bob. Fonte: O autor.

Os endereços das carteiras são especificados nas entradas e saídas através de *scripts*, que são responsáveis por determinar as condições necessárias para que uma moeda possa ser gasta por um proprietário. A saída de uma transação se dá na forma de um *script* que bloqueia o uso de determinada quantidade de *bitcoins*, enquanto a entrada se dá na forma de um *script* que desbloqueia o uso das moedas. Na maioria das transações, as saídas contêm o operador P2PKH (*Pay-to-Public-Key-Hash*), que trava o uso das moedas fornecendo o *hash* da chave pública do destinatário. Por consequência, somente o destinatário consegue liberar o uso das moedas, especificando no *script* de entrada sua assinatura digital e sua chave pública. Quando o *script* de entrada concatenado ao de saída produz uma sentença de de valor lógico verdadeiro, a transação é considerada válida.

Quando as ideias de uso da rede *bitcoin* para outros propósitos começaram a surgir, alguns usuários passaram a utilizar o espaço destinado ao endereço do destinatário

para inserir informações de outra natureza. Isso produz um *script* de saída que, por conter um endereço inválido, não pode ser desbloqueado por qualquer outro *script* de entrada, fazendo com que a transação permaneça por tempo indeterminado na lista de transações pendentes da rede. Para resolver o problema, foi adicionado ao protocolo *bitcoin* um operador chamado OP_RETURN, que pode ser utilizado como *script* de saída para adicionar até 40 bytes de informação qualquer.

A rede *bitcoin* é uma rede *peer-to-peer*, onde cada cliente está conectado a diversos outros clientes, com o objetivo de que as transações sejam propagadas por todos os nós participantes. Sempre que um cliente recebe uma nova transação válida, imediatamente a repassa para todos os nós ao qual está diretamente conectado. Assim, a transação é rapidamente propagada por toda a rede *peer-to-peer*, atingindo uma alta porcentagem dos nós em poucos segundos [Antonopoulos 2014].

3.2. Blocos

Um bloco pode ser entendido como um conjunto de transações, juntamente com um conjunto de informações que compõem seu cabeçalho. Cada bloco é identificado por um *hash*, gerado aplicando duas vezes o algoritmo SHA256 em seu cabeçalho. Na rede *bitcoin*, o registro de transações agrupa as mesmas em blocos idealmente "minerados" a cada 10 minutos. O conceito de bloco é relativamente simples, mas de fundamental importância para a compreensão da *blockchain*, estrutura de dados distribuída responsável pelo encadeamento dos blocos de transações. O primeiro bloco criado – e portanto um ancestral comum de todos os demais – é uma estrutura especial conhecida como bloco gênesis, e está implementado diretamente no software original do *bitcoin*. O bloco gênesis contém em sua estrutura uma única transação, que diferente de todas as demais, não possui entradas, apenas saídas.

3.3. Mineração

Mesmo depois de uma transação ser propagada por toda a rede, ela ainda não fará parte da *blockchain* até que seja verificada e incluída em um bloco através de um processo chamado mineração. Cada cliente mantém uma lista local de transações não verificadas. Essas transações são empacotadas em um bloco que exige elevado poder de processamento para ser finalizado com sucesso. O processo de mineração tem dois propósitos: criar novos *bitcoins*, e garantir que transações apenas sejam confirmadas mediante o emprego de grande esforço computacional.

Os mineradores são recompensados pelo seu trabalho de duas maneiras: a primeira delas é através da criação de novas moedas, que são incluídas em cada bloco recém formado, e a segunda é através das taxas recolhidas em cada transação que compõe o bloco. A criação de novas moedas se dá através de uma transação especial criada junto com um novo bloco, chamada transação *coinbase*. A quantidade de *bitcoins* que o minerador pode adicionar em cada bloco cai para a metade a cada 210.000 blocos – aproximadamente a cada 4 anos. No momento em que o primeiro bloco foi construído, em janeiro de 2009, a recompensa era de 50 BTC. Em novembro de 2012 a recompensa diminuiu para 25 BTC, e em julho de 2016, passou a ser de 12,5 BTC por bloco.

A taxa de criação de moedas decresce exponencialmente, e baseado nessa fórmula, a expectativa é de que em 2140 todas elas já estejam em circulação, não sendo possível

criar novos *bitcoins*. Nesse momento, existirão aproximadamente 21 milhões de *bitcoins* em circulação. Atualmente, as taxas de transação representam uma porcentagem pequena da recompensa total do minerador. Porém, conforme a criação de novos *bitcoins* por bloco for diminuindo, e o número de transações por bloco for aumentando, uma porcentagem cada vez maior da recompensa virá através das taxas. Depois de 2140, quando o número de moedas em circulação já tiver atingido seu limite, os mineradores serão incentivados a continuar cooperando apenas através das taxas de transação.

3.4. Verificação de Pagamento Simplificada

Clientes que armazenam uma cópia exata de toda a *blockchain* são chamados "nós completos". Porém, não é necessário que todos os usuários mantenham um registro completo de todas as transações. Alguns clientes possuem espaço de armazenamento bastante limitado, como os que rodam em dispositivos móveis. Nesses casos, um sistema de verificação de pagamento simplificada – *Simplified Payment Verification*, ou SPV – é utilizado, onde o cliente obtém apenas os cabeçalhos dos blocos, e requisita apenas as transações necessárias para a verificação. Dessa forma, a versão da cadeia de blocos que o cliente SPV necessita tem aproximadamente um milésimo do tamanho da versão completa da *blockchain*.

Nós completos verificam as transações varrendo todos os blocos anteriores da cadeia até que o primeiro bloco tenha sido atingido. Enquanto isso, clientes SPV verificam os blocos que foram adicionados depois da transação em questão, tendo uma confirmação de que a mesma foi aceita pela rede. Para que a verificação de pagamento simplificada seja confiável, é preciso que o cliente esteja conectado a nós honestos. Não é possível enganar um cliente SPV fazendo-o acreditar que uma transação está presente em um bloco ao qual não pertence, pois a existência da transação no bloco é verificada utilizando o conceito de Árvores de *Merkle*, que será detalhado adiante. Todavia, é possível ocultar uma transação existente. Em outras palavras, um cliente SPV pode provar que uma determinada transação existe, mas não pode provar a não existência de uma outra transação – que pode ser um gasto duplo da primeira, por exemplo.

3.5. Árvores de Merkle

Árvores de *Merkle*, também conhecidas como Árvores de Dispersão, são árvores binárias utilizadas para resumir e verificar de maneira eficiente a integridade de grandes conjuntos de dados. No protocolo *bitcoin*, são utilizadas para gerar um *hash* que serve como impressão digital do conjunto completo de todas as transações em um bloco, tornando muito mais eficiente a prova da existência de uma transação no mesmo. A árvore é construída calculando-se recursivamente o *hash* dos pares de transações, até que exista somente um *hash*, chamado Raiz de *Merkle*, que é armazenada no cabeçalho do bloco. Como a árvore é binária, é necessário um número par de transações. Quando existir um número ímpar de transações em um bloco, a última transação é duplicada para que a Raiz de *Merkle* possa ser calculada. O processo de obtenção da Raiz de *Merkle* para um número ímpar de transações é exibido na Figura 2.

O mesmo processo pode ser utilizado para calcular árvores de qualquer tamanho. Atualmente, devido ao uso cada vez mais intenso do *bitcoin* como meio de pagamento, os blocos minerados na rede normalmente possuem mais de mil transações. Clientes SPV conseguem provar que uma transação pertence a um bloco utilizando a Raiz de

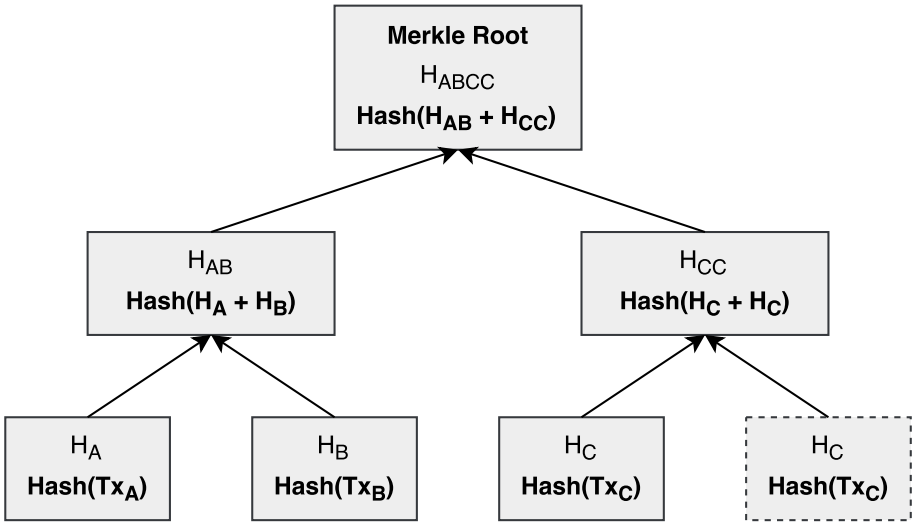


Figura 2. Representação da Árvore de *Merkle* de um bloco de transações. Fonte: Adaptado de [Antonopoulos 2014]

Merkle e um número reduzido de nodos selecionados da árvore. Conforme o exemplo exibido na Figura 3, suponha que o cliente deseja provar que a transação C pertence ao bloco. Para isso, basta que possua os nodos H_D, H_{AB} e H_{EFGH}, através dos quais é possível calcular os demais *hashes*, traçando um caminho de *Merkle* que leva até a raiz. Caso a raiz calculada seja a mesma especificada no cabeçalho do bloco, o cliente possui uma prova válida de que a transação C pertence ao mesmo. É importante observar a eficiência da verificação utilizando Árvores de *Merkle*. Para um bloco de N transações, é possível provar que uma transação específica está presente no bloco com apenas $\log_2 N$ verificações.

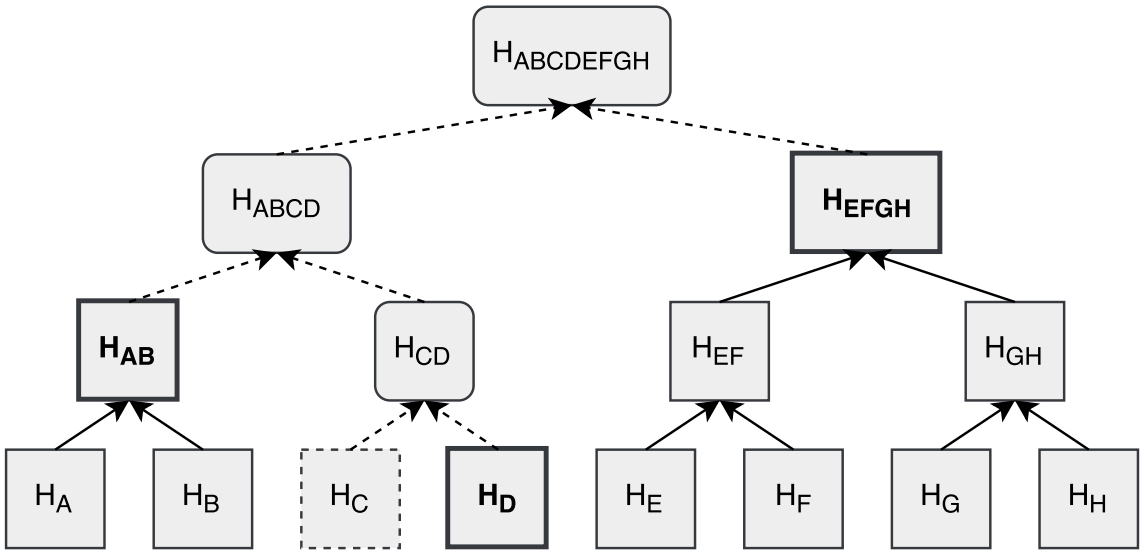


Figura 3. Caminho de *Merkle* utilizado para provar a inclusão de uma transação em um bloco. Fonte: O autor.

4. Blockchain

Cada bloco de transações possui um campo em seu cabeçalho referenciando o identificador do bloco anterior. A sequência de identificadores conectando cada bloco ao seu bloco pai cria uma cadeia – chamada *blockchain* – que vai em direção ao primeiro bloco gerado, conhecido como bloco gênesis. A Figura 4 exibe uma versão simplificada dos blocos de transações, mostrando a forma como os mesmos são encadeados. A *blockchain* pode ser considerada como um registro descentralizado de informações a respeito de todas as transações que ocorreram na rede *bitcoin*. Milhares de cópias da *blockchain* estão distribuídas por toda a rede, e a cadeia de blocos é construída de forma que quanto mais antigo for um registro, maior o esforço computacional necessário para que o mesmo seja modificado, fazendo com que ela possa ser considerada imutável. Para compreender o mecanismo de funcionamento da *blockchain*, entender o conceito de prova-de-trabalho é de fundamental importância, e será explicado em detalhes na subseção seguinte.

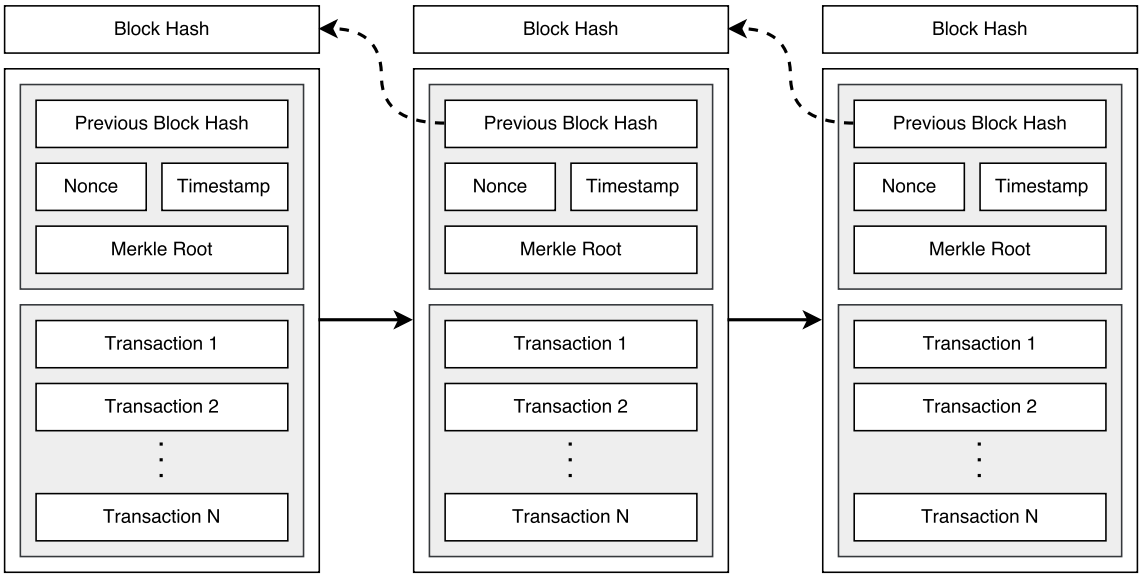


Figura 4. Simplificação da *blockchain*. Fonte: O autor.

4.1. Prova-de-trabalho

O conceito de prova-de-trabalho foi desenvolvido na década de noventa com o intuito de combater o abuso de *spam*, embora nunca tenha sido amplamente utilizado para esse fim [Bonneau et al. 2015]. A ideia por trás é simples: um usuário deve ser capaz de provar que realizou alguma tarefa, recebendo assim o direito de executar determinada ação. A tarefa consiste em resolver um desafio criptográfico que deve ser computacionalmente custoso, mas que possa ser facilmente verificado pelo avaliador. Em outras palavras, o usuário consegue provar através da solução do desafio que dedicou tempo suficiente na tarefa proposta.

Todos os nós da rede que contribuem com o processo de mineração estão em uma constante disputa global para solucionar o desafio criptográfico do novo bloco de transações que está sendo construído. Um dos parâmetros presentes no cabeçalho do bloco é um número de 32-bit chamado *nonce*, e o desafio consiste em repetidamente calcular o *hash* do cabeçalho variando esse valor, até que o resultado satisfaça determinada

condição. Mais especificamente, o *hash* do cabeçalho é calculado utilizando duas vezes o algoritmo SHA256, e a condição necessária é que o resultado seja menor do que um valor alvo pré-determinado. Assim que um cliente consegue solucionar o desafio, o bloco é distribuído através da rede, e sendo válido, é finalmente inserido na *blockchain*.

A dificuldade de criação de um bloco é proporcional ao valor alvo. A cada 2016 blocos – aproximadamente duas semanas – o valor alvo é ajustado para que a taxa de criação de novos blocos mantenha-se em aproximadamente 10 minutos. O valor alvo é definido por

$$T = T_{anterior} \cdot \left(\frac{t_{atual}}{2016 \cdot 10\, min} \right)$$

onde $T_{anterior}$ é o antigo valor, e t_{atual} é o tempo gasto para gerar os últimos 2016 blocos. Se os últimos 2016 blocos foram gerados em um tempo menor do que duas semanas, significa que o poder de processamento geral dos nodos aumentou, e então a dificuldade deve ser também aumentada [Tschorsch and Scheuermann 2016].

Para modificar um bloco já existente na *blockchain*, é necessário refazer a prova-de-trabalho, gerando um novo identificador. Por possuírem em seu cabeçalho o identificador do bloco anterior, essa modificação faz com que, em um efeito cascata, todos os blocos subsequentes precisem ser recalculados. Na rede *bitcoin*, uma transação é considerada confirmada quando seu bloco é finalizado e aceito pelos demais participantes. A partir disso, essa transação recebe uma nova confirmação a cada novo bloco posteriormente construído. Como refazer a prova-de-trabalho requer um enorme esforço computacional, a existência de uma longa cadeia de blocos faz com que a *blockchain* possa ser considerada imutável. Por convenção, uma transação é considerada aceita após atingir seis confirmações, o que leva em torno de uma hora. Quanto mais confirmações existirem, mais difícil e improvável se torna uma modificação posterior. As novas moedas criadas pela transação *coinbase* podem ser gastas depois de 100 confirmações. A partir desse momento, é praticamente impossível que a transação possa ser modificada, e para todos os efeitos práticos, ela pode ser considerada permanente.

4.2. Forks

Como a *blockchain* é uma estrutura descentralizada, nem sempre todos os nós da rede terão a mesma cópia exata a cada instante de tempo. Cada bloco possui apenas um pai, mas em algumas situações temporárias poderá ter múltiplos filhos, criados em ramificações diferentes da *blockchain*. A situação descrita é conhecida como *fork*, e acontece quando dois mineradores resolvem quase simultaneamente o algoritmo de prova-de-trabalho do bloco em que estão trabalhando. Assim, como nenhum deles sabe que outro bloco de mesmo número foi descoberto, ambos propagam seu bloco recém formado através da rede. Dependendo de diversos fatores da rede, cada cliente receberá primeiro uma ou outra solução, fazendo com que a *blockchain* temporariamente se divida em duas cadeias diferentes, como exibido na Figura 5.

Quando um cliente recebe um bloco, imediatamente começa a trabalhar na solução do bloco subsequente. Se esse cliente recebe um segundo bloco de mesmo número, ele o conecta em uma nova cadeia e segue trabalhando no mesmo bloco em que estava, aguardando até que uma das ramificações possa ser selecionada. Para resolver o problema, o conceito de cadeia mais longa foi atribuído à ramificação construída com base no maior

esforço computacional. Eventualmente, uma das ramificações se tornará mais longa, e os nós da rede são programados para resolverem os *forks* utilizando sempre a cadeia em que foi empregado o maior poder de processamento. Quando uma ramificação se tornar mais longa, todos os nós da rede a definirão como cadeia principal, mantendo a outra cadeia como secundária. A ocorrência de *forks* é bastante comum, embora quase sempre sejam resolvidos em um único bloco [Antonopoulos 2014].

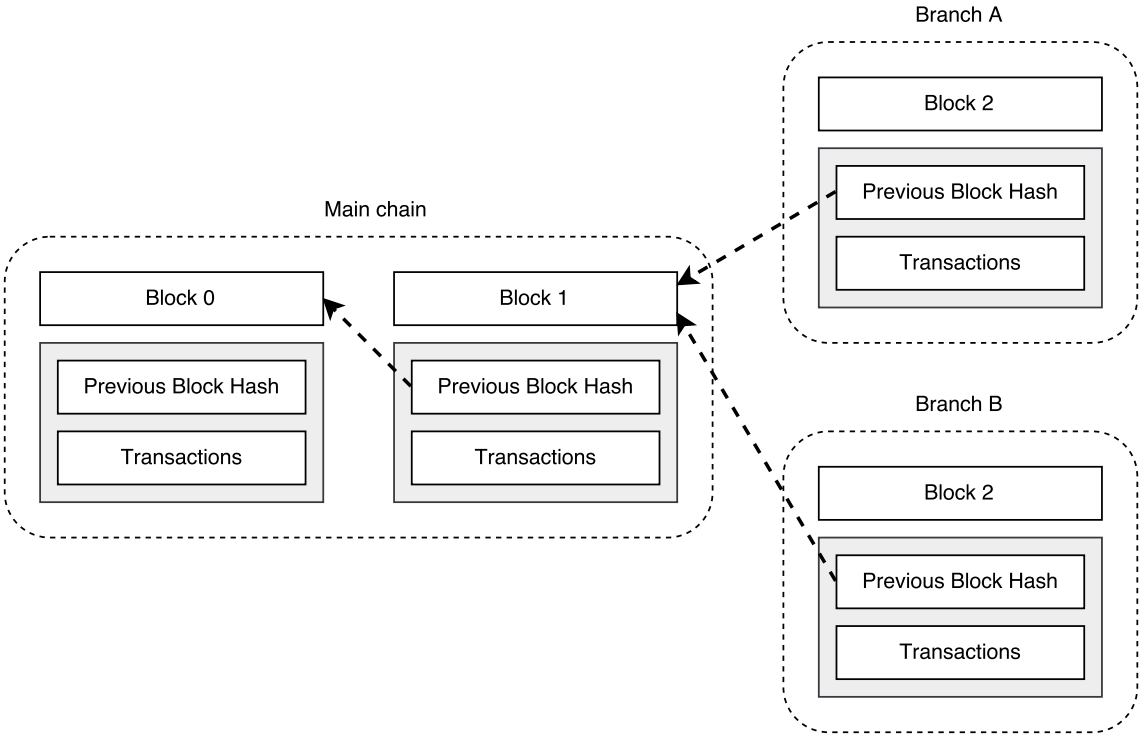


Figura 5. Representação de um *fork*. Fonte: O autor.

5. Proposta

A proposta para a segunda etapa deste trabalho é desenvolver uma aplicação que utilize a *blockchain* do *bitcoin* para o registro de arquivos de qualquer natureza, funcionando como uma espécie de cartório digital. A ideia de utilizar a *blockchain* para tal propósito é fazer com que os registros sejam distribuídos por toda a enorme rede de usuários que rodam o *software* do *bitcoin*. Assim, além de eliminar a dependência de um servidor centralizado, a imutabilidade e a segurança das informações são garantidas pelas características da *blockchain*. Como exemplo, poderia ser registrado um contrato assinado digitalmente pelas partes envolvidas, permitindo que qualquer uma delas forneça uma prova incontestável da existência do mesmo na data em que foi registrado. Todo tipo de arquivo pode ser registrado, como áudio, vídeo, imagem, documento de texto ou qualquer outro formato.

A aplicação será desenvolvida utilizando a linguagem *python*, e permitirá a entrada de arquivo de qualquer formato, utilizando o algoritmo SHA256 para produzir o *hash* do mesmo, gerando um identificador de 32 bytes. Será criada uma transação, inserindo o identificador gerado no campo OP_RETURN do *script* de saída. A aplicação irá

distribuir a nova transação criada na rede *peer-to-peer* do *bitcoin*, para que a mesma seja inserida na *blockchain*. Para que posteriormente seja obtida a prova de registro é preciso ter acesso ao arquivo original, que deverá ser inserido novamente na aplicação. O *hash* do arquivo será calculado novamente, e a aplicação fará uma busca em toda a *blockchain* até que o identificador seja encontrado no campo OP_RETURN do *script* de saída de alguma transação existente. Os dados referentes à transação e ao bloco serão fornecidos pela aplicação e podem ser conferidos utilizando qualquer outro meio de acesso à *blockchain*.

A realização da segunda parte do Trabalho de Conclusão será dividida em quatro etapas. A primeira etapa consiste na pesquisa e aprendizado de utilização das ferramentas e *APIs* necessárias para a comunicação com a rede *peer-to-peer* do *bitcoin*. A segunda etapa consiste na implementação do *software* responsável por registrar e localizar registros em toda a cadeia de blocos da rede *bitcoin*. Por fim, as duas últimas etapas consistem na escrita do trabalho, e preparação do material adequado para a apresentação do mesmo. A Tabela 1 exibe o cronograma planejado para o desenvolvimento do trabalho.

Tabela 1. Cronograma planejado para a segunda etapa do trabalho.

	Agosto	Setembro	Outubro	Novembro	Dezembro
Pesquisa	x				
Implementação		x	x		
Escrita		x	x	x	
Apresentação					x

6. Conclusão

Do ponto de vista tecnológico, o *bitcoin* trouxe inovações importantes que não se limitam apenas ao uso no setor financeiro. A ideia de uma cadeia de blocos construída de maneira descentralizada, onde qualquer usuário pode participar de sua criação, recebendo para isso recompensa pelo esforço computacional empregado e contribuindo para a confiabilidade da rede, é apontada por muitos como uma das inovações mais importantes dos últimos tempos. Para a elaboração deste artigo, foi realizada uma revisão bibliográfica visando compreender o funcionamento atual do protocolo *bitcoin* e da tecnologia *blockchain*. Por fim, foi proposta para a segunda etapa do trabalho a criação de uma aplicação que utiliza a *blockchain* para o registro de qualquer tipo de arquivo digital, sendo capaz de fornecer uma prova incontestável de sua existência na data em que foi registrado, uma vez que a modificação dos dados inseridos na cadeia de blocos se torna cada vez mais improvável na medida em que novos blocos subsequentes são inseridos.

Referências

Antonopoulos, A. M. (2014). *Mastering bitcoin: unlocking digital cryptocurrencies*. O'Reilly Media, Inc.

Bonneau, J., Miller, A., Clark, J., Narayanan, A., Kroll, J. A., and Felten, E. W. (2015). Sok: research perspectives and challenges for bitcoin and cryptocurrencies. In *Security and Privacy (SP), 2015 IEEE Symposium on*, pages 104–121. IEEE.

Buterin, V. (2014). A next-generation smart contract & decentralized application platform. *White Paper*.

Chaum, D. (1983). Blind signatures for untraceable payments. In *Advances in cryptology*, pages 199–203. Springer.

Chaum, D., Fiat, A., and Naor, M. (1990). Untraceable electronic cash. In *Proceedings on Advances in cryptology*, pages 319–327. Springer-Verlag New York, Inc.

Dai, W. (1998). B-money. *Disponível em: http://www.weidai.com/bmoney.txt. Acesso em: 30 de junho de 2017*.

Lamport, L., Shostak, R., and Pease, M. (1982). The byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)*.

Nakamoto, S. (2008). Bitcoin: a peer-to-peer electronic cash system. *Disponível em: https://bitcoin.org/bitcoin.pdf. Acesso em: 30 de junho de 2017*.

Osório, E. (2016). Blockchain: da criptomoeda ao distributed ledger. *Disponível em: http://www.blockchainview.com.br/apresentacoes/download/originalmy_edilson_osorio.pdf. Acesso em: 30 de junho de 2017*.

Szabo, N. (2008). Bit gold. *Disponível em: http://unenumerated.blogspot.com.br/2005/12/bit-gold.html. Acesso em: 30 de junho de 2017*.

Tschorsch, F. and Scheuermann, B. (2016). Bitcoin and beyond: a technical survey on decentralized digital currencies. *IEEE Communications Surveys & Tutorials*.