



UNIVERSIDADE DO VALE DO TAQUARI
CURSO DE SISTEMAS DE INFORMAÇÃO

**UTILIZANDO A BLOCKCHAIN COMO MODELO DE CONFIANÇA
PARA COMPROVAR A EXISTÊNCIA E IMUTABILIDADE DE UM
ARQUIVO DIGITAL**

Rafael Rodrigues da Silva

Lajeado, julho de 2020



Rafael Rodrigues da Silva

**UTILIZANDO A BLOCKCHAIN COMO MODELO DE CONFIANÇA
PARA COMPROVAR A EXISTÊNCIA E IMUTABILIDADE DE UM
ARQUIVO DIGITAL**

Monografia apresentada na disciplina de Trabalho de Conclusão de Curso II, do curso de Sistemas de Informação, da Universidade do Vale do Taquari – Univates, como parte da exigência para a obtenção do título de Bacharel em Sistemas de Informação.

Orientador: Prof. Dr. Alexandre Stürmer Wolf.

Lajeado, julho de 2020

AGRADECIMENTOS

À instituição de ensino Univates, essencial no meu processo de formação profissional, pela dedicação, por tudo o que aprendi ao longo dos anos do curso, pelo ambiente criativo e amigável proporcionado.

Aos professores, pelas correções e ensinamentos que me permitiram apresentar um melhor desempenho no meu processo de formação profissional ao longo do curso.

Ao professor Dr. Alexandre Stürmer Wolf, por ter sido meu orientador e ter desempenhado tal função com dedicação e amizade.

A todos os amigos por todo o apoio e pela ajuda, que muito contribuíram para a realização deste trabalho.

Em especial para minha mãe e meu pai (in memorian), pelo amor, incentivo e apoio incondicional.

A todos que direta ou indiretamente fizeram parte da minha formação, o meu muito obrigado.

RESUMO

Nos últimos anos, muito tem-se ouvido sobre criptomoedas e a sua tecnologia *blockchain*, que trouxe e apresentou um novo escopo de trabalho, principalmente para desenvolvedores e entusiastas da área de TI. Introduzida em 2008 e por trazer novas possibilidades e soluções, empresas espalhadas pelo mundo vêm implementando soluções em diversos modelos de negócios. Em suma, a *blockchain* dispõe de uma arquitetura segura, estruturada por meio de blocos criptográficos, registro de data e metadados no geral. Inicialmente foi aplicada ao setor financeiro, trazendo um novo sistema de investimento e negócio, as criptomoedas, vide a Bitcoin. Ao passar dos anos, outras soluções foram surgindo, como a Ethereum, onde por meio de *smart contracts*, permitiu-se a criação de ferramentas para diversos tipos de negócios, como por exemplo: sistemas descentralizados para o rastreamento de uma cadeia de produção (alimentos, equipamentos de informática, peças automotivas, entre outros), sistema para votações no geral, controle de histórico e dados de prontuários de hospitais, etc. O principal propósito desse trabalho, é utilizar a *blockchain* da criptomoeda Ethereum para assegurar a imutabilidade e garantir a existência de um arquivo digital por meio de uma aplicação *back-end* desenvolvida pelo autor. Tem-se como exemplos de arquivos digitais: contratos, registro de patentes e marcas, propriedade intelectual, documentos pessoais, entre outros. Para desenvolver a aplicação, foram utilizadas as linguagens de programação Solidity e JavaScript, com ajuda de bibliotecas e ferramentas específicas para o desenvolvimento de *smart contracts* e conexão com a *blockchain* Ethereum. Os principais resultados atingidos são: a prova de existência obtida através do comprovante/extrato gerado pela *blockchain* por meio de uma transação. Nela há informações sobre o registro, como por exemplo, o endereço público de quem realizou a transação, a data e hora do registro, a *hash* do documento, e outras; a possibilidade de verificar se um determinado documento está registrado na *blockchain*. Por meio da sua *hash*, caso não encontrada na *blockchain*, pode-se identificar possíveis alterações no documento original; o baixo

custo, tempo de registro e a agilidade de validação de documentos já que por estar situada na Internet, a *blockchain* pode ser acessada a qualquer momento e lugar do mundo.

Palavras-chave: Ethereum. *Blockchain*. *Smart contracts*.

ABSTRACT

In the last years, we have been listening a lot about crypto currency and its blockchain technology, which brought and shown a new scope of work, especially for developers and enthusiasts in the IT area. Introduced in 2008 worldwide and for bringing new possibilities and solutions camp, companies scattered worldwide were implementing its solution in many business models. In short, the blockchain has a secure architecture, through blocks that has cryptography, data and metadata. At the beginning it was applied in the financial area, bringing a new investment and business system, the crypto currency, more implicit the Bitcoins, but, by the fallen years, other new solutions were appearing, such as Ethereum, where through smart contracts, the creation of tools for different types of business has been allowed, such as for example: systems for tracking a production chain (food, computer equipment, automotive parts, etc.), general voting system, track history and medical records for hospitals, etc. The main purpose of this work is to use the Ethereum cryptocurrency blockchain to ensure immutability and ensure the existence of a digital file through a back-end application developed by the author. Examples of digital files include contracts, registration of patents and trademarks, intellectual property, personal documents, among others. To develop the application, Solidity and JavaScript programming languages were used, with the help of specific libraries and tools for the development of smart contracts and connection to the Ethereum blockchain. The main results achieved are: the proof of existence obtained through the voucher / extract generated by the blockchain through a transaction. It contains information about the record, such as the public address of the person who performed the

transaction, the date and time of the record, the hash of the document, and others; the possibility of verifying if a certain document is registered in the blockchain, through its hash, if not found in the blockchain, it is possible to identify possible changes in the original document; the low cost, registration time and the agility of document validation since, because it is located on the Internet, the blockchain can be accessed at any time and anywhere in the world.

Keywords: Ethereum. Blockchain. Smart contracts.

LISTA DE FIGURAS

Figura 1 - Processo de encriptação e decriptação da criptografia simétrica.	27
Figura 2 - Envio de uma mensagem criptografada a uma chave pública.	28
Figura 3 - Procedimentos para executar uma função <i>hash</i>	30
Figura 4 - Processo de certificação.	31
Figura 5 - Processo para gerar a assinatura digital de um documento.	32
Figura 6 - Procedimentos para verificar a assinatura digital de um documento. ...	33
Figura 7 - Processo de criação de um endereço Bitcoin.	37
Figura 8 - Formação do conjunto UTXO.	38
Figura 9 - Funcionamento da blockchain.	44
Figura 10 - Diferença de uma rede centralizada x distribuída (P2P).	47
Figura 11 - Exemplificação de nós da rede P2P.	48
Figura 12 - Encadeamento dos blocos.	50
Figura 13 - Criação do resumo <i>Merkle Root</i>	52
Figura 14 - Verificação de uma transação de um nó SPV.	53
Figura 15 - Número de confirmações no bloco 598859.	54
Figura 16 - Comando de conexão do cliente Geth.	74
Figura 17 – Diagrama de interação das ferramentas.	77

Figura 18 - Comprovante do registro na <i>blockchain</i>	81
Figura 19 - Comprovante do registro na <i>blockchain</i>	82
Figura 20 - Campo Input Data decodificado.....	83
Figura 21 – Verificação de um documento na <i>blockchain</i>	84
Figura 22 - Estatísticas da rede de teste Rinkeby.....	85
Figura 23 - Tempo médio de segundos da mineração na rede principal Ethereum.	86

LISTA DE GRÁFICOS

Gráfico 1 - Evolução na cotação da BTC de julho de 2012 a outubro de 2019.	36
Gráfico 2 - Oferta de Bitcoins ao longo do tempo.....	55

LISTA DE TABELAS

Tabela 1 - Tamanho em bits do resumo/DNA obtido de cada função <i>hash</i>	30
Tabela 2 - Campos que constituem uma conta Ethereum.....	41
Tabela 3 - Estrutura do cabeçalho de um bloco da rede.....	49
Tabela 4 - Comparativo de características entre Bitcoin X Ethereum.	56
Tabela 5 - Comparativo dos consensos PoW e PoS.....	60
Tabela 6 - Comparativo dos trabalhos relacionados.	68

LISTA DE ABREVIATURAS E SIGLAS

ABI	Application Binary Interface
AC	Autoridade Certificadora
AES	Advanced Encryption Standard
API	Application Programming Interface
APT	Advanced Packaging Tool
BTC	Bitcoin
DES	Data Encryption Standard
ETH	Ether
EVM	Ethereum Virtual Machine
GB	Gigabyte
HTTP	Hypertext Transfer Protocol
IBM	International Business Machines Corporation
ICP-Brasil	Infraestrutura de Chaves Públicas - Brasil
IDEIA	International Data Encryption Algorithm
IPC	Inter-process Communication
IPFS	InterPlanetary File System
JSON	JavaScript Object Notation
MB	Megabyte
MD5	Message Digest 5
NPM	Node Package Manager
P2P	Peer-to-peer

RIPEND	RIPE Message Digest
RPC	Remote Procedure Call
RSA	Rivest, Shamir and Adleman
SHA-1	Secure Hash Algorithm – 160 bits
SHA-256	Secure Hash Algorithm – 256 bits
SHA-512	Secure Hash Algorithm – 512 bits
SPV	Simplified Payment Verification
SQL	Structured Query Language
TB	Terabyte
TCP/IP	Transmission Control Protocol
VM	Virtual Machine

SUMÁRIO

1. INTRODUÇÃO	16
1.1 Contextualização.....	18
1.2 Problema de pesquisa.....	20
1.3 Objetivos	21
1.3.1 Objetivo Geral	21
1.3.2 Objetivos Específicos.....	21
1.4 Justificativa.....	22
1.5 Estrutura.....	23
2. REVISÃO TEÓRICA.....	24
2.1 Princípios Básicos	24
2.1.1 Conceito e técnicas.....	24
2.1.2 Criptografia simétrica	25
2.1.3 Criptografia Assimétrica	27
2.1.4 Funções <i>hash</i>	29
2.1.5 Certificado Digital	30
2.1.6 Assinatura Digital	31
2.2 Criptomoedas.....	33

2.2.1	Bitcoin	34
2.2.1.1	Carteira, endereços e criptografia de curvas elíptica	36
2.2.1.2	Transações	37
2.2.2	Ethereum	38
2.2.2.1	Smart Contract.....	39
2.2.2.2	Endereços e Transações	41
2.3	<i>Blockchain</i>	42
2.3.1	Estrutura e Funcionamento.....	42
2.3.2	Tipos de <i>blockchain</i>	44
2.3.3	Rede <i>peer-to-peer</i> (P2P).....	45
2.3.3.1	Nós da rede P2P da <i>blockchain</i>	47
2.3.4	Blocos	49
2.3.4.1	Bloco Gênese	50
2.3.4.2	Árvores de Merkle.....	51
2.3.4.3	Verificação de Pagamento Simplificado – Nós SPV	52
2.3.5	Mineração do protocolo Bitcoin.....	53
2.3.6	Comparativo entre Bitcoin x Ethereum.....	56
2.4	Provas de Consenso	57
2.4.1	Solução.....	57
2.4.2	Problema do gasto duplo	58
2.4.3	Problema dos Generais Bizantinos	58
2.4.4	<i>Proof of Work</i> (PoW)	58
2.4.5	<i>Proof of Stake</i> (PoS)	59
2.5	Interplanetary File System - IPFS.....	61
3.	TRABALHOS RELACIONADOS	62
3.1	Estudo 1	62
3.2	Estudo 2	63

3.3	Estudo 3	64
3.4	Estudo 4	65
3.5	Estudo 5	66
3.6	Comparativo dos trabalhos relacionados	67
4	MATERIAS E MÉTODOS.....	69
4.1	Tecnologias	70
4.1.1	<i>Smart Contract</i>	70
4.1.1	Truffle.....	71
4.1.2	Go Ethereum.....	71
4.1.3	web3.js – Ethereum Javascript API	71
4.1.4	InterPlanetary File System - IPFS.....	71
4.2	Desenvolvimento.....	72
4.2.1	Configurando o ambiente de desenvolvimento	72
4.2.2	Conectando na <i>blockchain</i> de teste da Ethereum.....	72
4.2.3	Desenvolvimento do <i>Smart Contract</i>	74
4.2.4	Desenvolvimento do <i>back-end</i>	76
5	TESTES E ANÁLISES DOS RESULTADOS	78
5.1	Cenário 1 – Contrato de aluguel	78
5.2	Cenário 2 – Propriedade intelectual	79
5.3	Prova de existência	79
5.4	Verificando a autenticidade do documento	83
5.5	Custo e Tempo para certificar um documento	84
6	CONCLUSÕES	87
6.1	Trabalhos futuros	88
	REFERÊNCIAS.....	89

1. INTRODUÇÃO

Em meados dos anos 90 com a emergência de empresas que apostaram no crescimento da Internet e em seu poder comercial, as transações realizadas através de *e-commerce* até hoje, utilizam uma terceira parte, que é representada pelas instituições financeiras. As instituições financeiras são responsáveis por gerenciar o pagamento, fazendo com que um determinado valor seja debitado de sua conta e creditado na conta do comerciante (NAKAMOTO, 2008)¹.

Este modelo de negócio funcionaria sem divergências, se não fosse o simples fato de ser baseado em um modelo de confiança (*trust based model*), onde o comerciante tem total confiança e certeza que, o cartão ou recurso utilizado no pagamento não é roubado ou não está em posse do responsável da compra por vias ilegais (NAKAMOTO, 2008). O modelo baseado em confiança, consiste em criar um modelo de negócio na qual as partes envolvidas no processo sintam-se à vontade para informar quaisquer que sejam os tipos de dados, pessoais ou não, sem criar o sentimento de desconfiança ou que determinada ação pode não ser segura.

Em função desse modelo de confiança, as transações precisam ser reversíveis, uma vez que os bancos não conseguem evitar que um processo jurídico seja aberto, ou uma compra estornada. Grande parcela dessas incertezas poderiam ser evitadas com o uso da moeda física presencialmente, porém, ainda não existe

¹ Satoshi Nakamoto é o pseudônimo do criador da criptomoeda Bitcoin. Até o presente momento, não se sabe a sua verdadeira identidade.

um sistema de pagamento seguro através de um canal de comunicação sem a utilização de uma parte confiável. O modelo anterior de confiança (*trust based model*), pode ser alterado pela prova de criptografia, permitindo que duas partes possam realizar transações livremente, sem uma terceira parte confiável (NAKAMOTO, 2008).

Durante o auge da crise financeira mundial de 2008, Satoshi Nakamoto publicou um *white paper* que propunha um sistema de dinheiro eletrônico e de pagamentos on-line de pessoa a pessoa, sem a necessidade de uma terceira parte ou instituição financeira envolvida, a Bitcoin e até então desconhecida *blockchain* (CARDOSO; PINTO, 2018). Mendanha (2017), define que a *blockchain* é um grande livro razão, responsável por manter um histórico completo de toda as transações, com data e hora. A informação é replicada e distribuída através dos computadores ligados à rede, portanto, não há um ponto central de controle ou falha. Por meio da criptografia e do poder computacional dos participantes da rede, toda transação antes de ser incluída na rede é verificada pelos mesmos participantes com o objetivo de reconhecer se uma informação maliciosa está sendo transferida. Essa verificação é realizada por meio de provas de consenso, permitindo que as transações sejam feitas de forma segura e rápida. Basicamente, a *blockchain* pode ser utilizada para realizar a função da terceira parte confiável, por “debaixo dos panos”.

Conforme Corrêa (2017), transações realizadas através de uma rede *blockchain* são computacionalmente irreversíveis, uma vez executada, não pode ser desfeita, pois, possui o caráter de imutabilidade. Nakamoto (2008), explica que existe um conjunto de tecnologias que se aplicadas em paralelo, são possíveis de contornar o problema de confiança, porém é um processo complexo e não trivial.

A *Blockchain* é um dos principais nomes quando se fala de segurança, aliado à criptografia e provas de consensos, tem-se uma solução quase que perfeita (ARAUJO; SILVA, 2017), a única forma de burlar a segurança seria conseguir poder computacional maior que a rede inteira. A *blockchain* não está limitada apenas ao uso de transações financeiras. Com todo o seu potencial e o auxílio da matemática, uma série de aplicações foram desenvolvidas utilizando-a como solução. Diversas

empresas adaptaram rapidamente suas necessidades garantindo a disponibilidade, integridade e imutabilidade dos dados sem um servidor centralizado ou supervisionado (MENDANHA, 2017).

1.1 Contextualização

A *blockchain* é estruturada a partir de uma arquitetura distribuída e descentralizada. Isso quer dizer que, não há um banco de dados central responsável, uma entidade ou encarregado específico por assegurar o controle das informações registradas na *blockchain* (ALVES *et al.*, 2018).

Ainda para Alves *et al.* (2018), a legitimidade da informação inserida é garantida pelos nós da rede. Em uma rede *peer-to-peer* (P2P), os nós atuam como cliente e servidor para os demais nós da rede. Portanto, não há um único local de controle (descentralização) e é através dos nós que a informação é distribuída. Um nó é qualquer dispositivo eletrônico, incluindo um computador, celular e até mesmo uma impressora, desde que possua conexão com a Internet e tenha um IP válido. O papel do nó é dar suporte a rede, mantendo uma cópia completa da *blockchain* e em alguns casos, é responsável por processar transações e inserir blocos.

Todo registro antes de ser inserido na *blockchain* é assegurado por tipos de nós específicos, o nó completo e o nó minerador. Inicialmente, a transação realizada via *blockchain* é validada pelo nó completo para que posteriormente o nó minerador a insira em um bloco a ser minerado. O bloco é constituído por uma estrutura de dados, com diferentes variáveis. Após ser minerado, os nós completos da rede realizam uma nova validação para checar se as informações do bloco estão de acordo com as demais regras da rede *blockchain* (MENDANHA, 2017).

Para que um bloco seja minerado, é preciso seguir regras estabelecidas por uma prova de consenso de acordo com a *blockchain* aplicada. As mais conhecidas e utilizadas atualmente são a *Proof of Work (PoW)* e *Proof of Stake (PoS)*. Toda informação inserida em um bloco é criptografada. O anonimato do usuário é garantido através de uma chave privada e pública, e a informação registrada na rede por ele é transformada em uma sequência fixa de números criptográficos que pode-se chamar de resumo *hash* (CORRÊA, 2017).

A empresa *OriginalMy*, criada no Brasil em 2015, baseia todos seus produtos em *blockchain*, sendo uma das principais referências deste ramo. Por meio de um programa completamente automatizado e seguro, é possível coletar provas públicas sobre conteúdo online, como redes sociais e sites, certificando-as em *blockchain*. Também são possíveis a verificação de autenticidade de arquivos e certificação digitais, incluindo acordos e contratos. Em contato realizado via e-mail com a empresa, destaca-se que as leis estão sempre a um passo atrás da inovação, portanto, em alguns casos, ainda é preciso efetuar a autenticação em um cartório físico, como por exemplo um registro civil ou de imóveis. Todavia, documentos que precisam de notariação de um cartório podem ser certificados pela *blockchain* para ter uma prova de precedência em casos onde o registro nos órgãos autenticadores ainda não está pronto. Tem-se o exemplo de um registro de patentes e marcas, na qual o indivíduo pode realizar a certificação pela *blockchain* para ter uma prova de existência na utilização da marca, enquanto aguarda o processo burocrático ser finalizado.

Para comprovar a propriedade intelectual e direitos autorais, a empresa PoEx Co. Limited, fundada por Manual Araoz e Esteban Ordano, desenvolveu o site “*Proof of Existence*”. O site permite o *upload* de um arquivo com qualquer que seja sua informação, e em seguida, a função *hash* de criptografia é executada, extraíndo a *hash* do arquivo. Por meio do pagamento de uma taxa, essa *hash* é inserida na *blockchain* e verificada pelos nós dispersos pela rede.

Revoredo (2019) apresenta em seu livro que a república da Geórgia adotou a tecnologia *blockchain* para o registro de terras de seus cidadãos. Por meio da solução desenvolvida, os moradores do país podem verificar a legitimidade de seus documentos sem a necessidade de compartilhar informações confidenciais. O funcionamento do sistema é simples: os usuários inserem seus documentos por meio de uma interface gráfica onde é criada uma *hash*, em seguida, a *hash* criada é enviada a *blockchain* pública da Bitcoin gerando como comprovação um carimbo de data e hora. A verificação de veracidade do documento se dá pela sua *hash*, portanto, qualquer edição ou alteração no documento resulta em outra *hash*, automaticamente o invalidando. Um dos problemas apontados que motivaram a Geórgia a utilizar a solução *blockchain*, foram os anos de corrupção no sistema

imobiliário do país, processos burocráticos e aumento de fraudes que desgastaram a confiança em instituições, gerando problemas econômicos. Assim como a Geórgia, outros países vêm tentando adotar este tipo de registro, como por exemplo, a autoridade máxima da Suécia neste ramo, a Lantmariet. O objetivo é agilizar as transações do setor imobiliário de 3 a 6 meses para aproximadamente 10 dias.

1.2 Problema de pesquisa

Atualmente, para um documento ser válido juridicamente, é preciso que ele seja “notarizado” por algum órgão público. Por exemplo, para um contrato entre duas partes ser válido, é preciso de uma terceira parte de confiança para o validar, neste caso, o cartório é responsável por autenticar o documento com o seu carimbo (NYGAARD, MELING e JEHL, 2019). Para aplicar os princípios básicos de segurança da informação a um documento digital, ou seja, manter a confidencialidade, integridade, disponibilidade e autenticidade, uma pessoa ou empresa deve recorrer as chaves públicas fornecidas pelo órgão nacional ICP-Brasil (Infraestrutura de Chaves Públicas Brasileira). Indiferente do meio escolhido pelo indivíduo, seja o cartório ou ICP-Brasil, para validar um documento haverá burocracia, custando-o tempo e dinheiro.

Por meio do aspecto de imutabilidade proporcionado pela *blockchain*, qualquer pessoa ou empresa poderia comprovar a existência e autenticidade de um arquivo digital. Um “DNA” único é extraído do documento (a *hash*) através de uma função *hash* de criptografia, portanto, caso um ponto ou caractere seja adicionado ou removido do documento, após a execução da função, outro DNA é gerado (MENDANHA, 2017). A prova de existência é consolidada quando o DNA obtido do documento é inserido na rede da *blockchain*. Todo documento inserido e verificado pela rede possui um carimbo de data e hora, tornando-o existente (MARTINS, 2018). Toma-se como exemplo prático um contrato digital, assinado digitalmente por duas partes. Para comprovar que o documento em mãos foi acordado entre as partes e que não houve nenhuma adulteração em seu conteúdo, pode-se utilizar o registro de existência e imutabilidade fornecida pela *blockchain*. Outro exemplo é a posse de propriedade intelectual ou pesquisas científicas. Em casos de plágio,

pode-se comprovar a autoria do documento em questão através do carimbo de tempo mantido pela *blockchain*.

1.3 Objetivos

Esta seção apresenta o objetivo geral e quais os objetivos específicos que este Trabalho de Conclusão de Curso propõe-se a atingir.

1.3.1 Objetivo Geral

O presente trabalho tem como objetivo experimentar um modelo de confiança para o armazenamento do DNA e validação de arquivos digitais através da exploração da tecnologia *blockchain*.

1.3.2 Objetivos Específicos

- Conceitualizar as *blockchain* da Bitcoin e Ethereum, realizando comparativo de suas características;
- Definir e elucidar os itens de acordo com a tecnologia: criptografia e seus tipos, funções *hash*, certificado e assinatura digital, provas de consenso, rede *peer-to-peer*, nós, blocos, *smart contracts*, *frameworks* e programas disponibilizados pela comunidade da Ethereum, utilizados na integração com a *blockchain*;
- Demonstrar cenários reais onde a proposta do trabalho é aplicada, como por exemplo, registrar uma propriedade intelectual de dois ou mais indivíduos; auxiliar na certificação de um contrato entre duas partes;
- Desenvolver uma aplicação *back-end* descentralizada (DApps) que utiliza a *blockchain* e *smart contract* da Ethereum para atestar os cenários apresentados, comprovando a existência e imutabilidade de um arquivo digital através do registro de tempo criado na *blockchain*;
- Apresentar a tecnologia *InterPlanetary File System* (IPFS) para auxiliar na comprovação da existência e imutabilidade de um arquivo digital.

1.4 Justificativa

A *blockchain* foi criada em 2008 juntamente ao Bitcoin, tendo como principal princípio atuar como um modelo de confiança. Mesmo após mais de 10 anos do seu surgimento, grande parte da população mundial não possui conhecimento sobre o que são as criptomoedas e nunca ouviram falar de *blockchain*. Em uma pesquisa realizada pelo banco HSBC em 11 países da Europa, Oceania, Ásia e Estados Unidos, 12.019 mil pessoas de diferentes idades foram entrevistadas e 59% nunca ouviram falar sobre a tecnologia. Entre os que já ouviram falar, 80% não souberam descrever como a tecnologia funciona (HSBC, 2017).

Por meio dos aspectos de segurança fornecido pela *blockchain* e da extração do DNA único de um documento, tem-se diversos cenários a qual esta solução poderia ser aplicada, alguns exemplos são: artistas deixariam de preocuparem-se com direitos autorais, visto que, registrariam suas músicas na rede, garantindo a sua autenticidade. Empresas poderiam evitar fraudes internas em seus dados organizando-os melhor e tornando-os imutáveis, bem como, evitar ataques *hackers*, com a criptografia da informação. Um documento poderia ser validado através da *blockchain*, sem a necessidade de comparecer fisicamente a um cartório. É importante destacar que o ato de registrar um documento na *blockchain* não consiste de fato em armazenar o documento e seu conteúdo, e sim, armazenar a comprovação que o arquivo existiu em determinado momento. Toda informação armazenada na *blockchain* possui caráter imutável, portanto, por meio de seu histórico, qualquer pessoa que possuir uma cópia do documento poderá comprovar sua existência de maneira incontestável através de uma verificação atestando sua autenticidade e imutabilidade.

Cardoso e Pinto (2018) explicam que a Internet impulsionou a forma como os documentos são distribuídos, com isso, a necessidade de se possuir fontes confiáveis para armazenar dados e documentos concebidos entre empresas e pessoas é cada vez maior, além de poderem ser consultados a qualquer momento. Afirmam também que devido à natureza de descentralização, segurança por criptografia e eliminação de uma terceira parte confiável, a *blockchain* pode ser

utilizada para autorizar, autenticar e auditar os mais variados tipos de dados, indiferente de sua origem.

1.5 Estrutura

O trabalho está estruturado em 6 capítulos. O primeiro capítulo introduz a tecnologia *blockchain* e contextualiza sobre as tecnologias e técnicas auxiliares, como rede *peer-to-peer*, criptografia, nós e blocos. Apresenta também qual o problema de pesquisa e a solução obtida por implementar tais tecnologias. O segundo capítulo consiste em apresentar o referencial teórico que contribui no entendimento da proposta. Tem como principais tópicos a explanação sobre o conceito de criptografia e seus tipos, certificado e assinatura digital, uma breve descrição sobre o que é a criptomoeda Bitcoin e Ethereum, estrutura e funcionamento de uma *blockchain*, mineração de um bloco, provas de consenso e a tecnologia IPFS. O terceiro capítulo detalha cinco trabalhos relacionados que auxiliaram no desenvolvimento do capítulo um e dois. O quarto capítulo apresenta as metodologias utilizadas que ajudaram na organização e na evolução da monografia como um todo. O quinto capítulo demonstra os testes e análises dos resultados obtidos por meio do desenvolvimento e objetivos do trabalho. A conclusão e trabalhos futuros são descritos no capítulo 6.

2. REVISÃO TEÓRICA

Este capítulo tem como finalidade elucidar os temas propostas nos objetivos gerais e específicos, situando os leitores, podendo servir de base para outros trabalhos acadêmicos. São abordados os conceitos e técnicas de criptografia, funções *hash*, certificado e assinatura digital, Bitcoin, Ethereum, *blockchain* e sua estrutura, mineração, provas de consensos e a tecnologia IPFS.

2.1 Princípios Básicos

Esta seção explanará sobre os conceito e técnicas de encriptação da informação, funções *hash*, certificado e assinatura digital, peças fundamentais no funcionamento da *blockchain*.

2.1.1 Conceito e técnicas

Segundo Fraulkner (2016), a encriptação é a arte e ciência de proteger a informação, transformando-a em um texto impossível de ser compreendido, chamado de texto cifrado. Apenas aqueles que possuírem a chave secreta conseguem descriptografar a informação.

Para Corrêa (2017), a criptografia deve ser um componente básico de qualquer aplicação conectada à Internet. É o aspecto de maior relevância para atestar a segurança da informação. Além de ser usada como base para outros protocolos e tecnologias, é composta pelos princípios de confidencialidade, integridade e não repúdio.

Segundo Ferguson, Schneier e Kohno (2015), é utilizada como forma de autenticação para a assinatura e certificação digital; garante o anonimato do autor e da informação; é de interesse em assuntos econômicos, político e física quântica. Qualquer tipo de informação pode ser criptografada por meio de cifras ou de códigos. A informação é cifrada por meio de substituição ou transposição dos caracteres originais. As cifras de substituição trocam os caracteres originais por outros, como por exemplo, a letra “A” por ser substituído por “^”. A cifra de transposição mistura os caracteres originais, exemplo, “BLOCKCHAIN” > “BNLIOACHKC” (MORENO; PEREIRA; CHIARAMONTE, 2005).

Para Moreno, Pereira e Chiaramonte (2005), os algoritmos criptográficos podem ser classificados em algoritmo de bloco e de fluxo. A cifra de bloco atua sobre um conjunto de dados. Um exemplo de bloco, poderia ser, antes de ser cifrado, o texto ser dividido em blocos de 8 a 16 *bytes* que serão cifrados ou decifrados. Caso o texto cifrado não complete o número mínimo de *bytes*, o restante é preenchido normalmente com zeros “0” até ter-se o número mínimo exigido pelo algoritmo em questão. Assim, se em um algoritmo o texto cifrado gerou 8 *bytes*, sendo necessário 16, 7 *bytes* serão preenchidos com o valor 0 e o 7, acrescentado ao final.

Na cifra de fluxo, a mensagem é cifrada *bit a bit*, em um fluxo constante. É conhecida também como criptografia de *stream* de dados, onde o operador XOR é utilizado entre o *bit* do dado e o *bit* obtido pela chave. Todo texto criptografado possui uma chave secreta. É por meio dessa chave que é possível obter o texto original (MORENO; PEREIRA; CHIARAMONTE, 2005).

2.1.2 Criptografia simétrica

De acordo com Fraulkner (2016), a criptografia simétrica é um algoritmo de encriptação de “dois caminhos” que utiliza a mesma chave para encriptar e descriptografar uma mensagem. Na teoria, tanto o emissor quanto o destinatário precisam conhecer a chave secreta, porém na prática sabe-se que é difícil compartilhar e manter a mesma chave em segurança.

Para Oliveira (2012), é o exemplo de criptografia mais antigo da ciência da computação, na qual ambas as partes devem manter uma chave em segredo. Essa chave pode ser chamada de senha e é usada para o emissor codificar a mensagem. O destinatário, utiliza a mesma senha para descriptografar a mensagem. A criptografia simétrica não garante os aspectos de não repúdio e autenticidade, diferente da criptografia assimétrica, detalhada na seção 2.1.4.

A lógica da criptografia simétrica retrata o seguinte: um emissor, chamado de A, criptografa um texto utilizando a chave secreta e um algoritmo específico. O receptor B utiliza o mesmo algoritmo e chave para descriptografar o texto com o objetivo de encontrar seu conteúdo original. O uso da chave privada em conjunto a encriptação faz-se necessário, pois, caso um intruso C saiba qual foi o algoritmo utilizado para encriptar o texto, a mensagem poderá ser manipulada. Conclui-se que, a segurança desse modelo de criptografia está na chave utilizada e não no algoritmo, desde que, os indivíduos A e B mantenham a chave em segredo. Toda mensagem enviada que utiliza essa técnica de criptografia é transmitida através de um canal seguro, que por sua vez, possui uma determinada chave secreta VOITECHEN (2015).

Segundo Oliveira (2012), a grande vantagem desta técnica é a simplicidade, uma vez que, é de fácil uso e os processos criptográficos são executados rapidamente. Destaca que quanto mais simples o algoritmo de encriptação e decriptação, maior é a facilidade de implementação e velocidade no processamento. Os principais tipos de criptografias simétricas são: AES e IDEIA, ambas com uma chave secreta de 128 *bits* e DES, com uma chave secreta de 56 *bits*.

Na Figura 1, o Emissor enviou uma mensagem encriptando-a com um algoritmo específico e com a sua chave secreta. O Receptor recebeu a mensagem e realizou o mesmo procedimento para obter o conteúdo original da mensagem. Um intruso somente conseguirá descriptografar a mensagem caso possuir a mesma chave secreta de Maria e João, estabelecida pelo canal seguro.

Figura 1 - Processo de encriptação e decriptação da criptografia simétrica.



Fonte: Evaltec (2019).

2.1.3 Criptografia Assimétrica

A criptografia assimétrica, também conhecida como criptografia de chaves públicas, permite que dois indivíduos troquem mensagens livremente entre si por meio de um par de chaves (CORRÊA, 2017).

Neste conjunto de chaves privada e pública, a chave pública é gerada através da chave privada, portanto, fazer o caminho inverso e tentar alcançar a chave privada através da chave pública, é computacionalmente inviável. O processo para obter a chave pública é baseado em problemas matemáticos e atualmente não existe uma solução eficaz, vide o impasse do logaritmo discreto (MARTINS, 2018).

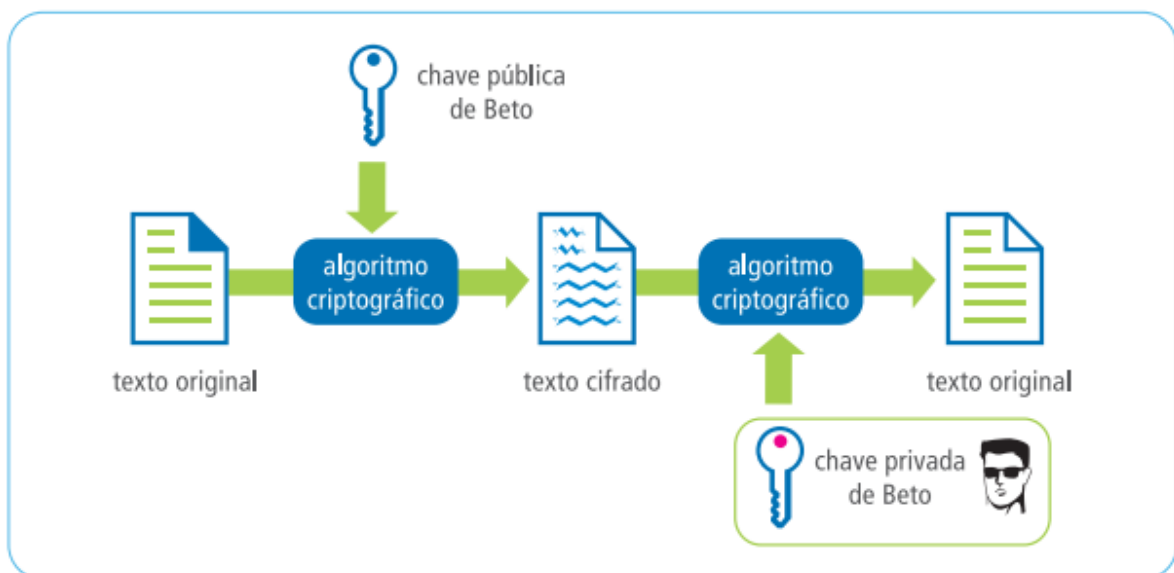
A chave privada deve ficar em posse de cada indivíduo, em segredo. Já a chave pública, deve ser disponibilizada na Internet. É através da chave pública que o indivíduo recebe a mensagem, e a partir de sua chave privada, a mensagem é descriptografada. Portanto, qualquer pessoa que possuir a chave privada de uma chave pública será capaz de ler a mensagem enviada ao endereço público (OLIVEIRA, 2012).

De acordo com Oliveira (2012), a grande vantagem em utilizar este tipo de criptografia é que qualquer pessoa pode enviar uma mensagem a um endereço público. A posse da chave privada é de responsabilidade do indivíduo, portanto,

enquanto a chave privada estiver em segurança, a confidencialidade da mensagem é assegurada.

Na Figura 2, o texto da mensagem é cifrado com a chave pública de Beto. Na sequência, Beto utiliza sua chave privada para descriptografar a mensagem. É importante lembrar que se outra entidade possuir a chave privada de Beto, a mensagem enviada também poderá ser descriptografada.

Figura 2 - Envio de uma mensagem criptografada a uma chave pública.



Fonte: IMED (2017).

Atualmente, um dos algoritmos largamente utilizado e conhecido para a criptografia de chave pública é o RSA. A segurança desse algoritmo é baseada na fatoração de dois números grandes, onde dois números primos são multiplicados para obter-se um terceiro número, na qual, é muito difícil obter os dois números a partir do terceiro número gerado (OLIVEIRA, 2012).

Segundo Cavalcante (2005), para fatorar um número de 200 dígitos, em 2005, seria necessário que um computador ficasse processando por 4 milhões de anos. Já para fatorar um número de 500 dígitos, seria preciso 10^{25} anos. Portanto, uma chave pública é gerada por meio da multiplicação de dois primos grandes aliada de informações complementares. A derivação da chave privada é efetuada através da chave pública que envolve fatorar um número grande. Logo, se um

número grande for bem escolhido, ninguém poderá descobrir qual a chave privada do indivíduo, pois, a segurança do RSA está na complexidade de fatoração de números grandes.

García (2019), explica que o algoritmo RSA é distinguido em três partes. A primeira consiste na geração das chaves privada e pública dos usuários participantes de um canal de comunicação seguro. A segunda parte é composta pela cifragem da mensagem, na qual por exemplo, para mandar uma mensagem encriptada para o usuário B, o usuário A deve utilizar a chave pública do usuário B. A terceira e última parte, representa a decifração da mensagem, onde o usuário B utiliza sua chave privada para decifrar mensagens recebidas pela sua chave pública.

2.1.4 Funções *hash*

Do inglês, *hash* ou *hashing*, é o ato de transformar um texto em uma sequência fixa de caracteres que representa o texto original. O processo de *hash* consistem em duas etapas. A primeira etapa é chamada de entrada e consiste em informar um texto na função *hash*, indiferente do tamanho ou quantidade de caracteres; a segunda etapa é chamada de resumo e é responsável pela criação da sequência fixa de caracteres de acordo com o texto original (SINGH e GARG, 2009).

Uma das principais características da função *hash* é, por exemplo, que um texto igual possui sempre o mesmo resumo/sequência de caracteres. Dado qualquer alteração no texto da entrada, ao executar novamente a função *hash*, o resumo obtido é totalmente diferente (CORRÊA, 2017). O resumo gerado através da função *hash* pode ser chamado de DNA, já que é único e é através desse aspecto que este trabalho propõe-se a comprovar a integridade de um arquivo digital.

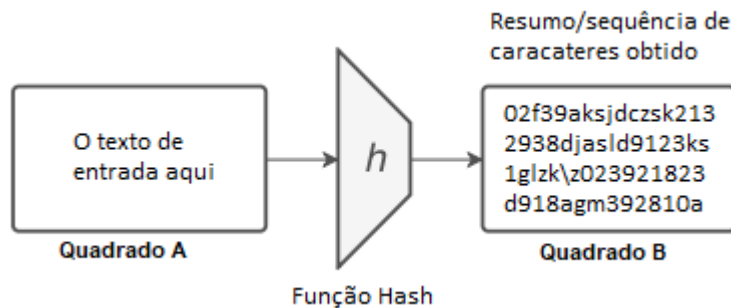
As funções *hash* mais comumente utilizadas comercialmente e na comunidade acadêmica são MD4, MD5 e a SHA-1, tendo como derivações a SHA-256 e SHA-512. As mais recomendadas para situações críticas são SHA-256 e SHA-512 (MORENO; PEREIRA; CHIARAMONTE, 2005). A Tabela 1 lista o tamanho da saída obtida em *bits* de cada função *hash*.

Tabela 1 - Tamanho em bits do resumo/DNA obtido de cada função *hash*.

Função <i>hash</i>	Saída/DNA gerado em <i>bits</i>
MD4	128 <i>bits</i>
MD5	128 <i>bits</i>
SHA-1	160 <i>bits</i>
SHA-256	256 <i>bits</i>
SHA-512	512 <i>bits</i>

Fonte: Elaborado pelo autor (2019).

Na Figura 3 é ilustrado o processo de execução de uma função *hash* em um texto de entrada. O texto do Quadrado A gerou o resumo/sequência de caracteres do Quadrado B.

Figura 3 - Procedimentos para executar uma função *hash*.

Fonte: Adaptada com base em Martins (2018, p. 14).

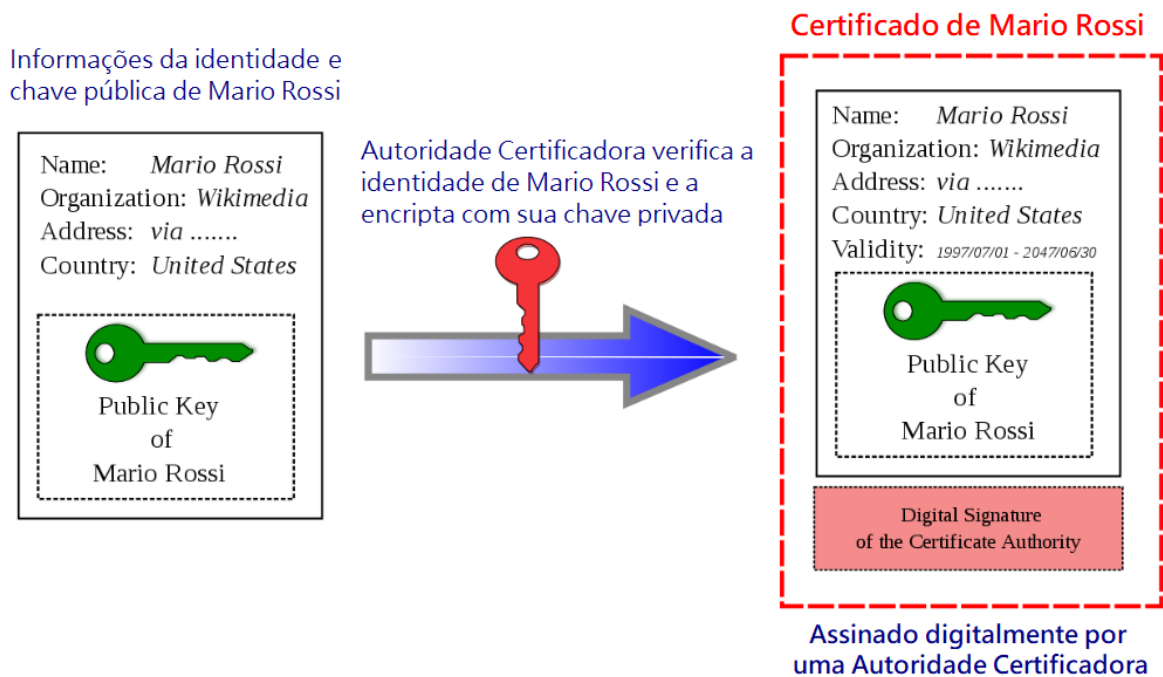
2.1.5 Certificado Digital

De acordo com Davis (2006), um certificado digital é um pedaço de código responsável por identificar um indivíduo na Internet, criando uma espécie de “*identidade digital*”.

Uma Autoridade Certificadora (AC) é responsável por atribuir uma chave pública a uma entidade ou pessoa, gerando um documento eletrônico com dados do titular e do órgão certificador emissor. A criptografia utilizada em certificados digitais é a criptografia assimétrica. Logo, um certificado digital contém os dados da AC emissora, o período de validade, política de utilização, nome e chave pública associada ao solicitante. No Brasil, o órgão responsável por gerenciar as chaves públicas é o ICP-Brasil (CORRÊA, 2017).

Conforme a Figura 4, o indivíduo Mario Rossi solicitou à autoridade certificadora um certificado digital. Uma chave pública foi concebida a “Mario Rossi” e a AC o certificou com uma data de validade.

Figura 4 - Processo de certificação.



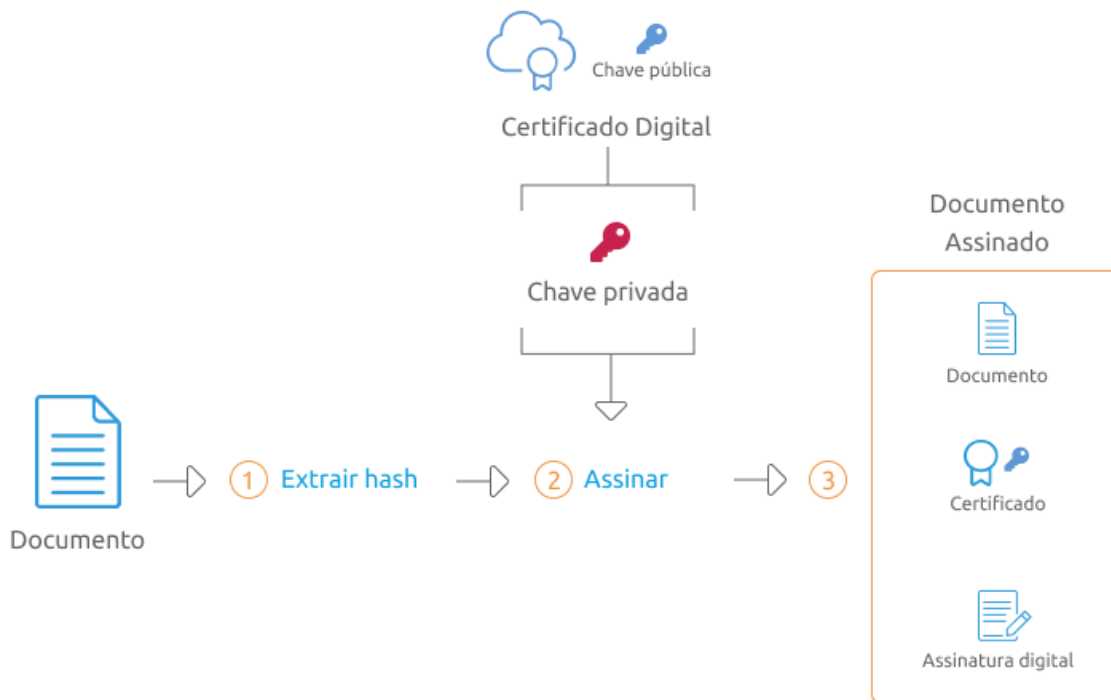
Fonte: Adaptada com base em WikiWand (2019).

2.1.6 Assinatura Digital

Uma assinatura digital possui o mesmo propósito de uma assinatura física. Tem como objetivo assegurar que um documento ou mensagem foi de fato enviada pelo emissor, garantindo a integridade, autenticidade e principalmente a irretratabilidade do objeto em questão (MACCORMICK, 2012).

Para assinar digitalmente um documento, o indivíduo deve gerar a *hash* da mensagem através de uma função de *hash*. A chave privada do indivíduo é utilizada para encriptar a *hash* obtida do documento, gerando assim a assinatura digital. A assinatura é anexada ao documento encaminhado, e a mensagem enviada às partes interessadas (MARTINS, 2018). A figura 5 exemplifica este processo.

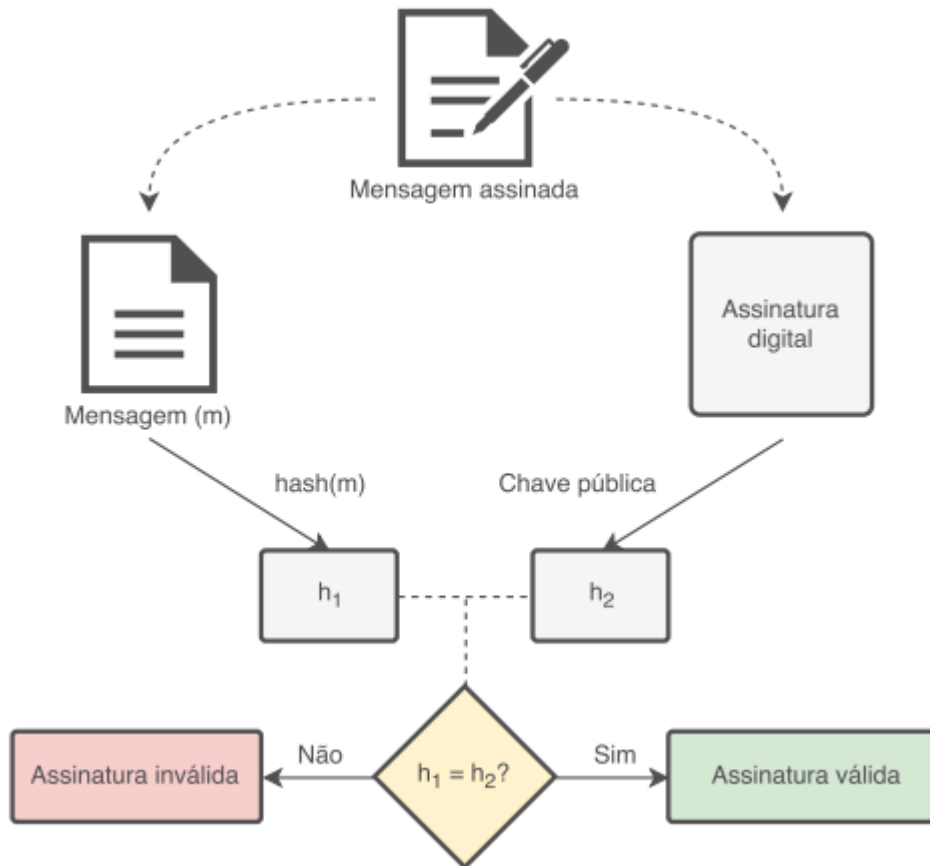
Figura 5 - Processo para gerar a assinatura digital de um documento.



Fonte: Bry (2019).

Conforme a Figura 6, as partes interessadas do documento podem verificar sua autenticidade encontrando a *hash* da mensagem, o *h1*. Para descryptografar a assinatura digital, é utilizada a chave pública do emissor da mensagem, o *h2*. Caso a *hash* do *h1* for igual a *hash* do *h2*, a mensagem é de fato do emissor e não houve nenhuma ação fraudulenta (MARTINS, 2018).

Figura 6 - Procedimentos para verificar a assinatura digital de um documento.



Fonte: Martins (2018, p. 18).

2.2 Criptomoedas

As criptomoedas são ativos digitais que operam sob uma rede completamente descentralizada, independente de uma terceira parte confiável. Para verificar e garantir a segurança das transações, uma das principais técnicas utilizada é a criptografia (MARTINS e VAL, 2016). Atualmente, há centenas de tipos de criptomoedas espalhadas pelo mundo, porém, esta seção contextualiza sobre as criptomoedas Bitcoin e Ethereum, que segundo CoinMarketCap (2020), são as criptomoedas de maior dominância até o momento.

2.2.1 Bitcoin

Bitcoin é uma moeda digital que utiliza um protocolo de comunicação descentralizado *open source*², que permite realizar transações financeiras. Vem crescendo lentamente desde o seu lançamento, em 2009, logo após a crise econômica mundial de 2007-2008 e tem como principal objetivo ser um livro razão de transferências. Ainda não se sabe quem é o inventor da criptomoeda. Várias pessoas tentaram se passar pelo criador, porém, nenhuma foi reconhecida oficialmente pela comunidade. O criador atende pelo pseudônimo de Satoshi Nakamoto (ZOHAR, 2018).

Segundo Kaushal, Bagga e Sobti (2017), a Bitcoin substitui uma *fiat money*³ em diversos aspectos, pois, ela pode ser gerenciada e transferida por qualquer pessoa, indiferente do país onde está, não há limites. Algumas transações podem possuir pequenas taxas e atualmente não é necessária nenhuma informação pessoal, garantindo o anonimato. As informações das transações são criptografadas, fornecendo total segurança. Apesar de ser considerado uma nova moeda, a criptomoeda vem enfrentando dois grandes desafios, a volatilidade em seu preço e o grau de aceitação. Espera-se que a volatilidade diminua conforme novas pessoas forem juntando-se a comunidade.

No Brasil, a Receita Federal instituiu obrigações para a prestação de informações referentes a movimentações de criptomoedas. A normativa Nº 1.888 de 3 de maio de 2019 prevê que qualquer pessoa que movimentar mais de 30 mil reais em criptomoedas deve declarar os valores a receita. As corretoras online nacionais, conhecidas também como *exchanges* devem fazer a prestação para seus clientes, portanto, se a movimentação for realizada por meio delas, a pessoa não precisa declarar por si. Se a movimentação for realizada em *exchanges* de fora

² Open source é um termo em inglês, que significa código aberto. Toda aplicação open source possui o seu código disponível livremente pela Internet.

³ Todo o dinheiro que possui uma entidade regulamentadora pode ser chamado de *fiat money*. Por exemplo, no Brasil e Estados Unidos, quem controla o Real e o Dólar é o respectivo governo.

do país, é o indivíduo que tem a obrigação de fazer a declaração (DIÁRIO OFICIAL DA UNIÃO, 2019).

De acordo com Corrêa (2017), a transação é realizada por meio de carteiras eletrônicas. Dois indivíduos podem trocar criptomoedas livremente entre si com auxílio de suas chaves privadas e públicas.

A Bitcoin é o primeiro protocolo criado totalmente descentralizado e fora do controle de uma instituição central ou de governo. Quem controla ou regula a criptomoeda são os nós espalhados pela rede *peer-to-peer* P2P, (detalhado na seção 2.3.3). Utiliza a prova de consenso *Proof of Work* (PoW) (apresentada na seção 2.4.4) para garantir que a falha *Byzantine* (seção 2.4.3) e o gasto duplo (seção 2.4.2) não ocorra (ZOHAR, 2018). Está associado a capacidade de escalabilidade; incentivos financeiros (recompensa os mineradores e responsáveis por concluir as transações, são conhecidos como nós mineradores e *full nodes* (conceitos aprofundados na seção 2.3.3.1); garante o anonimato dos participantes e a transferência da informação é feita de forma segura através da rede P2P (ZOHAR, 2018).

O sucesso do protocolo Bitcoin foi tanto que, diversas empresas inovaram seus produtos baseando-se nesse protocolo e na tecnologia *blockchain*. Através do seu uso, é possível criar sistemas de votação; armazenamento distribuído em nuvem; acompanhamento de uma cadeia de alimento, desde o plantio da matéria prima, até a exposição no mercado; contratos inteligentes; entre outros (MARTINS, 2018).

O Gráfico 1 ilustra a evolução da cotação da BTC nos últimos 8 anos. Nota-se que em 2013 o preço era aproximadamente \$ 120,00. Em abril de 2017, o preço saiu da zona dos \$ 1.000,00 e em dezembro do mesmo ano chegou quase nos \$ 20.000,00. Em 17 de outubro de 2019, é cotado em aproximadamente \$ 8.100,00.

Gráfico 1 - Evolução na cotação da BTC de julho de 2012 a outubro de 2019.



Fonte: TradingView (2019).

2.2.1.1 Carteira, endereços e criptografia de curvas elíptica

As Bitcoins de um indivíduo são gerenciadas através de uma chave privada e pública, formando o que chama-se de carteira (*wallet*). As chaves são as “senhas” da carteira, portanto, quem possuir a chave privada da carteira poderá transferir Bitcoins livremente. Já a chave pública é utilizada para receber Bitcoins (MARTINS, 2018).

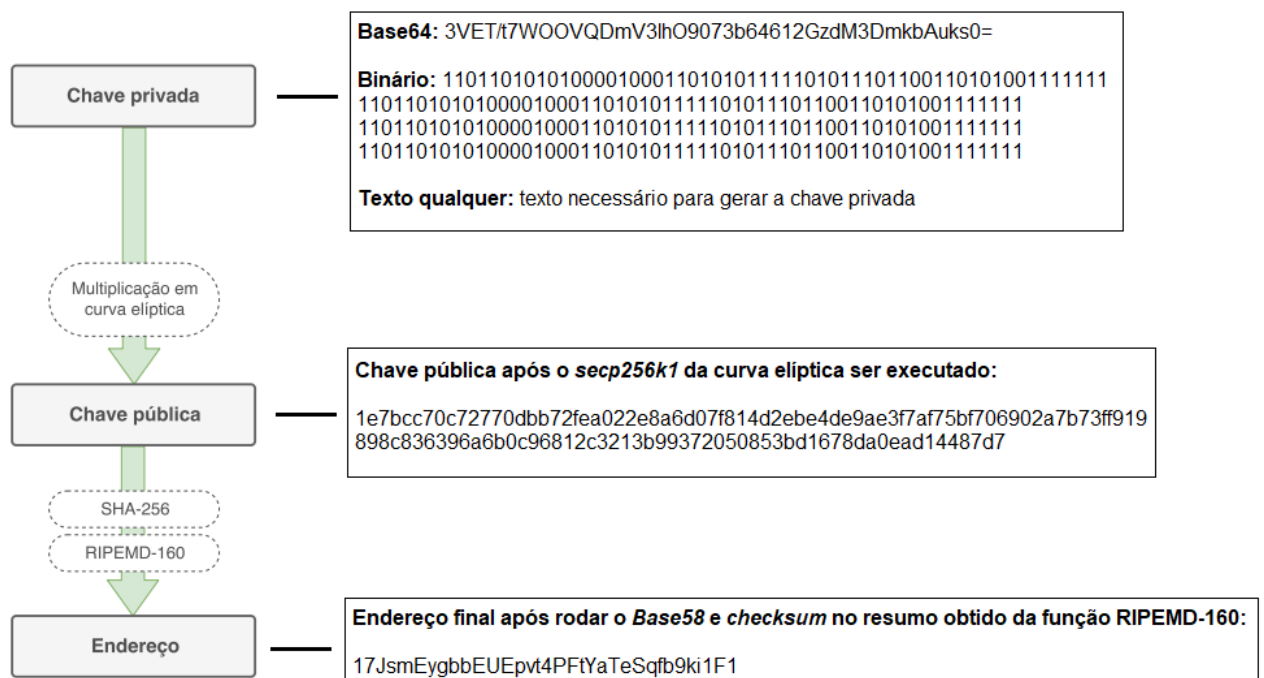
A criação de um endereço de Bitcoin dá-se a partir da sua chave privada que é composta por uma sequência de 256 *bits*. Por exemplo, pode ser um texto de 256 um ou zeros ($32 * 8 = 256$), um texto binário, um texto de *Base64*, um texto qualquer, entre outros. A chave pública é gerada através da chave privada e utiliza o padrão *secp256k1*. *Secp256k1* é um padrão de curva do algoritmo de criptografia de curvas elíptica. (GARCÍA, 2019).

Para ter-se o endereço final utilizado como chave pública, alguns procedimentos são necessários. Primeiro a chave pública obtida através da chave privada é inserida e executada como entrada em 2 funções *hash*'s. A primeira é a função *hash SHA-256* que gera uma sequência de 32 *bytes* e a segunda é a função *hash RIPEMD-160*, obtendo um endereço de 20 *bytes* em cima do conteúdo de 32 *bytes*. Como procedimento final, o endereço de 20 *bytes* é codificado através do

“*Base58Check*”, que é subconjunto do *Base64* e um *checksum* adicional, fornecendo proteção a erros de digitação (MARTINS, 2018). O algoritmo de codificação *Base58*, derivado do subconjunto *Base64* e que compõem o *Base58Check*, elimina os caracteres “+”, “/”, “l” (letra “i” maiúscula), “O” (letra “o” maiúscula), “1” (letra “l” minúscula), e “0” (numero “0”), pois, possuem similaridade entre si (ALFAIFI, 2017).

A Figura 7 exemplifica esse processo. Por meio de uma chave privada qualquer, a chave pública é obtida executando a multiplicação da curva elíptica. O resultado parcial então é codificado com os algoritmos SHA-256, *RIPEMD-160* e *Base58Check*. O resultado é a chave pública, que pode ser disponibilizada pela Internet para o indivíduo receber transações.

Figura 7 - Processo de criação de um endereço Bitcoin.



Fonte: Adaptada com base em Martins (2018, p. 23).

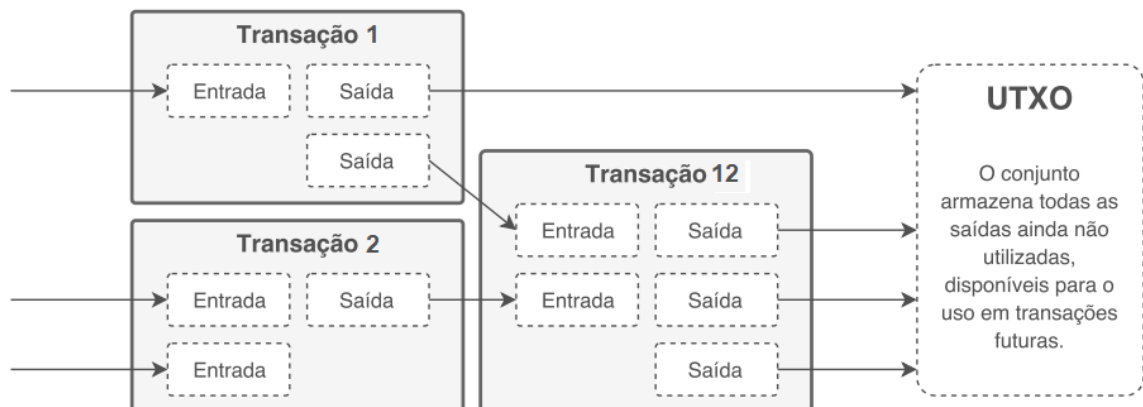
2.2.1.2 Transações

Na Bitcoin, cada transação possui uma ou mais entradas, referente a origem dos fundos recebidos, e uma ou mais saídas, que especifica o destino que o valor será transferido. Novas transações são armazenadas em um conjunto chamado de

UTXO, que significada “saída de transações não gastas” ou do inglês, “*unspent transaction outputs*”. O saldo de um indivíduo é somente visualizado na camada de aplicação, buscando facilitar a utilização do sistema. Na realidade, os fundos são salvos no conjunto UTXO, sendo que, a chave privada de cada usuário é a “senha” para desbloquear o saldo de determinada transação não gasta, habilitando a transferência das moedas disponíveis. Assim, o saldo de uma carteira Bitcoin é definido pelo total de transações não gastas existentes no conjunto UTXO que respectivo endereço possui para gastar (MARTINS, 2018).

A Figura 8 exemplifica a formação do conjunto UTXO. Para compor as entradas da Transação 12, a saída da Transação 1 e Transação 2 são utilizadas. O valor restante da Transação 1 e os fundos adicionados a transação 12 podem ser utilizados para a entrada de outras transações futuras.

Figura 8 - Formação do conjunto UTXO.



Fonte: Adaptada com base em Martins (2018, p. 24).

2.2.2 Ethereum

Atualmente, a Ethereum é a segunda maior rede *blockchain*, ficando atrás da rede Bitcoin. É uma plataforma de código aberto, portanto, qualquer indivíduo pode interagir com suas funcionalidades e recursos. O destaque da Ethereum em comparação ao Bitcoin é que nesta rede é possível executar a tecnologia de contratos inteligentes (*smart contracts*) e aplicações descentralizadas (Dapps). O uso de uma terceira parte confiável ou a probabilidade de fraudes em documentos é diminuída quando utiliza-se estes recursos. Pelos aspectos de segurança

fornecidos pela *blockchain*, as regras e condições estabelecidas em um *smart contract* acabam sendo imutáveis e a troca de informações entre os usuários é confiável (CARDOSO; PINTO, 2018).

Corrêa (2019) explica que a Ethereum foi proposta inicialmente em 2013, tendo seu lançamento oficial em 2015, após testes que recompensaram desenvolvedores em até 25 mil ETH (abreviação para *ethers*, moeda utilizada nesta rede). De acordo com França (2018), a *blockchain* da Ethereum possui algumas características inovadoras em relação ao Bitcoin, como a implementação de *smart contracts* e sua execução, realizada pelo o que chamado de *Ethereum Virtual Machine* (EVM). A tecnologia EVM foi a responsável pela consolidação da Ethereum no mercado, pois, ela atua como um ambiente completo de execução para os *smart contracts*. A EVM tem como base o conceito de *Turing-Complete*, onde é possível executar a maioria das instruções padrões de linguagens de programações, como por exemplo, laços de repetição e condições *if/else*. França ainda destaca que a *blockchain* da Bitcoin não permite a execução de laços de repetição. Corrêa (2017) esclarece que o consenso utilizado atualmente na Ethereum é o *Proof of Work* (PoW), no entanto, há uma versão em testes a qual trabalhará com a arquitetura do consenso *Proof of Stake* (PoS).

De acordo com o guia oficial para desenvolvedores do site da Ethereum (2020), além de possuir uma rede principal (Mainnet), a comunidade da Ethereum mantém algumas redes de testes (Testnet), como por exemplo, a Ropsten, Rinkeby e Goerli, que são utilizadas para testar os *smart contracts* e aplicações sobre determinadas condições antes de serem enviadas de fato para a rede principal.

2.2.2.1 Smart Contract

Para França (2018), um *smart contract* é um tipo de sistema que transfere ativos digitais de acordo com regras definidas. Deste modo, é possível facilitar uma negociação entre duas ou mais partes sem se preocupar-se todas as partes envolvidas irão cumprir com o combinado, portanto, não há necessidade de possuir interferência de uma terceira parte, como por exemplo, um cartório. Toma-se como exemplo a seguinte hipótese: Alice pode sacar até X unidades de uma moeda por

dia, Bob tem disponível para sacar até Y por dia, Alice e Bob podem sacar juntos qualquer valor e Alice pode em algum momento remover a possibilidade de Bob sacar.

Conforme Cardoso e Pinto (2018), o *smart contract* é mais uma inovação permitida pela *blockchain*. São contratos escritos em código puro auto executáveis, ou seja, não requerem obrigatoriamente de intervenção humana para iniciar ou finalizar sua execução. Por meio de um *smart contract*, é possível definir regras de um determinado negócio ou condições contratuais, deixando explícito por meio de códigos se X ocorrer, Y deve ser executado. Por ser um objeto autônomo, o mesmo não pode ser alterado ou excluído após estar na *blockchain*, pois, o *smart contract* sempre será executado de acordo com as regras estabelecidas na *blockchain*.

Para que estas regras sejam executadas automaticamente, é preciso que o respectivo código seja enviado a *blockchain*, onde o contrato é executado conforme as suas condições forem atendidas, garantidas que nunca serão alteradas em função dos aspectos de segurança e imutabilidade da *blockchain*. Os *smart contracts* do protocolo Ethereum permitem que desenvolvedores criem os mais diversos tipos de aplicações descentralizadas (DApp's), desde a monetização de jogos a aplicações de pagamentos, como por exemplo a remuneração de direitos autorais a artistas (FRANÇA, 2018). Corrêa (2019), explica que os *smart contracts* são processados na máquina virtual da Ethereum, a EVM, onde geralmente são escritos na linguagem Solidity (criada especialmente para este tipo de desenvolvimento) e compilados para o *bytecode* da EVM.

Segundo Corrêa (2019), é preciso pagar pela execução de um *smart contract*. Este pagamento é feito através do que é chamado de *gas* (convertido para *ether*, moeda da Ethereum) onde todo contrato para ser executado possui um valor estipulado de *gas*. O *gas* é estipulado de acordo com a complexidade do contrato, portanto, cada regra ou condição existente no contrato ocasiona em mais *gas* utilizado no processamento realizado por nós específicos da rede. O combustível (*gas*) é uma maneira de recompensar os nós responsáveis pelas execuções dos contratos inteligentes. Como forma de segurança, todo contrato possui um limite de *gas* a ser utilizado, com isto, um contrato nunca entrará em um loop de execução,

mesmo que o limite de combustível seja alto, pois, as *ethers* (convertidas posteriormente em *gas*) do endereço que está realizando a transação são finitas.

De acordo com a documentação oficial da Solidity, versão v0.4.21 (2020), pode-se excluir um contrato caso seja necessário, no entanto, é preciso prever e incluir uma função específica no código. Para que o contrato não fique exposto a atividades maliciosas, condições podem ser incluídas para executar a função de autodestruição, como por exemplo, informar uma senha ou uma informação específica.

2.2.2.2 Endereços e Transações

Criador da Ethereum, Buterin (2015), descreve em seu *white paper* que o estado da rede é composto por objetos chamados de contas ou endereços. Uma conta possui um endereço de 20 *bytes* e estados de “transições”, que são os dados de transferências entre contas. Existem dois tipos de contas: as externas, controladas por uma chave privada e as contas de contratos (*smart contract*), controlada pelo código do contrato. As contas externas não possuem código e são usadas para enviar e assinar transações. Contas do tipo contrato executam seu código sempre que recebem uma mensagem ou transação, permitindo a leitura e escrita em seu *storage* (armazenamento) de acordo com o código executado. Diferente da Bitcoin, o saldo de Ethereum é armazenado diretamente na conta, seja ela externa ou de contrato. A Tabela 2 elenca os campos que compõem uma conta.

Tabela 2 - Campos que constituem uma conta Ethereum.

Campo	Definição
<i>Nonce</i>	Um contador numérico usado para garantir que cada transação que o endereço cria e assina é processada somente uma única vez.
<i>Ether balance</i>	O saldo de <i>ethers</i> da conta.
<i>Contract code</i>	O código do contrato, quando há.
<i>Storage</i>	Armazena dados no geral, por padrão vem vazio.

Fonte: Butering (2015).

Ether é o combustível principal na Ethereum, sendo sempre utilizado no pagamento de taxas das transações. Nesta rede, o termo transação refere-se à

criação e assinatura de um conjunto de dados enviado de um endereço a outro. Para uma transação ser realizada, ela deve conter: o endereço do destinatário; a assinatura do remetente, o total de *ethers* enviado (opcional quando utilizado *smart contracts*); um valor para o campo STARTGAS, que representa o máximo de etapas computacionais que uma transação é permitida a fazer; e um valor para o campo GASPRICE que determina a taxa paga pelo remetente para cada etapa executada. Estes últimos dois campos são cruciais para o funcionamento da Ethereum, pois, servem como uma medida de proteção para evitar repetições infinitas ou desperdício de código. Caso a transação não for concluída, seja por problemas como por exemplo: o remetente não possui saldo suficiente ou a execução de um contrato excedeu o limite de *gas* estipulado, todo o estado de transição (informações da transação) são revertidas, exceto as taxas que são enviadas aos mineradores responsáveis pela execução da transação e/ou do contrato. O *gas* é a unidade de medida criada pela rede para controlar a execução de transações e contratos. Todo *gas* consumido é convertido em *ethers*, gerando assim as taxas da transação. Logo, para um ataque hacker ser bem-sucedido, além de precisar passar por todos os aspectos de segurança fornecidos pela estrutura da *blockchain*, é preciso que o *gas* proporcional ao código do ataque seja pago pelo atacante (BUTERIN, 2015).

2.3 Blockchain

A *blockchain* é um grande livro razão público de informações coletadas por meio de uma rede situada “acima” da Internet, “onde os olhos não veem”. Seu grande potencial inovador é resultado de como as informações são armazenadas (REVOREDO, 2019). Nessa seção serão apresentados o conceito de *blockchain* e como é estruturada e gerenciada pelos diferentes tipos de nós de uma rede P2P. Abordará também o que são blocos, quais suas funções, bloco gênese e mineração.

2.3.1 Estrutura e Funcionamento

Blockchain revolucionou a maneira de como dados são armazenados na Internet. A informação armazenada pode possuir qualquer origem: uma transação financeira; documentos pessoais; um acordo entre duas partes, entre outros. Para isso ocorrer, é necessário a confirmação de diversos dispositivos espalhados pela

rede, como computadores ou celulares. Uma vez que todos concordem com o dado que está sendo inserido, conhecido como consenso, a informação não poderá ser contestada, alterada ou removida sem a permissão daqueles que concordaram em inserir a mesma (MENDANHA, 2017).

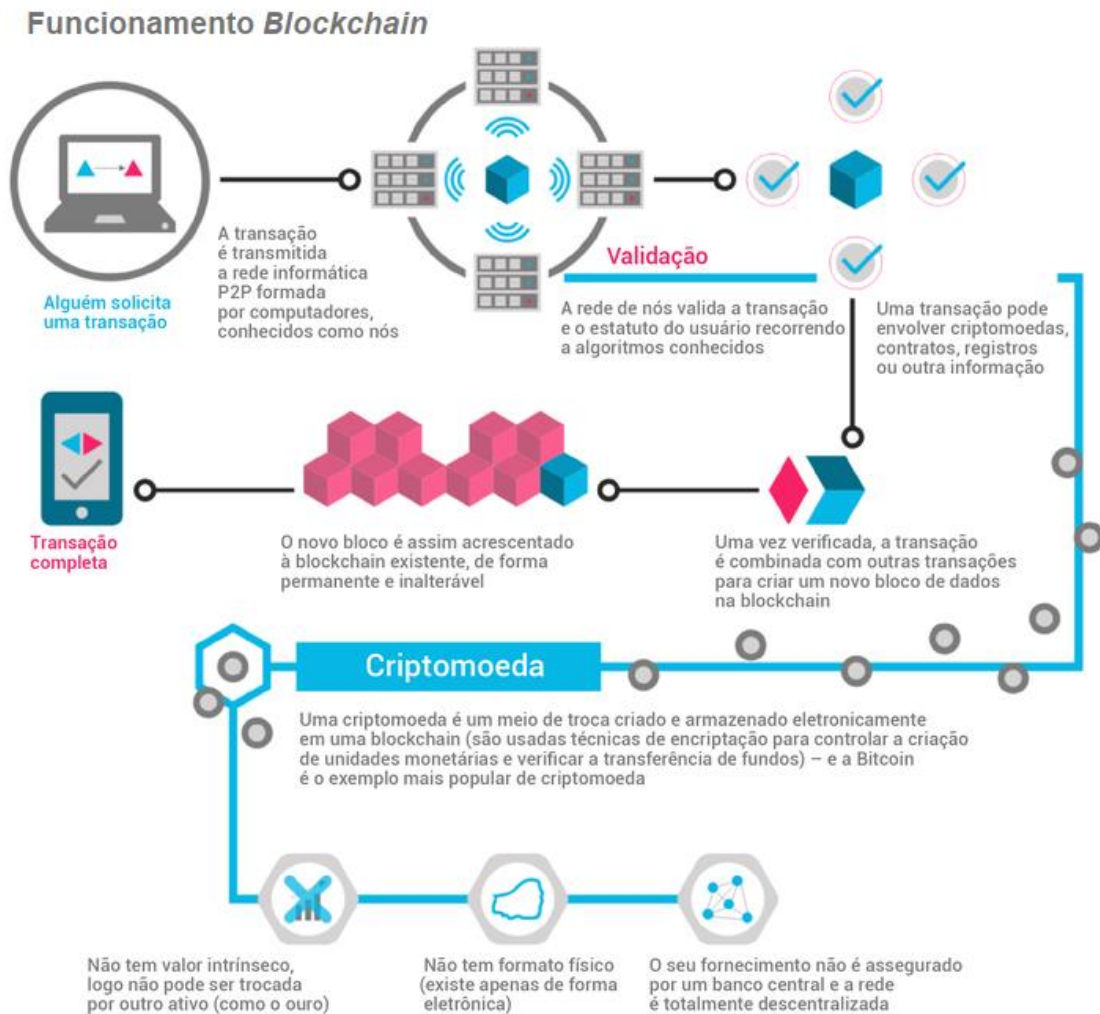
Segundo Corrêa (2017), em invés da informação ser mantida em um único banco de dados, como é feito da forma tradicional, a informação é mantida como cópia para cada dispositivo da rede, conhecido como nó. Esse processo utiliza uma rede *peer-to-peer* (P2P). Caso uma cópia na rede for danificada ou perdida, às outras diversas cópias da rede continuam seguras. Em consequência, se uma informação é alterada em um único nó com o objetivo de inserir um dado falso, todos os nós da rede saberão que a informação é falsa, não inserindo-a na rede.

A *Blockchain* possui esse nome em razão da maneira como funciona e armazena os dados, ou seja, os dados são “empacotados” dentro de um bloco, o qual está “acorrentado” a um outro bloco. O ato de criar uma ligação entre os blocos, formando uma cadeia de blocos é o que faz a *blockchain* ser digna de confiança. Uma vez que o dado está inserido em um bloco, o mesmo não pode ser alterado desde que o bloco anterior seja alterado e assim sucessivamente. Isto é praticamente impossível de ser feito, já que, os nós da rede não concordarão com tal ação (CORRÊA, 2017).

Normalmente, cada bloco possui diversas informações gravadas ao longo que as transações são realizadas, como por exemplo: Alice enviou a Bob 1 Bitcoin, mantendo a data e hora que a informação foi registrada. Por meio da transação, no bloco ainda é registrado a assinatura digital do indivíduo que realizou a transferência e uma identificação única que liga o bloco atual ao anterior, em formato de *hash*. É esta ligação que torna impossível que os dados sejam alterados ou que o um bloco seja inserido entre dois blocos existentes. Como resultado, cada bloco possui uma sequência *hash* do bloco anterior como forma de segurança, pois, à medida que a rede vai crescendo, mais blocos vão sendo criados, e cada vez é mais difícil de violar as informações. Quando combinado, todos esses conceitos criam um armazenamento de dados inquestionável, que não pode ser contestado ou dito como falso (MENDANHA, 2017).

A Figura 9 exibe o fluxo de uma transação incluída na *blockchain*. Após a transação ser solicitada, um nó específico a valida e a insere em um bloco. Posteriormente, este bloco é validado pelos nós restantes da rede e quando concluído, é inserido em definitivo na *blockchain*.

Figura 9 - Funcionamento da blockchain.



Fonte: Adaptada com base em Cardoso (2019).

2.3.2 Tipos de *blockchain*

Atualmente, há três formatos de *blockchain's*, a pública (*permissionless*), privada (*permissioned*) e híbrida. Na pública, o número de nós da rede P2P é desconhecido e sua estrutura é dinâmica, permitindo N entradas ou saídas de nós, não existe um ponto central de controle. Toda transação inserida na rede pode ser

visualizada por qualquer usuário, bem como, qualquer pessoa pode enviar ou verificar as transações, tornando-se um nó ativo da rede. Todos os nós da rede são anônimos, assim como as transações, por meio da criptografia, as informações são transformadas em um código ininteligível. Nessa categoria, classifica-se como pública, todas as *blockchain's* de criptomoedas, como por exemplo a Bitcoin e Ethereum (CORRÊA, 2017).

Uma *blockchain* privada, diferente da pública, possui o total controle dos nós conectados a sua rede, “*permissionados*” por uma entidade ou um grupo de organizações. É indicada para o uso corporativo, para consórcios de empresas de um ramo específico, como por exemplo, no setor de alimentação. Por meio da união de empresas participantes de um determinado processo, uma cadeia de alimentação totalmente rastreável e compartilhada pode ser criada, com o objetivo de conhecer e manter transparente todos os passos que X produto percorreu até chegar na banca de um supermercado. Este tipo de *blockchain* não necessita de uma criptomoeda interna (GREVE *et al.*, 2018).

Segundo REVOREDO (2019), as *blockchain's* híbridas possuem os mesmos benefícios de uma pública e privada. Uma *blockchain* híbrida oferece qualidades superiores a *blockchain's* “puras”, permitindo maior controle e flexibilidade sobre quais dados da rede podem ser compartilhados e quais devem ser mantidos em sigilo. Oferecem também mais rapidez nas transações, recursos de auditoria e segurança.

2.3.3 Rede *peer-to-peer* (P2P)

Em uma rede P2P, os usuários são responsáveis por utilizar e fornecer recursos ao mesmo tempo. Cada *peer*, conhecido como nó, disponibiliza uma parte do seu disco rígido (HD), poder de processamento e de Internet (*download/upload*) diretamente para outros nós, sem a necessidade de uma entidade central ou servidores estáveis (GRIBBLE *et al.*, 2001). Apesar dos nós da rede serem considerados iguais, há alguns que diferenciam-se por desempenhar funções adicionais. Existem os nós completos (*full nodes*), nós mineradores (*miner node*) e

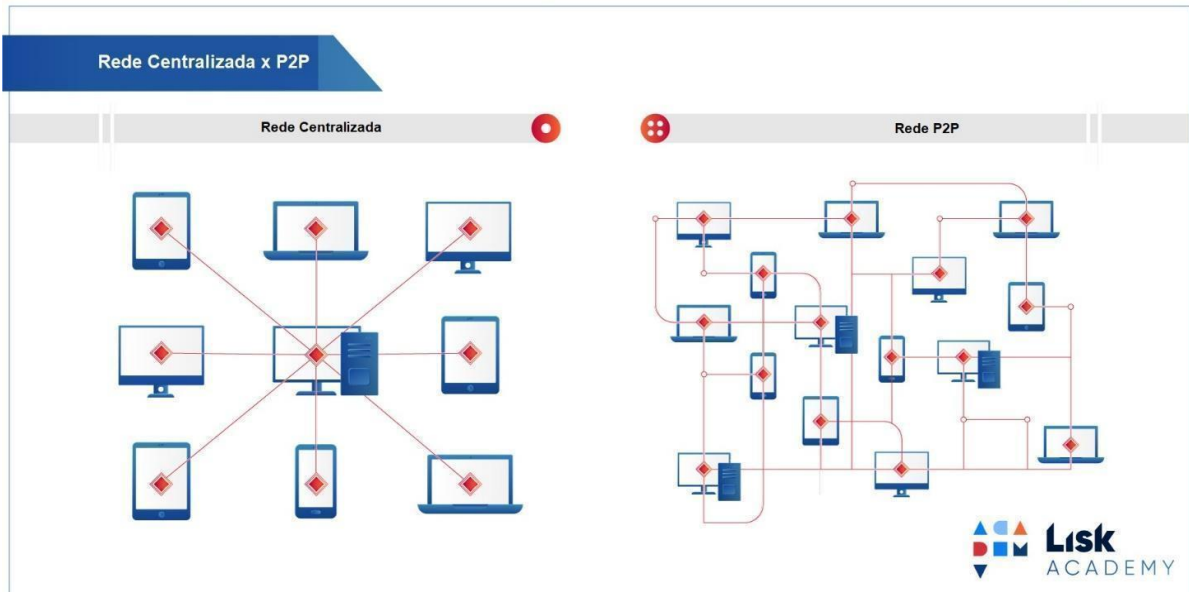
nodes SPV ou Clientes *lightweight* (MARTINS, 2018). Conceitos aprofundados na seção 2.3.3.1.

Uma conexão P2P é completamente diferente do modelo tradicional de cliente-servidor, pois, não há um ponto central de armazenando. Outro diferencial em relação ao modelo tradicional, é que na rede P2P a informação é constantemente registrada e distribuída entre os nós participantes. Como a informação não é armazenada em um único local, este aspecto é um grande avanço para o armazenamento de dados na Internet, já que, um ataque *hacker* precisa ser realizado em todos os nós. Caso a informação for perdida em alguns nós, basta que um possua os dados da respectiva *blockchain*, assim, os nós afetados pelo ataque podem baixar novamente as informações para ter outra vez a cópia original da *blockchain* (GRIBBLE *et al.*, 2001).

O surgimento e o papel que a rede P2P desempenha em conjunto com a tecnologia *blockchain* pode ser visto como um novo sistema de comunicação, totalmente seguro. Mesmo que os nós da rede fiquem “visíveis” durante o processo de concluir as transações ou minerar um bloco, a identidade e toda a informação dos participantes são mantidas em sigilo através da criptografia (MARTINS, 2018).

A Figura 10 exibe como os dispositivos são conectados em uma rede que é centralizada e outra rede que é distribuída. Na rede centralizada, nota-se que há um computador central, responsável por fazer o papel de servidor. Caso este servidor sofrer algum dano, todos os dados serão perdidos. Já na rede P2P, os *full nodes* possuem uma cópia completa de toda a *blockchain*, fazendo com que os dados sejam distribuídos entre os outros nós da rede.

Figura 10 - Diferença de uma rede centralizada x distribuída (P2P).



Fonte: Adaptada com base em Lisk Academy (2019).

2.3.3.1 Nós da rede P2P da *blockchain*

Os nós ou *peer*, são dispositivos responsáveis por manter a integridade das transações e registros da rede da *blockchain*. Todos os nós da rede estão interligados entre si, transformando a rede em um ambiente totalmente distribuído (ELROM, 2019).

Os *Full nodes* é como são chamados os dispositivos encarregados de manter a rede segura, além de armazenar as transações recém realizadas na *blockchain* em um espaço de memória volátil temporariamente, possuem também uma cópia completa da *blockchain* e são responsáveis por validar e retransmitir as transações e blocos seguindo regras de um consenso em comum. O nó denominado como *miner* (minerador) controla o estado da *blockchain*. É responsável por agrupar as transações disponíveis na memória dos *full nodes* em um bloco a ser minerado, recebendo Bitcoins como forma de incentivo por ajudar a rede após o mesmo ser adicionado ao final da cadeia de blocos (ELROM, 2019).

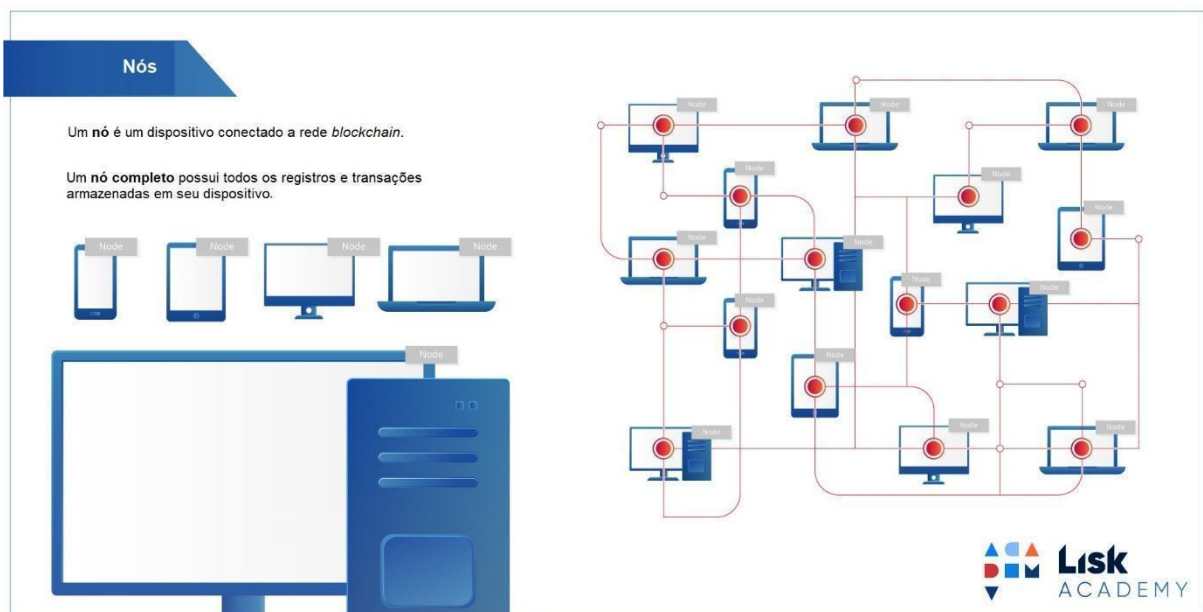
Segundo Martins (2018), *Bitcoin Core* é o nome de um software iniciado por Satoshi Nakamoto, onde através de seu uso o usuário pode ser um nó completo,

possuindo as seguintes funções: mineração, armazenamento de dados, gerenciamento de carteiras e distribuição da informação.

Há ainda um tipo de nó que realiza algumas funções básicas, sendo mais indicado para ser usado em dispositivos com menos capacidade de processamento e memória, como por exemplo, um celular. Este nó é chamado de SPV (MARTINS, 2018). Um nó SPV não precisa possuir toda a *blockchain* baixada em seu dispositivo, uma vez que, é utilizado somente para verificar se uma determinada transação está presente em um bloco específico. Este tipo de nó não se preocupa em verificar se o bloco é malicioso, já que precisa estar ligado diretamente a um *full node*. Em outras palavras, nós SPV são terminais de comunicação que monitoram a rede com constantes validações de transações e geralmente são utilizados pelas *exchanges* e carteiras online (ELROM, 2019).

A Figura 11 exemplifica como os nós estão conectados pela rede e classifica-os de acordo com a sua categoria, podendo ser um: celular, tablet, computador de mesa, notebook, entre outros. Qualquer dispositivo que possuir um IP válido pode ser utilizado como nós, como por exemplo, uma impressora.

Figura 11 - Exemplificação de nós da rede P2P.



Fonte: Adaptada com base em Lisk Academy (2019).

2.3.4 Blocos

Um bloco é uma estrutura de dados que armazena um conjunto de transações e informações da rede *blockchain*. Em cada transação registrada no bloco, há informações sobre a data, o valor monetário referente a criptomoeda e o emissor que realizou a transferência. Possui também, um identificador único, uma espécie de “impressão digital” que permite que todo bloco seja reconhecido pela rede (ALVES *et al.*, 2018).

O identificador único é armazenado no cabeçalho do bloco e é definido pela execução dupla do algoritmo *hash* SHA-256 que possui como entrada os dados dos campos da Tabela 4, formando o que é chamado de *block hash*. Além de salvar o seu identificador, o bloco atual salva o identificador do seu bloco anterior, entendido como bloco pai, formando a cadeia de blocos “*block chain*” na qual chega-se ao bloco gênese, detalhado na seção 2.3.4.1 (MARTINS, 2018).

A Tabela 4 representa a estrutura padrão do cabeçalho de um bloco de qualquer *blockchain*. O campo *previous block hash* grava o identificador do bloco anterior, garantindo que a cadeia de blocos seja formada para verificações futuras. O campo *merkle root* armazena um resumo do conjunto de transações do bloco, assunto detalhado na seção 2.3.4.2. O campo *timestamp* é responsável por salvar a data, hora, minuto e segundos que o bloco foi criado. Os campos *difficulty target* e *nonce* fazem referência a mineração do bloco e serão detalhados na seção 2.3.5. De acordo com Sousa (2019), um bloco da Ethereum possui campos com outras nomenclaturas, mas que remetem ao mesmo funcionamento da Bitcoin, como por exemplo o *parentHash* que é o *previous block hash* da BTC. Há também campos adicionais, como por exemplo o *gasLimit* e *gasUsed*, ambos contabilizam informações referentes ao *gas* utilizado na mineração do bloco.

Tabela 3 - Estrutura do cabeçalho de um bloco da rede.

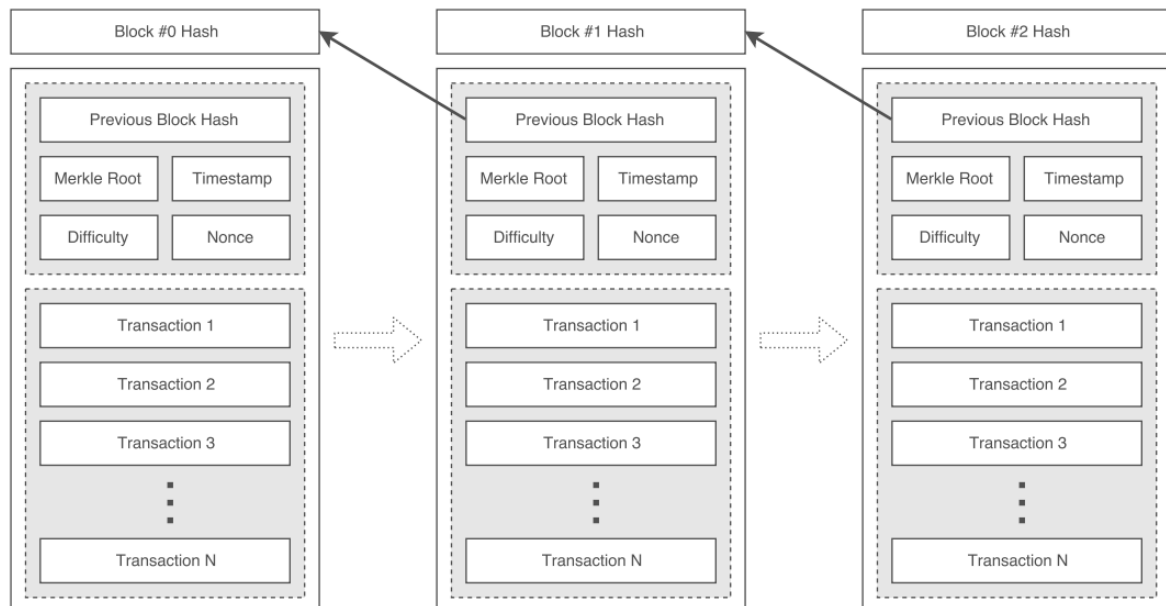
Campo	Definição
<i>Version</i>	Número usado para rastrear a versão do protocolo/ <i>software</i> utilizado.
<i>Previous Block Hash</i>	Identificador do bloco anterior.
<i>Merkle Root</i>	Um resumo de todas as transações inseridas no bloco.

<i>Timestamp</i>	Hora aproximada em que o bloco foi criado. É salvo no formato <i>Unix Epoch</i> .
<i>Difficulty Target</i>	Dificuldade estabelecida por um determinado algoritmo para um novo bloco ser minerado.
<i>Nonce</i>	Contador usado no cálculo da mineração de um bloco.

Fonte: Adaptada com base em Martins (2018, p. 31).

A Figura 12 exemplifica o encadeamento formado pelos blocos. A *hash* que é calculada duas vezes é representada no cabeçalho do bloco. Acima do cabeçalho, há um contador que identifica a posição em que o bloco foi criado (identificado como *block height*), porém, esse contador não faz parte do identificador único. O bloco gênese é o bloco de posição “#0”. O campo *previous block hash* faz referência ao identificador único do seu bloco anterior.

Figura 12 - Encadeamento dos blocos.



Fonte: Martins (2018, p. 31).

2.3.4.1 Bloco Gênese

Singh's (2016) explica que, o primeiro bloco minerado da *blockchain* do protocolo Bitcoin é referenciado como bloco gênese. É o bloco pai mais antigo de todos os blocos minerados após ele, deste modo, se uma linha reta for estabelecida com todas as transações atuais, será possível chegar ao bloco gênese efetuando o caminho inverso. Este bloco está codificado e presente por padrão em todo o

software que utiliza o protocolo da Bitcoin, um exemplo é o Bitcoin Core. De acordo com Blockchain.com (2019), seu *block hash* é `00000000019d6689c085ae165831e934ff763ae46a2a6c172b3f1b60a8ce26f` e todos os nós conectados à rede o conhecem. É a partir dessa *hash* que os nós estabelecem uma rota segura para manter uma cópia da *blockchain* atualizada e verificam se uma transação é confiável. De acordo com Corrêa (2019), a mineração do bloco gênese da Ethereum ocorreu em 20 de julho de 2015, data em que a rede oficialmente entrou em operação.

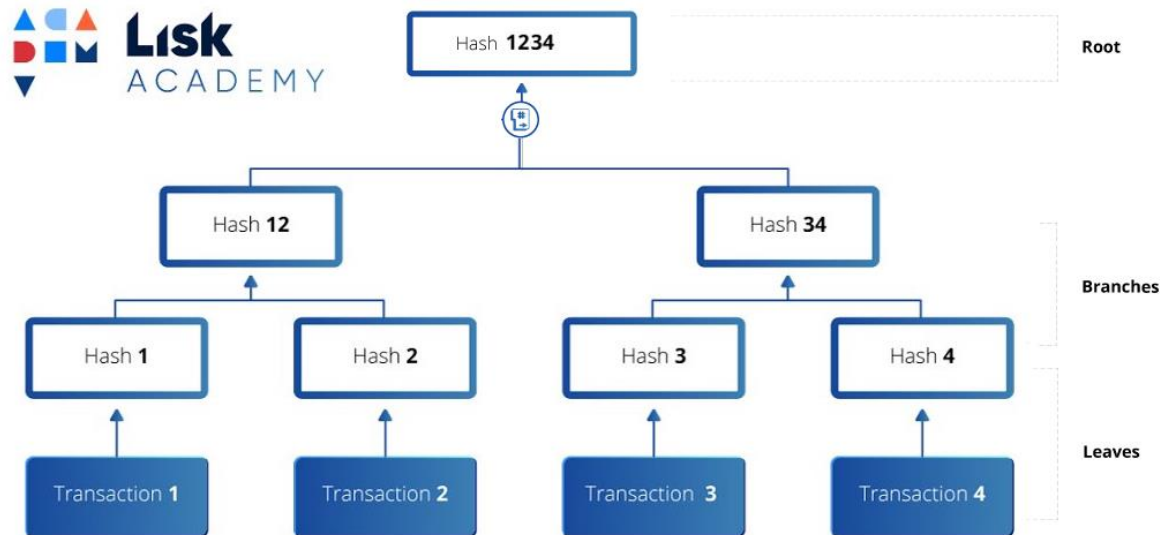
A primeira e única transação realizada no bloco gênese na Bitcoin, consequentemente criando o primeiro endereço da rede, foi de 50 BTC. A transação do tipo *coinbase* foi utilizada (detalhada na seção 2.3.5), contendo a mensagem codificada “*The Times 03/Jan/2009 Chancellor on brink of second bailout for banks*”, alusiva à uma matéria divulgada pelo jornal The Times, estabelecendo a data de constituição da *blockchain* (MARTINS, 2018).

2.3.4.2 Árvores de Merkle

As Árvores de Merkle ou Árvores de Dispersão são árvores binárias utilizadas para agrupar e verificar de maneira eficaz a integridade de volumosos conjuntos de dados. No protocolo Bitcoin, as árvores são usadas para gerar uma *hash* única, conhecido como *Merkle Root*. Na prática, cada transação do bloco é resumida pela função *hash SHA-256*. O resumo então, é concatenado a um resumo de outra transação, sempre em pares, no formato de árvore binária. Dado a estrutura criada se assemelhar a uma árvore, as transações iniciais são chamadas de “*leaves*” (folhas), as transações do meio de “*branches*” (galhos) e o resumo *hash* do topo, de “*root*” (raiz) (MARTINS, 2018). Buterin (2015) explica que o funcionamento das árvores na Ethereum segue os mesmos conceitos da Bitcoin.

Na Figura 13, um par é formado pelo resumo *hash* da transação “1” e pela *hash* da transação “2”, produzindo a *hash* concatenada “12”. No par ao lado, a *hash* “34” é obtida das transações “3” e “4”. Ao final, a *hash* “12” é concatenada ao par “34”, tendo como raiz (*merkle root*) a *hash* “1234”.

Figura 13 - Criação do resumo *Merkle Root*.



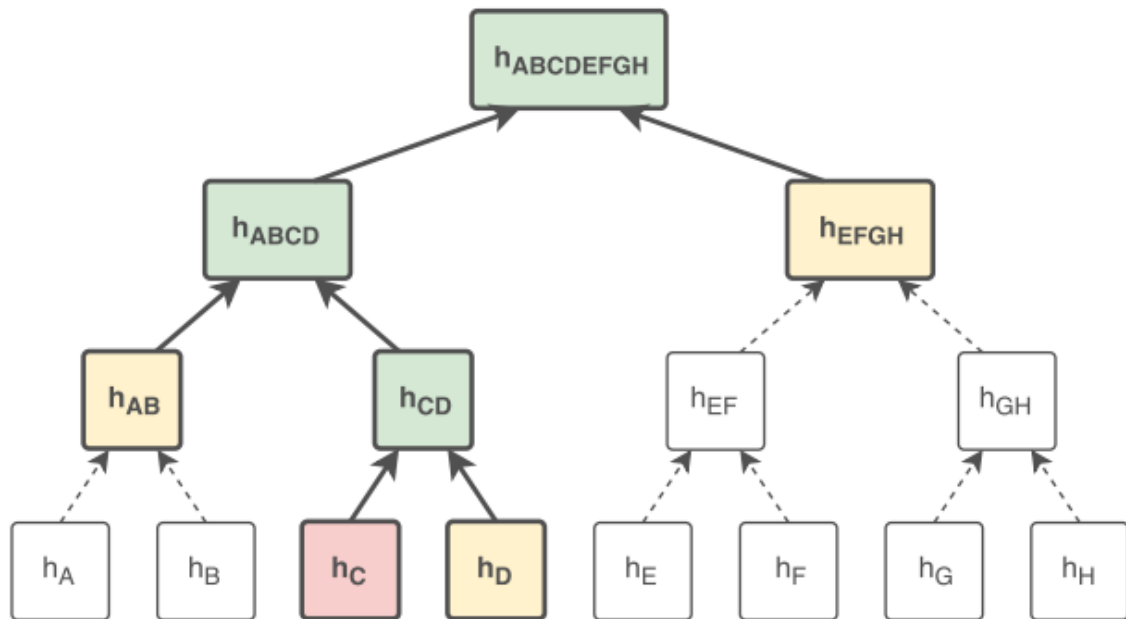
Fonte: Adaptada com base em Lisk Academy (2019).

2.3.4.3 Verificação de Pagamento Simplificado – Nós SPV

Atualmente, a maioria das *blockchain's* requerem um espaço considerável disponível no disco rígido (HD), vide a *blockchain* da criptomoeda Ethereum que possui mais de 30 Gb e a Bitcoin, com mais de 240 Gb. Para casos onde o dispositivo não possui tal quantidade disponível, um nó de formato “leve” pode ser adotado. Esses dispositivos são conhecidos como clientes SPV e são responsáveis por verificar se uma transação específica está presente em um bloco. Nós do tipo SPV não precisam baixar toda a *blockchain*, portanto, utilizam as árvores de *Merkle* como auxílio para verificar uma transação, onde é necessário ter apenas as transações pares que geram a raiz da árvore de acordo com a transação que se deseja verificar (GRUBER, LI, KARAME, 2018).

Na Figura 14, um determinado nó SPV quer saber se a transação h_C está presente em um respectivo bloco. Um *full node* “vizinho” ao nó SPV retorna apenas as transações necessárias para que o nó SPV calcule o resumo da árvore de *Merkle*. Com as transações h_D , h_{AB} e h_{EFGH} , o nó consegue realizar a verificação, sabendo se a transação pertence ou não ao bloco.

Figura 14 - Verificação de uma transação de um nó SPV.



Fonte: Martins (2018, p. 34).

O nó SPV possui somente as informações do cabeçalho do bloco em questão, a raiz de *Merkle* (*Merkle Root*) é uma delas, logo, o *full node* ligado diretamente ao nó SPV fornece todas as transações necessárias para que o nó calcule o resumo da raiz. Se a raiz calculada for idêntica a raiz do cabeçalho, o nó SPV possui uma prova incontestável que a transação pertence a determinado bloco. Por meio da árvore de *Merkle*, pode-se dizer que a verificação de transações é eficiente, pois, em um bloco com N transações a complexidade produzida para comprovar uma transação é de $2 \log_2 N$ valores de *hash* (MARTINS, 2018).

2.3.5 Mineração do protocolo Bitcoin



A mineração tem como propósito a inclusão de blocos ao final da *blockchain*, permitindo que ela escale com segurança e transparência. Como forma de incentivo, após o novo bloco ser incluída na *blockchain*, uma recompensa em BTC é entregue ao nó minerador, sendo que, a taxa de cada transação registrada no bloco também é direcionada ao mesmo nó. Quando um bloco é inserido ao final da cadeia, todas as suas transações recebem o que é chamado de “confirmação”. Do mesmo modo que, a cada vez que um novo bloco é inserido, as transações dos

blocos anteriores também recebem uma confirmação. Tem-se como medida de segurança, onde uma transação pode ser considerada totalmente assegurada e imutável com o total de 6 confirmações. O número de confirmações representa o nível de profundidade da transação na *blockchain* (MARTINS, 2018).

A Figura 15 exibe uma transação inserida no bloco de número 598859 no dia 11 de outubro de 2019. Do dia 11 até o dia 18 de junho de 2020, o número total de confirmações foi de 36.482, ou seja, 36.482 blocos foram minerados após essa transação.

Figura 15 - Número de confirmações no bloco 598859.

Detalhes

Jogo da velha	f8f18d6cb6e02e9d3477c3d04dda5eaa174d2cac6dbf15896dcfbc7fea58603f
Status	Confirmado
Hora da Recepção	<u>2019-10-11 01:00</u>
Tamanho	291 bytes
Peso	1.164
Incluído no Bloco	 598859
Confirmações	36.482 
Total de Entrada	0.00168571 BTC
Total de Saída	0.00167698 BTC
Taxas	0.00000873 BTC
Taxa por byte	3.000 sat/B
Taxa por unidade de peso	0.750 sat/WU
Valor quando transacionado	US\$ 14,38

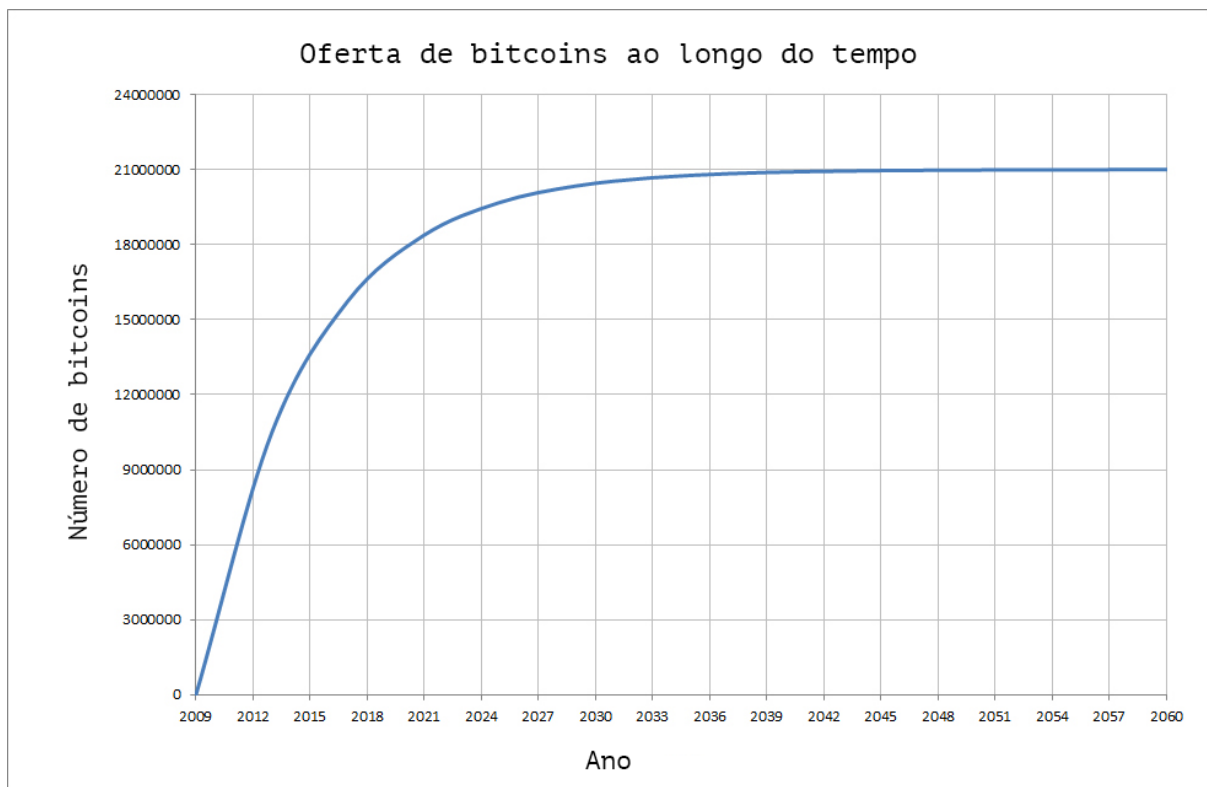
Fonte: Adaptada com base em Blockchain.Info (2020).

As moedas recompensadas ao minerador responsável por inserir um bloco a cadeia são criadas através de uma transação chamada de *coinbase*. Por ser a primeira transação do bloco, transações do tipo *coinbase* não possuem referências de saída, somente de entrada. O total de moedas entregue ao minerador é relativo ao total de blocos já gerados (ELROM, 2019). A primeira transação da história da

Bitcoin pelo bloco gênese gerou um total de 50 BTC, e desde então, esse valor é reduzido em 50% a cada 210.000 blocos – evento conhecido como *halving*. Um bloco é gerado em média a cada 10 minutos, logo, um *halving* acontece aproximadamente a cada 4 anos. A recompensa atual é de 12.5 BTC (em 20 de outubro de 2019) e no próximo *halving* será de 6.25 BTC. O alvo da Bitcoin é garantir que o número de bitcoins existentes não ultrapasse os 21 milhões, limitando a base monetária para que uma severa deflação não ocorra. Conforme o total de BTC dado como incentivo ir diminuindo, a tendência é que as taxas de cada transação sejam aumentadas gradativamente até o limite ser atingido, após este fato ocorrer, os mineradores serão recompensados somente por meio das taxas das transações (SINGH'S, 2016).

O Gráfico 2 representa o número total de Bitcoins minerados de acordo com os anos. Atualmente tem-se minerado aproximadamente 18 milhões de moedas, e estima-se que em 2040 o número limite de 21 milhões seja alcançado.

Gráfico 2 - Oferta de Bitcoins ao longo do tempo.



Fonte: Agner (2018, p. 80).

2.3.6 Comparativo entre Bitcoin x Ethereum

Bitcoin e Ethereum são semelhantes em diversos aspectos, no entanto, em razão da Ethereum ter sido lançada após a Bitcoin com o objetivo de atender outras soluções e não apenas a financeira, elas possuem algumas diferenças. A Tabela 4 exibe um comparativo entre as suas características.

Tabela 4 - Comparativo de características entre Bitcoin X Ethereum.

Características	Bitcoin	Ethereum
Tipo	Pública (Martins, 2018).	Pública (Revoredo, 2019).
Carteiras e endereços	Um tipo de carteira. Usada para realizar transferências entre os endereços (Martins, 2018).	Dois tipos de carteira. A externa, igual a Bitcoin, utilizada para enviar e assinar transações e outra para contratos, que executam códigos e condições (Revoredo, 2019).
Como o saldo é definido	Entre as saídas não gastas. UTXO (Martins, 2018).	Na conta/endereço (Revoredo, 2019).
Protocolo de consenso	<i>Proof of Work</i> (Martins, 2018).	<i>Proof of Work</i> (Revoredo, 2019).
Tempo para minerar um bloco (Revoredo, 2019).	Aproximadamente 10 minutos.	Aproximadamente 15-20 segundos.
Transação x Tempo (Revoredo, 2019).	Baixo custo: 7 transações por segundo. Alto custo: 3 transações por segundo.	Aproximadamente 300 transações por bloco. 15 Transações por segundo.
Sistema de taxas (Revoredo, 2019).	Pago em bitcoins.	Pago em <i>ethers</i> convertidos de <i>gas</i> de acordo com o consumido em uma transação ou execução de contrato.
Tamanho por bloco (MB) (Revoredo, 2019).	Sempre 1 MB.	Até 1 MB. É definido dinamicamente pelos minerados.
Ranking no mercado de criptoativos (CoinMarketCap, 2020).	1ª posição.	2ª posição.

Fonte: Elaborado pelo autor (2020).

2.4 Provas de Consenso

As provas de consensos são um dos aspectos mais importantes da tecnologia *blockchain*. Estes protocolos criam um sistema de consenso irrefutável entre os dispositivos de uma rede distribuída, impedindo que informações falsas sejam inseridas ao longo da cadeia de blocos, tornando-se peça-chave para a segurança e manutenção da *blockchain* (AGNER, 2018). As principais provas existentes atualmente e suas devidas soluções serão apresentadas nessa seção.

2.4.1 Solução

Para um bloco ser inserido a *blockchain*, todos os nós seguem regras de uma prova de consenso em comum. A prova estabelecida busca suprir os problemas de corretude, acordo e utilidade. Por meio do aspecto de corretude, os nós verificadores devem conseguir diferenciar uma transação maliciosa de uma transação legítima. Através do acordo, a cadeia de blocos deve-se manter confiável e única, e a partir da utilidade, a aplicação da solução disponibilizada pela *blockchain* deve ser útil, não demorando na verificação de transações e mineração de blocos. Por exemplo, se uma *blockchain* leva mais de um dia para concluir uma única transação, seu uso não é prático, portanto, a solução não é eficaz (MENDANHA, 2017).

É a partir de uma prova de consenso, estabelecida como regra da rede, que os nós respondem à pergunta “Como garantimos com total certeza o que é verdadeiro e o que é falso?”. O consenso estabelecido pelos nós da rede garante o mesmo estado de toda a *blockchain*, tornando-a um ecossistema independente de auditoria. As principais funções de uma prova de consenso são: manter a cadeia de blocos sempre atualizada, garantindo que todo novo bloco seja verdadeiro, enquanto ao mesmo tempo, incentiva os indivíduos da rede a realizarem tal ação; verificar se uma nova transação não foi forjada; impedir que uma única entidade controle ou prejudique toda a cadeia; mesmo que alguns nós falhem em concordar com determinado consenso, se a rede obter mais de 51% de unanimidade, o novo bloco ou transação é considerado verdadeira; evitar problemas de gasto duplo e dos generais Bizantinos (NAKAMOTO, 2008).

2.4.2 Problema do gasto duplo

Pela característica de descentralização da *blockchain*, não é necessário que haja uma autoridade central responsável por gerenciar as criptomoedas dos usuários da rede, papel esse, que é atribuído aos nós. Todavia, um indivíduo pode tentar duplicar suas moedas realizando duas transações com a mesma quantia, neste caso, os nós responsáveis entram em um consenso para verificar qual das transações deve ser considerada válida e qual deve ser descartada. Atualmente existem vários algoritmos de consensos, a exemplo do algoritmo conhecido como *Proof of Work* ou prova de trabalho (detalhado na seção 2.4.4), foi o primeiro a solucionar esta falha com eficácia, posteriormente, outras provas de consenso foram surgindo, buscando solucionar problemas evidenciados em consensos já existentes (ANTONOPOULOS, 2014).

2.4.3 Problema dos Generais Bizantinos

Segundo Martins (2018), o problema dos generais bizantinos (*Byzantine Fault Tolerance*) foi desenvolvido para solucionar falhas em sistemas distribuídos, onde um dos problemas consiste em tentar fazer com que todos os atores obtenham sucesso na troca de informações em uma rede comprometida, ou seja, a informação distribuída deve ser legítima mesmo que na rede, exista atores tentando inserir informações forjadas. Por meio deste problema, pode-se usar como analogia uma rede P2P, na qual um nó malicioso tenta inserir milhares de blocos e transações com informações falsas, com o objetivo de prejudicar a rede ou realizar gastos duplos, duplicando sua quantidade de moedas indevidamente.

2.4.4 Proof of Work (PoW)

Criada em conjunto com a Bitcoin, o consenso prova de trabalho ou *Proof of Work* possui esse nome em razão dos nós mineradores calcularem repetidamente uma *hash* alvo estabelecido pelo cálculo ($difficulty_target = original_target / target$) (ANTONOPOULOS, 2014). Todos os nós da rede que contribuem na mineração estão em uma ininterrupta disputa mundial para solucionar a equação responsável por permitir que um bloco seja adicionado ao final da cadeia. A rede possui uma variável chamada de *difficulty_target* responsável por sinalizar qual a *hash* (*block*

hash) esperada do próximo bloco a ser atualizado na *blockchain*. Por meio do campo *nonce*, escolhido arbitrariamente pelo minerador (geralmente incrementando +1 a cada iteração do *while*) o cabeçalho do bloco a ser inserido é calculado utilizando a função SHA-256 duas vezes ($shasha = sha256(sha256().digest())$), tendo como entrada os dados das variáveis do cabeçalho do mesmo. O cálculo é repetido até que a *hash* encontrado seja menor que o alvo estabelecido. Quando encontrado, o novo bloco deve ser distribuído pela rede, exibindo claramente as variáveis utilizadas para se chegar na *hash*, uma vez validado pelos *full nodes* da rede, o bloco é adicionado ao final da cadeia e o minerador recebe criptomoedas como recompensa. Ao final da mineração de um bloco, todos os nós mineradores passam a trabalhar imediatamente na geração do próximo bloco (MARTINS, 2018).

Com a evolução contínua do poder computacional, e com a adesão de novos mineradores a rede, inevitavelmente, alguns desses conseguirão gerar blocos em menos de 10 minutos. Como consequência, caso um total de 2016 blocos forem criados em menos de duas semanas, o cálculo da prova de consenso é ajustado com o propósito de manter os 10 minutos necessários para minerar um bloco válido. Para a rede Bitcoin, esse tempo deve ser mantido constantemente, caso contrário, os 21 milhões de BTC seriam minerados em um curto espaço de tempo (ANTONOPOULOS, 2014).

2.4.5 Proof of Stake (PoS)

A prova de consenso conhecida como *Proof of Stake* (PoS) foi criada em 2012 por Sunny King e Scott Nadal como alternativa para resolver problemas evidenciados pela PoW, como por exemplo, o consumo de energia excessivo pela prova de trabalho, a possível baixa na escalabilidade, uma vez que a cada 210.000 blocos há o *halving*, diminuindo o incentivo na mineração de blocos, e principalmente para evitar o ataque de 51%, onde é possível efetuar um gasto duplo possuindo o controle de 51% ou mais da rede, removendo da rede transações na qual o montante de moedas já foi transferido para outro endereço, assim, as moedas acabam ficando na carteira do endereço de origem. A prova estabelecida pela PoS é baseada no número de moedas que um nó possui, por consequência, se um nó quer minerar, é preciso juntar um número considerável de moedas. Nela, ao invés

de poder computacional (*hash power*), tem-se “*stack power*”, a qual não é necessário depender de energia, pois, não há nenhuma equação para ser resolvida. O identificador único do bloco é similar ao do PoW, porém, os nós são limitados, garantindo a segurança necessária e diminuindo os custos de energia. O incentivo ao nó minerador é oriundo das taxas de transações da rede (ELROM, 2019).

A função desempenhada pelo nó é definida pela quantidade de moedas que possui, assim, a distribuição da informação é alcançada em consenso com menos gasto de energia e custos no geral. Ataques de negação de serviços (DDOS) e fraudes ainda são possíveis, no entanto, os atacantes não conseguem forjar novas moedas, com isso, caso alguém arriscar-se em tentar falsificar algum bloco, há chances de o dinheiro ser perdido, por esse motivo, a possibilidade de um ataque é baixa. Qualquer nó da rede pode ser um minerador através do *stack* de moedas. Atualmente, existem duas maneiras de tornar-se um nó minerador; um nó pode utilizar um nó considerado como honesto para ajudar no *stacking*, ou então, há a possibilidade de votar em *full nodes*. A descentralização é limitada ao passo que somente alguns nós conseguem guardar a maioria das moedas, tendo assim, maior controle sobre a rede. Para o trabalho de minerar, um nó é escolhido aleatoriamente (MENDANHA, 2017). A Tabela 5 compara as principais características da PoW e PoS.

Tabela 5 - Comparativo dos consensos PoW e PoS.

Característica	PoW	PoS
Geração de novos blocos	O primeiro nó que resolver uma equação através do seu poder de processamento.	Seleção aleatória baseada na quantidade de moedas guardadas por um único nó.
Recompensa/incentivo	Ao minerar um bloco e através das taxas de transações contidas nele.	Taxa de transações da rede.
Energia e recursos consumidos	Placas específicas de mineração (ASIC) e grande quantidade de espaço ocupado em disco.	Pouco consumo de energia elétrica.

Fonte: Elrom (2019).

Além dos consensos citados acima, conforme explica Corrêa (2017), há ainda os consensos *Practical Byzantine Fault Tolerance* (PBFT), *Delegate Proof-of-*

Stake (DPOS) e Lista de Nós Únicos (LNU), utilizada pela *altcoin* Ripple em *blockchain* privadas.

2.5 Interplanetary File System - IPFS

O IPFS é um sistema P2P que conecta vários dispositivos a um sistema de arquivos por meio de um modelo de armazenamento endereçado de acordo com o conteúdo do arquivo. Todos os nós possuem a mesma influência sobre a rede e não precisam confiar um no outro para conectarem entre si e realizarem a transferência de arquivos. O conceito por trás do IPFS consiste em gerar a *hash* única e imutável do arquivo replicando o arquivo para os demais nós da rede, portanto, se um arquivo possuir a mesma *hash*, mesmo que seu nome seja diferente, ele será armazenado uma única vez. Os arquivos são acessados através de um link estático mais a *hash* do documento, por exemplo: o arquivo X gerou a *hash* “QmfHvMPYw3TmxYmLY7hU3yVEZ5bSRmB3YBJM6TZfJJf3Pp”, então, para visualizar este arquivo o link a ser acessado é o <https://ipfs.io/ipfs/QmfHvMPYw3TmxYmLY7hU3yVEZ5bSRmB3YBJM6TZfJJf3Pp>. (MENDANHA, 2017). Esta tecnologia será utilizada em conjunto com a *blockchain* para atender os objetivos do trabalho.

3. TRABALHOS RELACIONADOS

Esta seção apresenta os trabalhos relacionados que auxiliaram no entendimento do assunto e no desenvolvimento do referencial teórico da presente monografia. Conforme alguns autores evidenciaram em seus textos, a tecnologia *blockchain* está em constante evolução e vem sendo utilizada como solução para modelos de diferentes negócios, e não somente para transações financeiras como proposto originalmente.

3.1 Estudo 1

Corrêa (2017), constata que aplicações baseadas na tecnologia *blockchain* podem garantir acurácia na tempestividade de arquivos adicionados a rede. Explicou as diretrizes e conceitos utilizados em conjunto com a *blockchain*, como, criptografia simétrica e assimétrica, certificado e assinatura digital, *blockchain's* pública e privada, estrutura completa de um bloco, rede P2P, provas de consenso e mencionou sobre as criptomoedas Bitcoin e Ethereum.

Corrêa (2017), desenvolveu dois protótipos. O primeiro é uma aplicação para baixar os elementos da *Hyperledger Fabric* da empresa IBM situados sobre *container's* e subir instâncias por meio do *docker* para popular uma rede P2P interna. Os comandos foram executados em uma VM do Linux, embora, tenha destacado que pode ser utilizado também o *PowerShell* do *Windows*. Ao executar as instruções necessárias, o *docker* baixou e subiu uma instância com a *Hyperledger* instalada. Com todas as imagens baixadas, a ferramenta criou a *chaincode*, simulando os nós da rede P2P. A *Hyperledger Fabric* não possui custos em transações e utiliza a prova de consenso *Proof of Stake*.

O segundo protótipo utilizou a ferramenta *Hyperledger Composer Playground* também desenvolvida pela IBM, para criar de forma simples e intuitiva uma *blockchain* com os atributos e parâmetros necessários para seu funcionamento. Os resultados dos testes são exibidos por meio de uma interface amigável. Mediante os testes, foi possível obter o *timestamp* (data e hora) do momento em que o arquivo foi inserido em um bloco, obtendo a comprovação do registro de tempo fornecido pela *blockchain*.

3.2 Estudo 2

No trabalho de Martins (2018), é enfatizado que são necessários fundos em um endereço Bitcoin para que uma transação seja efetivada pelos mineradores. Inicialmente, desenvolveu uma aplicação para a criação de um endereço, permitindo que uma chave privada seja gerada por meio de uma frase aleatória ou uma frase escolhida pelo usuário. A frase informada é transformada em um texto criptografado com auxílio do algoritmo SHA-256. A chave pública foi gerada através da biblioteca *pyBitcoinTools* mediante a multiplicação de curvas elípticas e o ponto gerador da curva *secp256k1*. O endereço parcial foi então criptografado pela função *hash* SHA-256 e na sequência pela função *RIPEMD-160*. Tem como endereço público final o resultado parcial codificado em *Base58Check*.

Martins (2018), desenvolveu uma aplicação para realizar transações por meio da *blockchain* do Bitcoin. A linguagem utilizada foi Python e contou com o auxílio da API BlockCypher na qual não está ligada diretamente à rede P2P. Usou as chaves geradas, privada e pública para inserir o DNA extraído de um documento. O registro na rede é feito por meio de um *script* de saída, *OP_RETURN*, sendo que a taxa cobrada pelo minerador é na ordem de 0,0001 BTC. O registro foi validado por um minerador e inserido em um bloco na rede após aproximadamente 10 minutos, tempo esse, que é o estimado para que um novo bloco seja minerado.

Para comprovar a existência do documento, o autor utilizou em sua aplicação, outra API, disponibilizada pela SmartBit, usando o *script* de saída e informando o DNA do documento após o *OP_RETURN*, se o DNA já está inserido

na *blockchain*, o registro será encontrado retornando a data exata em que o mesmo foi de fato inserido na rede, comprovando sua existência.

Martins (2018), realizou ainda uma pesquisa de viabilização do uso da *blockchain* do Bitcoin, uma vez que, as taxas cobradas pelos mineradores são altas, se comparado a outras *blockchain's*. A pesquisa, efetuada em 2017, constatou que no início do ano a taxa cobrada era de \$ 0,35 por transação. No final do ano, a taxa era de \$ 25,17 por transação, um aumento de 7091% neste período. Neste mesmo período, usuários da *blockchain* da criptomoeda Ethereum eram cobrados em média \$ 0,66 por transação.

3.3 Estudo 3

Mendanha (2017), implementou uma aplicação semelhante ao Estudo 1 (seção 3.1) e Estudo 2 (seção 3.2) por meio de uma documentação detalhada, envolvendo documento de requisitos funcionais, diagrama de componentes e sequência. Sua proposta difere dos demais, pois, possui como adicional a possibilidade de armazenar o documento para fazer o *download* de seu comprovante através da ferramenta IPFS. Utilizou como auxílio o banco de dados BigchainDB.

O serviço foi disponibilizado em um *Web Service* da empresa Amazon, dividindo-o em três instâncias. Uma instância exclusiva para o banco de dados, outra para a ferramenta IPFS e outra para a comunicação entre IPFS e BigchainDB através dos protocolos HTTP e TCP/IP.

Mendanha (2017) destaca que, a grande vantagem de usar o BigchainDB é pela sua escalabilidade. Uma vez que seja preciso aumentar o processamento, o modelo de escalabilidade horizontal pode ser aplicado, criando instâncias e consequentemente distribuindo o serviço, obtendo maior desempenho computacional.

A aplicação desenvolvida permite ao usuário gerar um par de chaves, privada e pública; fazer o upload de um arquivo PDF; obter o DNA desse arquivo; registrar e consultar o arquivo na *blockchain*; baixar o “comprovante” de existência do arquivo

e transferir o DNA do arquivo, informando sua chave privada e uma chave pública destino. O autor realizou um questionário de avaliação para medir a experiência do usuário. Ao final, por meio do questionário, concluiu que os usuários entrevistados souberam utilizar o programa, sabendo também qual é o seu propósito.

3.4 Estudo 4

Bartoletti *et al.* (2017), desenvolveram um *framework* para analisar as diferentes informações contidas na *blockchain* da Bitcoin, Ethereum e dados de sites externos, como *wiki's*, fóruns de discussão e site específicos. Em 2017, as respectivas *blockchain's* ocupavam cada um total de 130 Gb e 300 Gb. Os autores destacam que apenas uma parte dessa quantia de armazenamento é relativa às transações e que realizar esse tipo de análise permite obter "*insights*", como por exemplo, indicadores econômicos que preveem a tendência do mercado; tentativas de atividades criminosas, como o ataque de negação de serviço; *ransomware*; taxa média por transação, quantidade de blocos gerados por dia, mês, ano, o uso de metadados no geral, entre outros.

Para a *blockchain* da Bitcoin, utilizou-se como auxílio a biblioteca BitcoinJ e a interface do *software* Bitcoin Core. Já para a *blockchain* da Ethereum, as bibliotecas utilizadas foram as Parity e web3j. Os dados foram capturados a partir do bloco de posição 473100 de ambas as *blockchain's*. Analisou-se os metadados do retorno do parâmetro OP_RETURN, as taxas das transações, e a identificação de cada endereço público, buscando encontrar ocorrências referentes a possíveis crimes realizados. Para os dados externos, buscou-se coletar apenas informações de sites com mais de mil transações (exemplos: colu.com e omnilayer.org). Os dados analisados foram armazenados nos bancos de dados MongoDB e Mysql e gerenciados com a linguagem SQL do *software* DBMS, com o objetivo de observar a respectiva escalabilidade.

O *framework* desenvolvido permite navegar entre os blocos de qualquer *blockchain*. No caso da Bitcoin, os seguintes passos são realizados: por meio do Bitcoin Core, o *hash h (block hash)* é capturado de um bloco de posição *i (block height)*; o *hash h* é coletado pela biblioteca BitcoinJ; a posição *i* é incrementada a

cada *loop*. O *loop* começa na posição 0 e é finalizado no último bloco. Pelos métodos *blockchain.start(i)* e *blockchain.end(j)* é possível escolher qualquer intervalo de bloco. Os experimentos foram realizados através de um computador com as especificações; Quad-core Intel Core i5-4440 CPU @ 3.10 GHz; 32 Gb de memória RAM e 2 Tb de disco rígido.

Ao final do experimento, os autores chegaram à conclusão de que o *framework* desenvolvido é eficaz e cumpre o que é chamado de “*ad-hoc approach*”, sendo um *framework* com propósito único, implementado a fim de realizar análises para cruzar dados de *blockchain's* e de sites externos. O experimento foi desenvolvido com as melhores práticas de reutilização, portanto, por ser *open source*, uma comunidade específica ou indivíduos da área podem utilizar o *framework* livremente. Em relação aos bancos de dados usados, a partir do trabalho, não foi possível evidenciar diferenças significantes no desempenho das *queries*, mesmo sem a utilização de *joins*, no entanto, o MondoDB mostrou-se mais eficiente na escrita das consultas.

3.5 Estudo 5

Homayoun *et al.* (2019), explica que em 2017, aproximadamente 5 bilhões de pessoas estavam conectadas por meio de um celular, demonstrando o impacto do uso de celulares na vida do ser humano. De acordo com pesquisas da empresa McAfee, 2017 foi o ano de explosão de vírus para celulares. Em 2017, o antivírus Kaspersky detectou 5,730,916 instalações maliciosas, 94,368 *Trojans* para aplicativos de bancos (conhecido como *Internet Banking*) e 544,107 programas de tipo “*ransomware Trojans*”. Em razão disso, os autores propuseram-se desenvolver um *framework* para a detecção de programas maliciosos em lojas de *download* de aplicativos antes de serem de fato baixados pelo usuário final. Com a combinação de análises estáticas e dinâmicas para encontrar estes programas maliciosos, foi possível detectar com mais precisão a presença destes programas. O *framework* é especializado em buscar vírus no geral, em todos os arquivos da estrutura de um celular, ou seja, o B2MDF (*Blockchain-Based Malware Detection Framework*) pode ser utilizado para qualquer algoritmo de *machine learning*. O B2MDF ainda dispõe

de melhorias que podem ser utilizadas por terceiras partes para desenvolverem o seu próprio antivírus.

3.6 Comparativo dos trabalhos relacionados

Cada trabalho relacionado mostrou-se eficiente em suas conclusões. Por meio de suas particularidades, foi possível compreender os problemas apresentados, enriquecendo o entendimento da proposta definida nesta monografia.

Os estudos 1, 2 e 3, intitulado e desenvolvido respectivamente de “Estudo da aplicação de estrutura *blockchain* com Proof of Stake para arquivamento de documentos com registros no tempo”, Corrêa (2017), “Prova de existência de arquivos digitais utilizando a tecnologia *blockchain* do protocolo Bitcoin”, Martins (2018), e “Assegurando a propriedade e imutabilidade de documentos digitais: uma prova de conceito utilizando *blockchain*”, Mendanha (2017), apresentaram uma proposta similar, na qual os autores desenvolveram uma aplicação para o registro e verificação de um arquivo digital na *blockchain*. A proposta de Mendanha (2017) se destaca dos demais autores, pois, ainda é possível transferir o documento para um endereço público. Cada trabalho apresentou diferentes características em relação ao desenvolvimento da aplicação, diversificando as técnicas e tecnologias utilizadas.

O estudo 4, de nome “A general *framework* for *blockchain* analytics”, desenvolvido por Bartoletti *et al.* (2017), utilizou dados de *blockchains*'s e de sites externos para cruzar análises, como por exemplo a tendência do valor de mercado de ativos digitais. Utilizou diferentes tecnologias de acordo com a *blockchain* analisada, Bitcoin e Ethereum.

O quinto trabalho “A Blockchain-based Framework for Detecting Malicious Mobile Applications in App Stores”, desenvolvido por Hodayoun *et al.* (2019), baseou sua aplicação desenvolvida em *blockchain* para a detecção de arquivos maliciosos em aplicativos disponibilizados por “*app's stores*”.

A Tabela 6 elenca o que foi utilizado em cada trabalho para a execução de sua respectiva proposta.

Tabela 6 - Comparativo dos trabalhos relacionados.

Áreas	Corrêa (2017)	Martins (2018)	Mendanha (2017)	Bartoletti et al. (2017)	Homayoun et al. (2019)
<i>Blockchain</i>	Privada	Bitcoin	Privada	Bitcoin, Ethereum	Privada
Consenso	<i>Proof of Stake</i>	<i>Proof of Work</i>	Não informado	<i>Proof of Stake, Proof of Work</i>	Não informado
API de Integração	<i>Hyperledger Fabric, Hyperledger Composer Playground</i>	BlockCypher	Não informado	BitcoinJ/Bitcoin Core, Parity/web3j	Não informado
Banco de Dados	Não informado	Não utilizou	BigchainDB	MongoDB, Mysql	Não informado
Serviço em Nuvem	IBM Cloud	Não utilizou	<i>Web Service Amazon</i>	Não utilizou	Não utilizou
Outras ferramentas	Docker e Virtual Machines	Biblioteca <i>pyBitcointools</i>	IPFS	DBMS	Android APK's

Fonte: Elaborado pelo autor (2019).

4 MATERIAS E MÉTODOS

Esta seção aborda os procedimentos metodológicos adotados e aplicados ao longo do trabalho. As técnicas e conceitos explanados no referencial teórico são aplicados na prática nas seções 4.1 e 4.2, demonstrando o desenvolvimento da aplicação proposta.

De acordo com Marconi e Lakatos (2003), os modelos científicos não são aplicados apenas em ciências, no entanto, nem todos os campos que utilizam esses modelos são ciências. Conclui-se que, o uso de métodos científicos que compõem um conjunto de atividades metódicas permite ao pesquisador alcançar seu objetivo com maior facilidade e clareza.

A abordagem aplicada é a Qualitativa, onde Gerhardt e Silveira (2009) afirmam que este tipo de pesquisa não busca representar números e sim, aprofundar o entendimento de um determinado assunto. De acordo com a proposta do trabalho, busca-se explicar o porquê de usar a tecnologia aqui explanada, trazendo fatos sobre os aspectos de segurança e imutabilidade da informação. Também, demonstra o passo a passo do que deve ser feito, utilizando-se ferramentas já existentes para o desenvolvimento dos códigos da aplicação *back-end*, alcançando assim, o objetivo principal do trabalho: a comprovação da existência e imutabilidade de um arquivo digital.

Para Gerhardt e Silveira (2009), a pesquisa Aplicada, usada neste trabalho, tem como propósito gerar conhecimentos para a resolução de um problema prático ou objetivo em específico, podendo envolver interesses locais ou verdades/fatos. Neste caso, através do entendimento da estrutura e

funcionamento da *blockchain*, chega-se à conclusão de que esta tecnologia pode ser utilizada para o registro de arquivos digitais com baixo custo e tempo, auxiliando na conquista do objetivo geral e objetivos específicos do trabalho.

Para alcançar os objetivos gerais, é utilizada como auxílio a pesquisa Exploratória. Segundo Marconi e Lakatos (2003) propõe-se a analisar os diferentes fenômenos e fatos relacionados ao trabalho, permitindo maior familiaridade aos assuntos abordados através de diversos procedimentos sistemáticos, como a coleta de dados e ajuda nas tomadas de decisões.

Para os procedimentos, é utilizado a pesquisa Bibliográfica, pois, o referencial teórico tem o propósito de dispor as informações de maneira que simplifique os objetivos estabelecidos no trabalho, estabelecendo uma ordem estrutural do assunto, partindo do conceito e técnicas de criptografia até as provas de consensos utilizadas em *blockchain* e a tecnologia IPFS. Conforme Marconi e Lakatos (2003), por meio da pesquisa bibliográfica, todo material publicado, indiferente o meio, tem o intuito de aproximar o pesquisador com o tema definido, buscando estimular soluções inovadoras para o problema de pesquisa apresentado. O conteúdo descrito não é uma simples repetição do que já foi mencionado, e sim, novas conclusões e perspectivas em paralelo ao que já está evidenciado.

4.1 Tecnologias

Nesta seção serão explanadas as tecnologias utilizadas no desenvolvimento da aplicação *back-end*. Todas as ferramentas descritas abaixo possuem como foco a *blockchain* da Ethereum.

4.1.1 Smart Contract

Um *smart contract* foi utilizado para realizar o gerenciamento das *hash's*. Optou-se por um contrato inteligente pois através dele é possível criar *deadlines*, fazer com que outros contratos interajam com ele e vice-versa, gerenciar os endereços validando por exemplo, que um endereço envie somente duas *hash's*

para o *smart contract*, além de possuir a característica de ser usado amplamente no desenvolvimento de aplicações descentralizadas.

4.1.1 Truffle

A ferramenta Truffle do *framework* Truffle Suíte auxilia no desenvolvimento de aplicações descentralizadas. Foi utilizada para testar, compilar e realizar o *deploy* do *smart contract* na *blockchain*. Optou-se por esta ferramenta, pois, seus comandos são de fácil compreensão e podem ser executados diretamente na linha de comando do terminal do Ubuntu.

4.1.2 Go Ethereum

Para efetuar transações na *blockchain* pública da Ethereum é preciso se comunicar com um nó da rede. A ferramenta Go Ethereum ou Geth realiza esta ponte, fazendo com que a máquina onde há este cliente instalado se passe por um nó. É possível tornar a máquina local um *miner node*, *full node* ou *light node*.

4.1.3 web3.js – Ethereum Javascript API

A ferramenta web3.js é uma coleção de bibliotecas que permite a interação com um nó local ou remoto da Ethereum diretamente em arquivos de extensão JS utilizando um canal de comunicação através de um provedor *Remote Procedure Call* (RPC) ou *Inter-process Communication* (IPC). Para trabalhar com esta ferramenta foi preciso estar conectado há um nó da rede Ethereum. Utilizou-se a API para desbloquear a carteira e realizar os registros na *blockchain*.

4.1.4 InterPlanetary File System - IPFS

A proposta deste trabalho é comprovar a existência e imutabilidade de um arquivo digital por meio do seu *hash*, portanto, esta ferramenta pode ser utilizada em conjunto ou em paralelo a *blockchain*. Um diferencial significativo entre usar a *blockchain* ou o IPFS é que além do IPFS gerar o *hash*, ele armazena também o conteúdo do arquivo digital, diferente da *blockchain*, onde apenas é armazenado informações do tipo texto, porém, o IPFS carece nos aspectos de

segurança fornecidos pela *blockchain*, como por exemplo: o consenso proporcionado pela cadeia de blocos.

4.2 Desenvolvimento

Nesta seção serão abordados os passos e métodos realizados no desenvolvimento da aplicação *back-end*. Inicialmente, tinha-se como objetivo utilizar a API BitcoinJ para comunicar-se com a *blockchain* da Bitcoin e ferramentas que trabalhassem com a linguagem Java. No entanto, ao decorrer do desenvolvimento do trabalho, por razões de conteúdos disponíveis na *web*, uma comunidade mais ativa e mais rapidez e menos taxas no registro das transações, optou-se por utilizar a *blockchain* da Ethereum com foco no uso de um *smart contract* e ferramentas na linguagem JavaScript e Solidity.

4.2.1 Configurando o ambiente de desenvolvimento

Primeiramente foi preciso configurar o ambiente de desenvolvimento. O sistema operacional utilizado foi o Ubuntu por possuir mais flexibilidade no uso e suporte das ferramentas manipuladas. Os códigos foram escritos no editor de texto VSCode onde é possível baixar extensões para dar *highlight* nas linguagens, neste caso, JavaScript e Solidity. O web3.js, IPFS e demais pacotes do node.js foram gerenciados pelo *Node Package Manager* (NPM). As ferramentas Truffle e Go Ethereum foram baixadas pelo *Advanced Packaging Tool* (APT) do Ubuntu.

4.2.2 Conectando na *blockchain* de teste da Ethereum

Antes de qualquer interação com a *blockchain*, foi preciso primeiro se comunicar com um nó da rede escolhida (mais detalhes sobre as redes na seção 4.2.3). O cliente Geth foi utilizado para realizar a integração com a *blockchain*. Para este trabalho, um *light node* já supre as necessidades, assim, não é necessário sincronizar com a rede realizando o *download* de toda a *blockchain*. Um *light node* realiza o *download* da *blockchain* a partir do momento em que a comunicação é criada/estabelecida entre o computador local e a rede. O trabalho

está delimitado a somente enviar a *hash* em uma transação, portanto, nenhum bloco será minerado, este papel é atribuído aos *miner nodes* da rede.

Após realizar a conexão com o canal de comunicação entre a máquina local e a *blockchain*, a carteira/endereço utilizado como endereço de saída nas transações a serem efetuadas foi criado. Uma senha foi informada no momento da criação da carteira. Todo endereço criado neste nó fica salvo em um diretório específico da máquina local (*keyfiles*), podendo ser transferidos a outros nós e vice-versa. Por padrão e motivos de segurança, os endereços deste nó ficam bloqueados, ou seja, não é possível utilizá-los até que sejam desbloqueados. O desbloqueio do endereço foi realizado por meio de uma função exclusiva que descriptografa o arquivo (*keyfile*) correspondente ao endereço manipulado. A carteira é bloqueada novamente quando o canal de comunicação é fechado, portanto, as carteiras/endereços devem ser sempre desbloqueadas a cada nova conexão.

A Figura 16 exibe o comando de conexão responsável por criar o canal de comunicação entre a máquina local e a *blockchain*. A rede informada é a de teste “--rinkeby”, o canal de comunicação é pelo tipo RPC “--rpc”, no “--rpcapi” é dito quais métodos podem ser utilizados a partir da comunicação estabelecida. Ao final do comando é informado que a máquina trabalhará como um *light node* “--syncmode=light”. Abaixo é exibido os dados da conexão. A cada bloco minerado na rede, uma nova linha de informação é exibida no terminal.

Figura 16 - Comando de conexão do cliente Geth.

```

rrs@rrsdesk: ~
File Edit View Search Terminal Help
rrs@rrsdesk:~$ geth --allow-insecure-unlock --rinkeby --rpc --rpcapi="admin,eth,net,web3,personal,
txpool" --syncmode=light
INFO [06-21|17:28:27.774] Starting Geth on Rinkeby testnet...
INFO [06-21|17:28:27.774] Dropping default light client cache      provided=1024 updated=128
INFO [06-21|17:28:27.775] Maximum peer count                ETH=0 LES=10 total=50
WARN [06-21|17:28:27.775] The flag --rpc is deprecated and will be removed in the future, please us
e --http
WARN [06-21|17:28:27.776] The flag --rpcapi is deprecated and will be removed in the future, please
use --http.api
INFO [06-21|17:28:27.776] Smartcard socket not found, disabling      err="stat /run/pcscd/pcscd.comm:
no such file or directory"
INFO [06-21|17:28:27.783] Starting peer-to-peer node                instance=Geth/v1.9.15-stable-0f7
7f34b/linux-amd64/go1.14.2
INFO [06-21|17:28:27.783] Allocated cache and file handles         database=/home/rrs/.ethereum/rin
keby/geth/lightchaindata cache=64.00MiB handles=2048
INFO [06-21|17:28:27.850] Allocated cache and file handles         database=/home/rrs/.ethereum/rin
keby/geth/lespay cache=16.00MiB handles=16
INFO [06-21|17:28:27.929] Persisted trie from memory database      nodes=355 size=50.43KiB time="75
6.147µs" gcnodes=0 gcsize=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [06-21|17:28:27.945] Initialised chain configuration          config="{ChainID: 4 Homestead: 1
DAO: <nil> DAOSupport: true EIP150: 2 EIP155: 3 EIP158: 3 Byzantium: 1035301 Constantinople: 36606
63 Petersburg: 4321234 Istanbul: 5435345, Muir Glacier: <nil>, YOLO v1: <nil>, Engine: clique}"
INFO [06-21|17:28:27.969] Added trusted checkpoint                block=6619135 hash="37dbc0...df9d6
1"
INFO [06-21|17:28:27.970] Loaded most recent local header         number=6697245 hash="187cd7...d158
a1" td=11953821 age=1d19h39m
INFO [06-21|17:28:27.973] Configured checkpoint registrar         address=0xebe8eFA441B9302A0d7eaE
Cc277c09d20D684540 signers=4 threshold=2
INFO [06-21|17:28:28.032] UDP listener up                         net=enode://4453534891c3734b0d9e
e63b6003fa544dbfd9314a3f639ee1bc580cb3b1d771dbd1a2c62d1ded5ac7acf8242eab1e8994ea2730516d998d0e7f21a
55fd75d3c@[::]:30303
WARN [06-21|17:28:28.034] Light client mode is an experimental feature
INFO [06-21|17:28:28.034] New local node record                   seq=32 id=f5101c7cf9239a99 ip=12
7.0.0.1 udp=30303 tcp=30303
INFO [06-21|17:28:28.035] Started P2P networking                  self=enode://4453534891c3734b0d9
ee63b6003fa544dbfd9314a3f639ee1bc580cb3b1d771dbd1a2c62d1ded5ac7acf8242eab1e8994ea2730516d998d0e7f21
a55fd75d3c@127.0.0.1:30303
INFO [06-21|17:28:28.036] IPC endpoint opened                     url=/home/rrs/.ethereum/rinkeby/
geth.ipc
INFO [06-21|17:28:28.037] HTTP endpoint opened                    url=http://127.0.0.1:8545/
cors= vhosts=localhost
INFO [06-21|17:28:28.042] Mapped network port                    proto=tcp extport=30303 intport=
30303 interface=NAT-PMP(192.168.31.1)
INFO [06-21|17:28:28.049] Mapped network port                    proto=udp extport=30303 intport=
30303 interface=NAT-PMP(192.168.31.1)
INFO [06-21|17:28:29.992] Block synchronisation started
INFO [06-21|17:28:36.821] Imported new block headers              count=192 elapsed=118.602ms numb
er=6697437 hash="e33cc7...3e99f2" age=1d18h51m
INFO [06-21|17:28:37.160] Imported new block headers              count=384 elapsed=121.943ms numb
er=6697821 hash="e058de...fd1fb1" age=1d17h15m

```

Fonte: Elaborado pelo autor (2020).

4.2.3 Desenvolvimento do *Smart Contract*

O próximo passo foi a construção do *smart contract* utilizado no gerenciamento das *hash's* (o DNA do arquivo). Para delimitar o escopo deste trabalho, o contrato possui somente dois requisitos:

1. Permitir a adição de um DNA extraído de um arquivo digital, indiferente de sua extensão, PDF, JPG, DOCX, etc;
2. Validar os DNA's já inseridos, bloqueando a inserção de duplicatas.

Com o código do contrato implementado, utilizou-se a ferramenta Truffle para realizar a sua compilação e *deploy* na *blockchain*. Foi preciso configurar a rede na qual o Truffle compila e faz o *deploy* do contrato, podendo ser qualquer *blockchain* pública, principal/testes ou privadas. Duas redes de testes públicas foram experimentadas, a “Ropsten” e “Rinkeby”.

No período do desenvolvimento, a rede Ropsten apresentou-se menos ativa em relação a Rinkeby. Muitas vezes o contrato não era enviado a *blockchain* (*deploy*) por não possuir nós ativos na rede e por este motivo, a rede Rinkeby foi adotada como a rede principal no desenvolvimento, a qual mostrou-se mais ativa. O ato de enviar o contrato pra *blockchain* é constituído por uma transação, ou seja, é preciso informar um endereço de saída. O endereço de entrada passa a ser o endereço gerado pelo *deploy* do contrato.

A rede de testes possui as mesmas regras da rede principal, portanto, é preciso que o endereço de saída possua *ethers* em sua conta para pagar o *gas* da execução do contrato, mesmo que na rede de testes não exista valor para estas moedas. *Ethers* podem ser adquiridas através do que é chamado de *Faucet*. São sites disponibilizados por membros da comunidade que distribuem *ethers* livremente entre as redes de testes. As moedas utilizadas para este trabalho foram solicitadas no site <https://faucet.rinkeby.io/>, onde é preciso criar um *tweet* via Twitter informando o endereço público da carteira, após, é necessário colar a URL do *tweet* no site e aguardar uns segundos. Neste *Faucet* pode-se utilizar também o Facebook. Com as *ethers* na carteira, o *deploy* pôde ser realizado.

É importante ressaltar que todo *deploy* gera um endereço para o contrato. Interações com este contrato são realizadas por meio do endereço gerado. Como o endereço é público, qualquer pessoa pode utilizá-lo, garantindo que as suas

condições e regras não serão alteradas pelo aspecto de imutabilidade da *blockchain*.

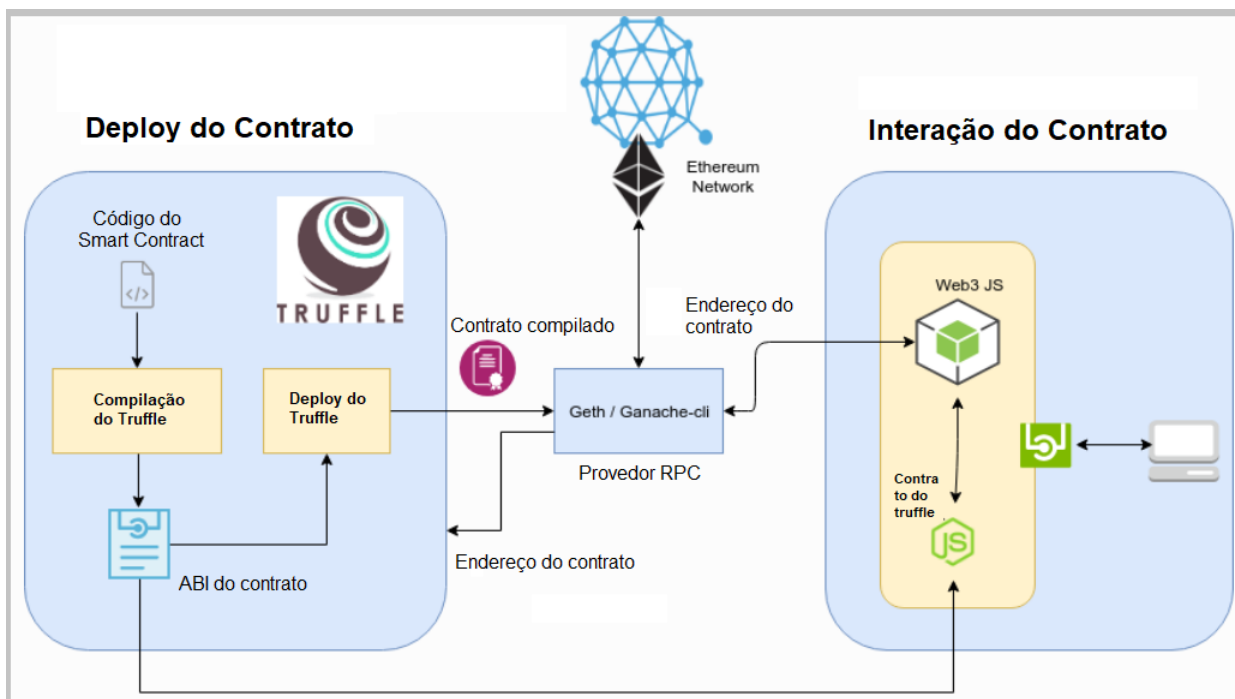
4.2.4 Desenvolvimento do *back-end*

O *back-end* tem como base principal a API *web3.js*. Duas classes em JavaScript foram criadas para servirem de auxílio na interação com o *smart contract* enviado a *blockchain*. Uma classe é responsável pelo provedor do *web3.js*, neste caso é a máquina local transformada em nó pelo cliente Geth. E a outra classe é encarregada de fornecer o *Application Binary Interface* (ABI) do *smart contract*. A ABI é um arquivo JSON com as informações e funções do contrato *deployed*.

Com as classes auxiliares desenvolvidas, em testes iniciais foi possível verificar que a conexão com a *blockchain* e interação com o contrato estava funcionando. Passou-se então para o próximo passo do trabalho, que consistiu no desenvolvimento das classes responsáveis por desbloquear o endereço de saída utilizado; extrair a *hash* do documento; enviar o arquivo para o IPFS e registrar a *hash* do arquivo na *blockchain*. Um método adicional foi desenvolvido para consultar rapidamente a *hash* que foi extraída do documento e armazenada na transação.

A Figura 17 demonstra os procedimentos descritos nesta seção. O cliente Geth é utilizado como provedor RPC para a compilação e *deploy* do Truffle e para a interação do *back-end* da API *web3.js* a partir do endereço do contrato. A ABI do contrato fornecida pelo Truffle é utilizada também pela *web3.js*.

Figura 17 – Diagrama de interação das ferramentas.



Fonte: Adaptada com base em HuronDocs (2020).

Pela expertise adquirida ao longo da execução do trabalho, recomenda-se sempre utilizar a mesma versão da ferramenta nos projetos a serem desenvolvidos, pois, a tecnologia está em constante evolução e há sempre novas versões sendo lançadas onde melhorias são implementadas, e funções existentes são modificadas.

5 TESTES E ANÁLISES DOS RESULTADOS

Dois cenários fictícios foram elaborados com o propósito de atestar e analisar os resultados. Os cenários criados baseiam-se em necessidades diárias de indivíduos que buscam certificar seus documentos, indiferente de sua origem, como prova de existência e autenticidade, evitando assim possíveis fraudes. Tratando-se de comprovar a veracidade de documentos, analogias ao ramo jurídico são facilmente estabelecidas, sendo comum a autenticação de documentos em órgãos competentes. Uma vez que a estrutura da rede *blockchain* fornece aspectos de segurança que garantem a existência e imutabilidade da informação, estes documentos podem ser certificados por ela. Há discussões sobre a *blockchain* substituir o poder de notoriedade destes órgãos, no entanto, ainda não há um parecer definitivo, já que os governos, profissionais da área e a população precisam estar alinhados sobre os benefícios e fragilidades da tecnologia. Os cenários idealizados e demais testes são apresentados nas seções abaixo.

5.1 Cenário 1 – Contrato de aluguel

Um arquivo digital, como por exemplo um PDF, que representa um contrato de aluguel estabelecido entre três partes, o locatário, locador e fiador. Como em um contrato nem tudo pode acontecer como o esperado, o não cumprimento das cláusulas podem acarretar em futuros desentendimentos, portanto, as partes interessadas do contrato podem registrar a *hash* do documento na *blockchain*, a fim de obter a comprovação de que o contrato

estabelecido existe e que foi acordado entre todas as partes. Por meio do *smart contract* desenvolvido e com a *hash* única do documento, basta que um único registro seja realizado na *blockchain*.

5.2 Cenário 2 – Propriedade intelectual

Uma propriedade intelectual, seja artística ou literária, como por exemplo, músicas, pinturas, artigos científicos e trabalhos acadêmicos representados em um arquivo digital. Não há um número fixo de partes envolvidas, a obra pode ser de uma única pessoa como também, de N colaboradores. Duas regras elaboradas pelo autor podem ser adotadas para manter o registro dos participantes da obra, são elas:

1. Para que a transação com a *hash* do documento seja realizada na *blockchain*, cada participante deve obrigatoriamente assiná-la com o seu endereço/chave pública. O registro dos participantes pode ser salvo em um campo específico de texto da transação, chamado de *Input Data* ou então pode-se utilizar *smart contracts* que permitem assinaturas de diferentes chaves, conhecidos como *Multi-Signatures Wallets*;
2. Ao invés de cada participante assinar, seus nomes e assinaturas podem constar no conteúdo do próprio documento, assim, basta que a transação seja assinada por apenas um endereço.

5.3 Prova de existência

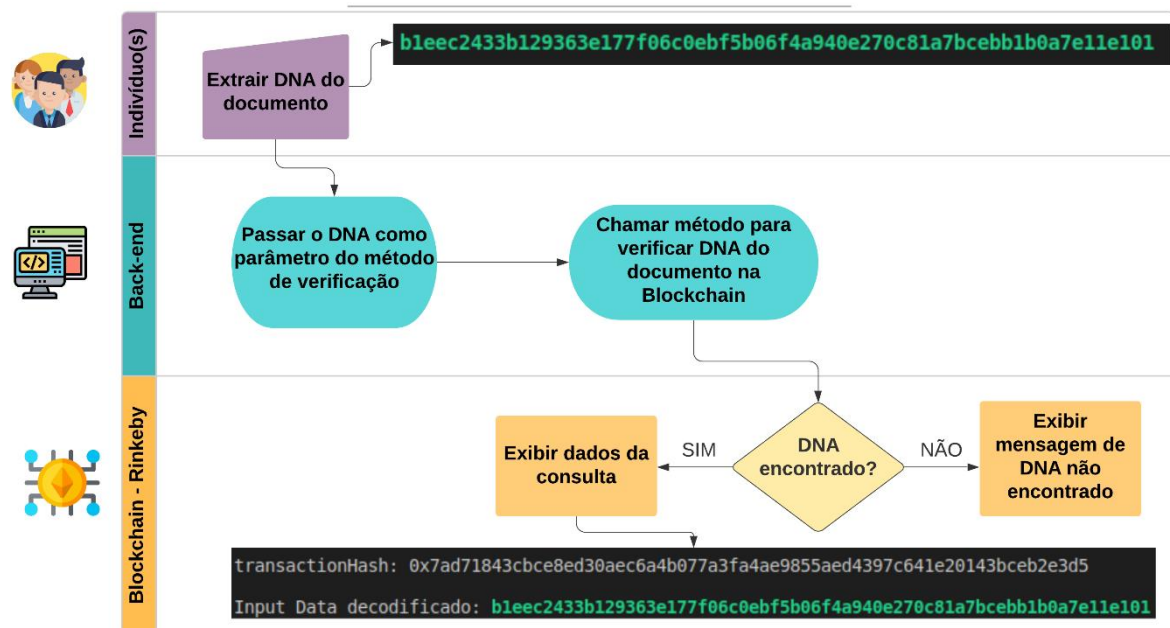
Os cenários foram criados para exemplificar casos onde a tecnologia *blockchain* pode ser utilizada. Ela não está restrita a atender apenas eles, podendo ser usada para comprovar a existência e imutabilidade de qualquer documento, seja qual for sua origem ou propósito. Baseando-se nos cenários, chega-se à conclusão de que a *blockchain* é sim um meio de comprovar a existência e imutabilidade de arquivos digitais, pois, quando submete-se o DNA extraído do arquivo a um registro da *blockchain*, tem-se a comprovação do

registro pela identificação da transação que pode ser distribuída para as partes envolvidas no processo.

A Figura 18 demonstra este processo. Inicialmente, os indivíduos submetem seu documento para extrair seu DNA, na sequência, o DNA é informado como parâmetro da função que realiza o registro na *blockchain*, após, tem-se o desbloqueio da conta utilizada e a criação do *link* IPFS. Por fim, é conferido se o DNA já está inserido na *blockchain*, se sim, a condição que bloqueia a inserção de DNA's iguais desenvolvida na seção 4.2.3 é atendida e exibe-se uma mensagem de alerta ao usuário. Se não, o registro é realizado e como saída é gerado as informações da transação criada. Uma das informações exibidas na saída é a identificação da transação (*transactionHash*) que é utilizada para comprovar a existência que o DNA do documento foi armazenado na rede.

se o DNA do documento, utilizou-se aqui o mesmo arquivo inserido na seção 5.3, na sequência este DNA é informado como parâmetro da função que consulta as transações na *blockchain* e ao final tem-se a validação se o DNA foi encontrado. Caso encontrado, exibe-se como saída a identificação da transação (*transactionHash*) utilizada como prova de existência. Caso não encontrado, mostra-se uma mensagem de aviso. Tomando como exemplo os documentos dos cenários das seções 5.1 e 5.2, uma vez que o DNA único do documento acordado entre as partes for inserido na *blockchain*, qualquer alteração posterior resultará em outro DNA.

Figura 21 – Verificação de um documento na *blockchain*.



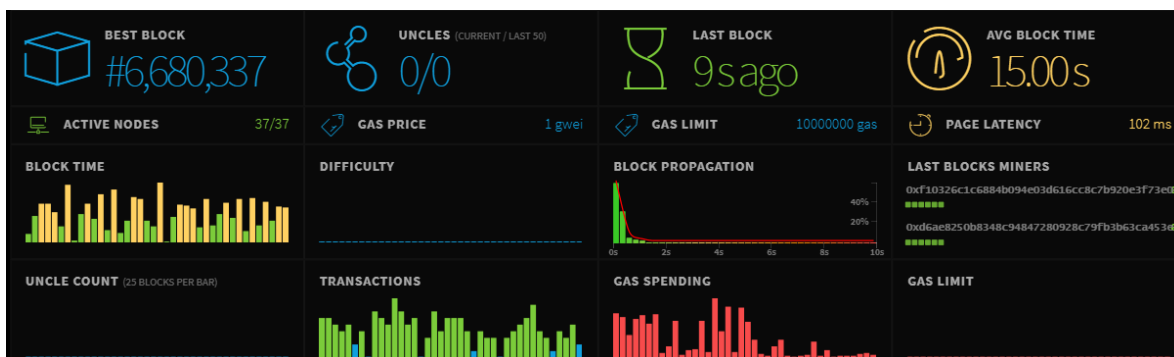
Fonte: Elaborado pelo autor (2020).

5.5 Custo e Tempo para certificar um documento

No período de desenvolvimento e testes do trabalho, o custo das transações em relação ao *smart contract* desenvolvido foram de frações de *ethers*, como mostra o campo *Transaction Fee* da Figura 19, 0,000133699 *ethers*, que convertidos para reais no dia 16 de junho de 2020 resultaram em R\$ 0,16 (16 centavos), pagando para cada *gas* (*gas price*) consumido o valor de 0.000000001 *ethers*. De acordo com o gráfico "Média de Gas pago por Ethereum"

(site oficial <https://etherscan.io/chart/gasprice>) da rede principal, a média de *ethers* pago por *gas* consumido no dia 16 de junho foi de 0.000000035 (quando este trabalho foi desenvolvido as redes de testes não possuíam gráficos), portanto, se este registro ocorresse na rede principal, o custo seria de R\$ 5,60. Um *smart contract* mais robusto, com mais condições e tratamentos ocasionaria em um custo maior. Na rede de teste Rinkeby, a média para certificar um documento foi de 15 segundos (tempo que um bloco é minerado). A Figura 22 exibe algumas estatísticas da rede de teste utilizada disponibilizadas pelo site <https://www.rinkeby.io/#stats>. Por meio deste site é possível visualizar o total de blocos minerados, o tempo médio de mineração de um bloco, preço e limite de *gas*, número de nós etc.

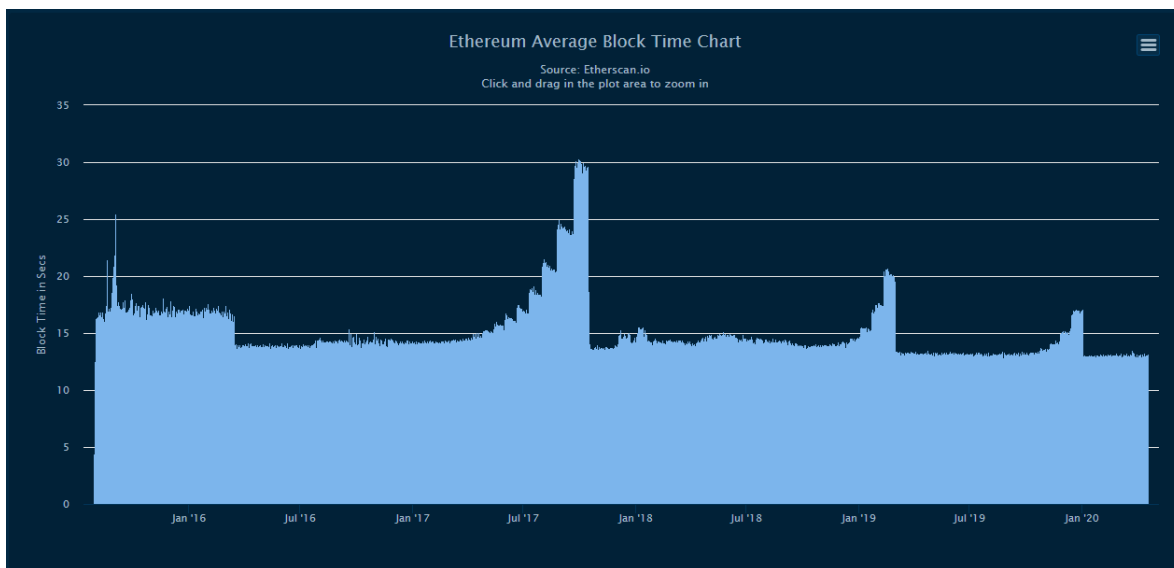
Figura 22 - Estatísticas da rede de teste Rinkeby.



Fonte: RinkeBy.io (2020).

Com todas as validações realizadas com sucesso, como teste final, realizou-se o *deploy* do *smart contract* na rede principal, onde *ethers* reais foram adquiridos para realizar algumas transações. O resultado atingido foi similar a rede de testes, uma vez que, a rede de teste e a rede principal possuem o mesmo funcionamento. Um exemplo é a Figura 23, onde pode-se visualizar que o tempo atual de mineração de um bloco na rede principal e de testes são similares.

Figura 23 - Tempo médio de segundos da mineração na rede principal Ethereum.



Fonte: Etherscan.io (2020).

Embora exista outras plataformas de *blockchain*, como a Bitcoin, a Ethereum mostrou-se eficaz nos testes realizados, por manter uma estrutura sólida na qual é possível auto executar códigos, sem a intervenção humana e pela rapidez no registro de transações, sendo muito mais ágil que a Bitcoin no aspecto bloco/transação minerado por segundo.

6 CONCLUSÕES

A presente monografia teve como objetivo a apresentação de um referencial teórico descrevendo os conceitos e técnicas utilizadas em paralelo a tecnologia *blockchain*, bem como, a explanação do funcionamento e estruturação da própria *blockchain*. De acordo com o problema de pesquisa levantado, o qual buscava aplicar uma solução para comprovar a existência de um arquivo digital e conforme o estudo construído, confirma-se que é possível certificar documentos via *blockchain*, como por exemplo, contratos, patentes e marcas, documentos pessoais, diplomas, entre outros. Por possuir o caráter de imutabilidade, toda informação armazenada em sua cadeia de blocos se torna irreversível, deixando de existir somente quando não houver mais cópias completas da cadeia de blocos pela rede ou Internet.

Baseando-se no problema de pesquisa e no referencial teórico produzido, conclui-se então que pela aplicação *back-end* desenvolvida, pelos aspectos de segurança da *blockchain* e da característica única do DNA do documento, é possível usar este conjunto de técnicas para ter-se um modelo de confiança para a comprovação de existência e imutabilidade de um arquivo digital. Para realizar a inserção do documento a cadeia de blocos, inicialmente é preciso ter o documento em mãos, na sequência, o DNA ou resumo *hash* deve ser extraído, posteriormente, com o pagamento da taxa da transação, o registro na *blockchain* é efetuado. O registro é realizado por meio de um método que permite a inclusão do DNA como conteúdo de um dos campos da transação. Para realizar a

verificação do arquivo, ou seja, conferir se o seu DNA está na *blockchain*, é preciso extrair o DNA do documento, e por meio do DNA, utilizar outro método que varre transação por transação, de cada bloco, e ao final, se encontrado, a identificação do registro é informado ao usuário.

Outra tecnologia usada em paralelo a *blockchain* é a IPFS, que mostrou-se eficaz no que foi proposta, salvando o conteúdo do documento em um *link* único distribuído pela Internet, já que, a *blockchain* atualmente armazena apenas texto. Embora não ser a solução de todos os problemas, a *blockchain* pode ser utilizada para descentralizar e automatizar processos de soluções de vários contextos que resultam na redução de custos, fraudes, riscos, otimização e velocidade dos processos, transparência, segurança e a eliminação de uma terceira parte responsável por administrar o processo, como por exemplo as instituições de pagamentos financeiros ou autenticação de documentos.

6.1 Trabalhos futuros

Tem-se como trabalhos futuros diversas ideias que surgiram ao longo do desenvolvimento da monografia. A iniciar pelo desenvolvimento de um *front-end* para que usuários possam interagir com a aplicação e ver com os próprios olhos a proposta aplicada e a praticidade que a *blockchain* fornece. Outra ideia é implementar melhorias no *smart contract*, como por exemplo, permitir que antes de um documento ser enviado a *blockchain*, confirmações possam ser feitas pelas partes envolvidas no processo através de seus e-mails, sem a necessidade de um *login*, disponibilizando o documento e o certificando em um consenso transparente e de comunicação clara, bem como, criar um repositório local de documentos via banco tradicional, mantendo assim a referência (ponteiro) de cada transação criada na *blockchain* para cada documento armazenado no banco, seria possível também enviar o documento para e-mails das partes interessadas. Com todo o escopo da aplicação desenvolvido, *back-end* e *front-end*, seria possível integrar o sistema com outras soluções, não apenas limitando-se ao armazenando dos DNA's de documentos.

REFERÊNCIAS

AGNER, Marco. **Bitcoin para Programadores**. Blockchain Hub, Instituto de Tecnologia & Sociedade do Rio – 2018. Disponível em: <https://itsrio.org/wp-content/uploads/2018/06/bitcoin-para-programadores.pdf>. Acesso em 30 out. 2019.

ALFAIFI, Reem. **Probabilistic cryptosystem with two pairs of private/public keys**. Novembro de 2017. ISBN: 978-1-5386-0872-2. Disponível em: <https://ieeexplore.ieee.org/document/8251927>. Acesso em 14 out. 2019.

ANTONOPOULOS, Andreas M. **Mastering bitcoin: unlocking digital cryptocurrencies**. O'Reilly Media, Inc., 2014. Disponível em: <https://unglueit-files.s3.amazonaws.com/ebf/05db7df4f31840f0a873d6ea14dcc28d.pdf>. Acesso em 08 out. 2019

ARAÚJO, Henrique Pereira; SILVA, Rebecca Bignardi Arambasic Rebelo. **A TECNOLOGIA DIGITAL BLOCKCHAIN: ANÁLISE EVOLUTIVA E PRAGMÁTICA**. Revista Refas - 2017. ISSN: 2359-182x. Disponível em: <http://www.revistarefas.com.br/index.php/RevFATECZS/article/view/98/118>. Acesso em 20 set. 2019.

BARTOLETTI, Massimo; LANDE, Stefano; POMPIANU, Livio; BRACCIALI, Andrea. **A general framework for blockchain analytics**. Middleware '17 18th International Middleware Conference Las Vegas, Nevada — Dezembro, 2017. Disponível em: <https://dl.acm.org/citation.cfm?id=3152831>. Acesso em 01 out. 2019.

Blockchain.Info – **Informações de uma transação inserida na blockchain**. Disponível em: <https://www.blockchain.com/pt/btc/tx/f8f18d6cb6e02e9d3477c3d04dda5eaa174d2cac6dbf15896dcfbc7fea58603f>. Acesso em 18 jun. 2020.

Blockchain.Info. **Explorador de blocos da Bitcoin**. Disponível em: <https://www.blockchain.com/pt/btc/block/00000000019d6689c085ae165831e934ff763ae46a2a6c172b3f1b60a8ce26f>. Acesso em 03 nov. 2019.

Blog EVALTEC. **Criptografia de dados e gerenciamento de chaves**. Disponível em: <https://www.evaltec.com.br/criptografia-de-dados-e-gerenciamento-de-chaves/>. Acesso em 19 out. 2019.

CARDOSO, Bruno ADV. **Blockchain – Como Funciona** – 2019. Disponível em: <https://brunocardosoadv.com/contratos-inteligentes/blockchain-como-funciona/>. Acesso em 14 out. 2019.

Bry. **O que é uma assinatura digital?** - 2019 Disponível em: <https://www.bry.com.br/blog/o-que-e-uma-assinatura-digital/>. Acesso em 11 out. 2019.

BUTERIN, Vitalik. **A Next-Generation Smart Contract and Decentralized Application Platform** – 2013. Disponível em: <https://github.com/ethereum/wiki/wiki/White-Paper>. Acesso em 27 abril 2020.

CARDOSO, João A. A.; PINTO Jefferson de Souza. **Blockchain e Smart Contracts: Um Estudo Sobre Soluções para Seguradoras**. CONGRESSO DE GESTÃO, NEGÓCIOS E TECNOLOGIA DA INFORMAÇÃO – 2018. Disponível em: <https://eventos.set.edu.br/index.php/congenti/article/view/9618/4325>. Acesso em 25 mar. 2020.

CAVALCANTE, André L.B. **Teoria dos Números e Criptografia**. UPIS Faculdades Integradas – Faculdade de Tecnologia – 2005. Disponível em: http://ssystem08.upis.br/repositorio/media/revistas/revista_informatica/Cavalcante_teorias_numeros_criptografia_2005_UPIS.pdf. Acesso em 19 out. 2019.

COINMARKETCAP. **Top 100 Cryptocurrencies by Market Capitalization**. Disponível em: <https://coinmarketcap.com/>. Acesso em 02 nov. 2019.

CORRÊA, Otávio Augusto. **ESTUDO DA APLICAÇÃO DE ESTRUTURA *BLOCKCHAIN* COM PROOF OF STAKE PARA ARQUIVAMENTO DE DOCUMENTOS COM REGISTRO NO TEMPO.** - UNIVERSIDADE FEDERAL DE SANTA CATARINA - 2017. Disponível em: <https://repositorio.ufsc.br/handle/123456789/187862>. Acesso em 20 set. 2019.

CORRÊA, Luís Felipe Brasil. **MONETIZOR: API DE PAGAMENTOS AUTOMATIZADOS UTILIZANDO *BLOCKCHAIN* PARA E-SPORTS.** - UNIVERSIDADE FEDERAL DE SANTA CATARINA – 2019. Disponível em: <https://repositorio.ufsc.br/bitstream/handle/123456789/202738/TCC.pdf?sequence=1&isAllowed=y>. Acesso em 25 mar. 2020.

Diário Oficial da União. **INSTRUÇÃO NORMATIVA Nº 1.888, DE 3 DE MAIO DE 2019.** Disponível em: <http://www.in.gov.br/web/dou/-/instru%C3%87%C3%83o-normativa-n%C2%BA-1.888-de-3-de-maio-de-2019-87070039>. Acesso em 19 out. 2019.

ELROM, Elad. **The *Blockchain* Developer - A Practical Guide for Designing, Implementing, Publishing, Testing, and Securing Distributed *Blockchain*-based Projects.** Apress, Julho de 2019. ISBN: 978-1-4842-4847-8. Disponível em: https://www.researchgate.net/publication/334652142_The_Blockchain_Developer_A_Practical_Guide_for_Designing_Implementing_Publishing_Testing_and_Securing_Distributed_Blockchain-based_Projects. Acesso em 3 out. 2019.

Ethereum.org. **Guia oficial para os desenvolvedores da rede Ethereum.** Disponível em: <https://ethereum.org/en/developers/>. Acesso em 10 jul. 2020.

Etherscan.io. **Explorador oficial da rede Ethereum.** Disponível em: <https://etherscan.io/>. Acesso em 16 jun. 2020.

FAULKNER, Jonh. **Getting Started with Cryptography in .NET** - 2016. Disponível em: <https://books.google.com.br/books?id=rH6CCwAAQBAJ&printsec=frontcover&hl=pt-BR#v=onepage&q&f=false>. Acesso em 9 out. 2019.

FERGUSON, Niels; SCHNEIER, Bruce; KOHNO, Tadayoshi. **The Context of Cryptography**. - Outubro de 2015. 10.1002/9781118722367.ch1. Disponível em: https://www.researchgate.net/publication/316368852_The_Context_of_Cryptography. Acesso em 8 out. 2019.

FRANÇA, Alexandre J. J. **Aplicação de Token-Curated Registry no combate às notícias falsas**. UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE – 2018. Disponível em: <https://monografias.ufrn.br/jspui/handle/123456789/8124>. Acesso em 25 mar. 2020.

GARCÍA, Andrea Del Nido. **Análisis e implementación del algoritmo de firma Digital de Curvas Elípticas Múltiples (MECDSA) para blockchain**. Universidad Politécnica de Madrid – 2019. Disponível em: http://oa.upm.es/55590/1/TFG_ANDREA_DEL_NIDO_GARCIA.pdf. Acesso em 20 out. 2019.

GERHARDT, Tatiana Engel; SILVEIRA, Denise Tolfo. **Métodos de Pesquisa**. UFRGS Editora – 2009. Disponível em: <http://www.ufrgs.br/cursopgdr/downloadsSerie/derad005.pdf>. Acesso em 24 mar. 2020.

GREVE, Fabíola; SAMPAIO, Leobino; ABIJAUDE, Jauberth; COUTINHO, Antonio; VALCY, Ítalo; QUEIROZ, Sílvio. **Blockchain e a Revolução do Consenso sob Demanda**. Universidade Federal de São Carlos, São Paulo – 2018. Disponível em: <http://www.sbrc2018.ufscar.br/wp-content/uploads/2018/04/Capitulo5.pdf>. Acesso em 14 out. 2019.

GRIBBLE, Steven; HALEVY Alon; IVES Zahary; RODRIG Maya; SUIU Dan. **What Can Peer-to-Peer Do for Databases, and Vice Versa?**. University of Washington Seattle, WA USA - 2001. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=14113B5200A708655B6F1534955119FB?doi=10.1.1.27.8638&rep=rep1&type=pdf>. Acesso em 14 out. 2019.

GRUBER, Damian; LI, WENTING; KARAME, Ghassan. **Unifying Lightweight Blockchain Client Implementations**. NEC Laboratories Europe - 2018. Disponível em: https://www.ndss-symposium.org/wp-content/uploads/2018/07/diss2018_10_Gruber_paper.pdf. Acesso em 26 out. 2019.

HART-DAVIS; Guys. **Mastering Microsoft VBA** - Wiley Publishing, Inc. 2006. ISBN: 978-0-7821-4436-5 Disponível em: <https://books.google.com.br/books?id=n-2NtSJNHBYC&pg=PA424&dq=what+is+digital+certificate&hl=pt-BR&sa=X&ved=0ahUKEwi0xNHnz5fIAhXhFLkGHUJICpkQ6AEIOzAC#v=onepage&q&f=false>. Acesso em 15 out. 2019.

HOMAYOUN, Sajad; DEHGHANTANHA, Ali; PARIZI, Reza M.; CHOO, Kim-Kwang Raymond. **A Blockchain-based Framework for Detecting Malicious Mobile Applications in App Stores**. 2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE), Edmonton, AB, Canada, 2019, pp. 1-4. Disponível em: <https://ieeexplore.ieee.org/document/8861782>. Acesso em 31 out. 2019.

HSBC. **Trust in Technology** – 2017. Disponível em: <https://www.hsbc.com/-/files/hsbc/media/media-release/2017/170609-updated-trust-in-technology-final-report.pdf>. Acesso em 3 out. 2019.

HuronDocs. **Ethereum Smart Contract Deployment Architecture** – 2020. Disponível em: <https://hurondocs.readthedocs.io/en/latest/ethereum-smartcontracts.html>. Acesso em 06 abril 2020.

IMED. **Uma análise da complexidade do algoritmo RSA implementado com o teste probabilístico de Miller-Rabin**. – 2019. ISSN: 2359-3539 Disponível em: <http://www.itl.gov.br/images/publicacoes/cartilhas/cartilhaentenda.pdf>. Acesso em 10 de out. 2019.

IMED. **REVISTA DE EMPREENDEDORISMO, INOVAÇÃO E TECNOLOGIA**. - 2017 Disponível em: <https://seer.imed.edu.br/index.php/revistasi/article/view/1639/1296>. Acesso em 18 jun. 2020.

KAUSHAL, Puneet Kumar; BAGGA Amandeep; SOBTI Rajeev. **Evolution of Bitcoin and security risk in Bitcoin wallets**. International Conference on Computer, Communications and Electronics (Comptelix), Julho de 2017. ISBN: 978-1-5090-4708-6. Disponível em: <https://ieeexplore.ieee.org/document/8003959>. Acesso em 13 out. 2019.

Lisk Academy. **Hashing** - 2019. Disponível em: <https://lisk.io/academy/blockchain-basics/how-does-blockchain-work/what-is-hashing>. Acesso em 23 out. 2019.

MACCORMICK, John. **Nine Algorithms That Changed the Future: The Ingenious Ideas That Drive Today's Computers**. Princeton University Press, 2012. Disponível em: <https://www.jstor.org/stable/j.ctt7t71s>. Acesso em 11 out. 2019.

MARCONI, Marina de Andrade; LAKATOS, Eva Maria. **Fundamentos de Metodologia Científica**. 5. ed. São Paulo: Atlas, 2003. 305 p. Acesso em 08 out. 2019.

MARTINS, Armando Nogueira; VAL, Eduardo Manual. **CRYPTOCURRENCY: NOTES ON ITS OPERATION AND INSTITUTIONAL PERSPECTIVES IN BRAZIL AND MERCOSUR**. RDIET, Brasília, V.11, nº1, p. 227 – 252, Jan-Jun, 2016. Disponível em: <https://portalrevistas.ucb.br/index.php/RDIET/article/download/6796/4559>. Acesso em 02 nov. 2019.

MARTINS, Thiago Fonseca. **Prova de existência de arquivos digitais utilizando a tecnologia *blockchain* do protocolo Bitcoin** - UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL - 2018. Disponível em: <https://www.lume.ufrgs.br/bitstream/handle/10183/177585/001065600.pdf?sequence=1>. Acesso em 19 set. 2019.

MENDANHA, Gabriel Oliveira. **ASSEGUANDO A PROPRIEDADE E IMUTABILIDADE DE DOCUMENTOS DIGITAIS: UMA PROVA DE CONCEITO UTILIZANDO *BLOCKCHAIN***. UNIVERSIDADE FEDERAL DO CEARÁ CAMPUS DE QUIXADÁ - 2017. Disponível em:

http://www.repositorio.ufc.br/bitstream/riufc/24738/1/2017_tcc_gomendanha.pdf.

Acesso em 19 set. 2019.

MORENO, Edward David; PEREIRA, Fabio Dacêncio; CHIARAMONTE, Rodolfo Barros. **Criptografia em Software e Hardware**. 2005. ISBN: 85-7522-069-1
Disponível em:

<http://www.martinsfontespaulista.com.br/anexos/produtos/capitulos/143116.pdf>.

Acesso em 20 out. 2019

NAKAMOTO, Satoshi. **Bitcoin: A Peer-to-Peer Eletronic Cash System**. Disponível em: <https://Bitcoin.org/Bitcoin.pdf>. Acesso em 25 set. 2019.

NYGAARD, Racin; MELING, Hein; JEHL, Leander. **Distributed Storage System based on Permissioned Blockchain** - 2019. ISBN: 978-1-4503-5933-7. Disponível em: <https://dl.acm.org/citation.cfm?id=3297544>. Acesso em 02 out. 2019

OLIVEIRA, Ronielton Rezende. **Criptografia simétrica e assimétrica-os principais algoritmos de cifragem**. Segurança Digital [Revista online], v. 31, p. 11-15, 2012. Disponível em:

<http://www.ronielton.eti.br/publicacoes/artigorevistasegurancadigital2012.pdf>.

Acesso em 9 out. 2019.

P.H. Alves, R. Laigner, R. Nasser, G. Robichez, H. Lopes, M. Kalinowski, “**Desmistificando Blockchain: Conceitos e Aplicações**”, Em: C. Maciel, J. Viterbo (Orgs). “Computação e Sociedade”, Sociedade Brasileira de Computação - 2018 Disponível em: <http://www-di.inf.puc-rio.br/~kalinowski/publications/AlvesLNRLK19.pdf>. Acesso em 12 out. 2019.

REVOREDO, Tatiana. **Blockchain: Tudo o Que Você Precisa Saber**. Amazon Digital Services LLC - KDP Print US – 2019. ISBN: 9781687405715.

Rinkeby.io. **Explorador oficial da rede de testes Rinkeby da Ethereum**. Disponível em: <https://www.rinkeby.io/#stats>. Acesso em 16 jun. 2020.

SACHCHIDANAND, Singh; NIRMALA Singh. **Blockchain: Future of financial and cyber security**. 2016 2nd International Conference on Contemporary Computing and Informatics (IC3I), Noida, 2016, pag. 463-467. Disponível em: <https://ieeexplore.ieee.org/abstract/document/7918009>. Acesso em 23 out. 2019.

SINGH, Mahima; GARG, Deepak. **Choosing Best Hashing Strategies and Hash Functions** - IEEE International Advance Computing Conference - 2009. ISBN: 978-1-4244-2927-1. Disponível em: <https://ieeexplore.ieee.org/document/4808979/citations?tabFilter=papers#citations>. Acesso em 10 out. 2019.

SOLIDITY. **Documentação oficial da linguagem Solidity**. Disponível em: <https://solidity.readthedocs.io/en/v0.4.21/introduction-to-smart-contracts.html?highlight=self%20destruction>. Acesso em 18 jun. 2020.

SOUSA, Suzane Carvalho. **USO DA BLOCKCHAIN DO ETHEREUM PARA A REALIZAÇÃO DE LEILÕES DE ENERGIA ELÉTRICA ATRAVÉS DA CRIAÇÃO DE FERRAMENTA WEB** - UNIVERSIDADE FEDERAL DO CEARÁ CENTRO DE TECNOLOGIA DEPARTAMENTO DE ENGENHARIA ELÉTRICA – 2019. Disponível em: http://www.repositorio.ufc.br/bitstream/riufc/49959/3/2019_tcc_scsousa.pdf. Acesso em 29 mar. 2020.

TradingView. **Gráfico para realizar análises da criptomoeda BTC** - 2019. Disponível em: <https://br.tradingview.com/chart/?symbol=BITSTAMP%3ABTCUSD>. Acesso em 11 out. 2019.

VOITECHEN, Dainara Aparecida. **ANÁLISE E COMPARAÇÃO DE ALGORITMOS PARA CRIPTOGRAFIA DE IMAGENS**. UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ, Ponta Grossa – 2015. Disponível em: http://repositorio.roca.utfpr.edu.br/jspui/bitstream/1/6438/1/PG_COADS_2015_2_03.pdf. Acesso em 19 out. 2019.

WikiWand. **Public key certificate** - 2019. Disponível em: https://www.wikiwand.com/en/Public_key_certificate. Acesso em 10 out. 2019.

ZOHAR, Aviv. **Recent Trends in Decentralized Cryptocurrencies (Invited Talk)**. In Proceedings of 49th Annual ACM SIGACT Symposium on the Theory of Computing, Montreal, Canada, Junho de 2017. ISBN: 978-1-4503-4528-6 Disponível em: <https://dl.acm.org/citation.cfm?id=3079074>. Acesso em 11 out. 2019.