UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Gabriel Estevam de Oliveira

Proposta de Carimbo do Tempo Descentralizado e com Acurácia utilizando contratos inteligentes na Blockchain Ethereum

Florianópolis
2022

Gabriel Estevam de Oliveira

Proposta de Carimbo do Tempo Descentralizado e com Acurácia utilizando contratos inteligentes na Blockchain Ethereum

Dissertação submetida ao Programa de Pós-Graduação em Ciência da Computação para a obtenção do título de mestre em Ciência da Computação.
Orientador: Prof. Jean E. Martina, Dr.
Coorientador: Prof. Martín A. G. Vigil, Dr.

Florianópolis
2022

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Estevam de Oliveira, Gabriel
Proposta de Carimbo do Tempo Descentralizado e com Acurácia utilizando contratos inteligentes na Blockchain Ethereum / Gabriel Estevam de Oliveira ; orientador, Jean Everson Martina, coorientador, Martín Augusto Gagliotti Vigil, 2022.
78 p.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós-Graduação em Ciência da Computação, Florianópolis, 2022.

Inclui referências.

1. Ciência da Computação. 2. Carimbo do tempo. 3. Blockchain. 4. Ethereum. 5. Contratos inteligentes. I. Everson Martina, Jean. II. Augusto Gagliotti Vigil, Martín. III. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Ciência da Computação. IV. Título.

Gabriel Estevam de Oliveira

Proposta de Carimbo do Tempo Descentralizado e com Acurácia utilizando contratos inteligentes na Blockchain Ethereum

O presente trabalho em nível de mestrado foi avaliado e aprovado por banca examinadora composta pelos seguintes membros:

Prof. Diogo Menezes Ferrazani Mattos, Dr.
Universidade Federal Fluminense

Prof. Roben Castagna Lunardi, Dr.
Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul

Profª. Thaís Bardini Idalino, Dra.
Universidade Federal de Santa Catarina

Certificamos que esta é a **versão original e final** do trabalho de conclusão que foi julgado adequado para obtenção do título de mestre em Ciência da Computação.

Documento assinado digitalmente
PATRICIA DELLA MEA PLENTZ
Data: 24/08/2022 11:54:29-0300
CPF: 959.243.700-97
Verifique as assinaturas em https://v.ufsc.br

Profª. Patricia Della Méa Plentz, Dra
Coordenadora do Programa

Documento assinado digitalmente
JEAN EVERSON MARTINA
Data: 24/08/2022 09:34:19-0300
CPF: 003.355.939-25
Verifique as assinaturas em https://v.ufsc.br

Prof. Jean E. Martina, Dr.
Orientador

Florianópolis, 2022.

AGRADECIMENTOS

Agradecimentos ao Prof. Jean Martina e ao Prof. Martín Vigil pelas valiosas orientações e a minha família por todo o suporte.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES).

RESUMO

Carimbos do tempo identificam que um determinado documento digital existia em um momento no passado. Isso pode ser utilizado, por exemplo, como comprovação de direitos autorais. Existem serviços que se propõem a produzir carimbos do tempo em blockchain, registrando dados através de transações na blockchain. Estas transações são inseridas em blocos pelos mineradores, os quais atribuem um timestamp para cada bloco. Desta forma, as transações são ancoradas temporalmente pelos timestamps dos blocos. Este trabalho concentra-se na acurácia destes carimbos do tempo, especificamente na blockchain Ethereum. Constatou-se que a acurácia média dos carimbos do tempo na blockchain principal da Ethereum são de algumas dezenas de segundos. Além disso, descobriu-se que os mineradores podem equivocadamente ancorar transações com tempo no passado, o que é uma falha grave para serviços de carimbos do tempo. Por isso, propôs-se um serviço descentralizado baseado em blockchain que utiliza contratos inteligentes e provedores de tempo distintos. Com o modelo proposto, a acurácia média dos carimbos do tempo é da ordem de milissegundos. Por fim, foi conduzida uma análise de custos para o modelo na blockchain Ethereum.

Palavras-chave: Carimbo do tempo. Blockchain. Ethereum. Contratos inteligentes. Acurácia.

ABSTRACT

Timestamps allow us to identify a date and time when a piece of data existed or an event took place. For example, we use timestamps to establish the date when we grant a patent. Services that offer trusted timestamps on the blockchain exist, where one creates a timestamp on a value by sending to the blockchain a transaction containing the value, which is eventually confirmed in a block a miner creates and timestamps. Our work focuses on the accuracy of such timestamps created on the Ethereum blockchain. It was estimated that the mean accuracy on the Ethereum Main network is a few tens of seconds. Surprisingly, we found out empirically that miners can backdate data using the timestamp on a block, which is a serious flaw in timestamping services. To address this problem, we propose a new decentralized timestamping service that combines smart contracts and distinct time providers. The service offers timestamps with higher accuracy of milliseconds. Finally, We conduct a cost analysis of our service, and we also discuss alternatives to cut costs.

Keywords: Timestamp. Blockchain. Ethereum. Smart contracts. Accuracy.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Provar a existência de um documento digital pode valer uma fortuna. Em março de 2021 um NFT (*non-fungible token*) foi vendido pelo artista digital Mike Winkelman por 69 milhões de dólares (IGNACIO, 2021). Os NFTs são ativos digitais registrados publicamente na Internet que identificam unicamente um documento digital. Suponha que uma arte digital "vaze" na Internet antes de ser devidamente registrada. A pessoa que conseguir fazer o registro primeiro terá a posse do ativo digital. Nesse contexto, os NFTs podem ser vistos como um registro de direitos autorais moderno.

Uma prova de existência identifica que um determinado documento digital existia em um momento no passado. Um NFT é uma forma de criar a prova a existência para um documento digital. Os NFTs foram padronizados pela ERC-721 (Ethereum Request for Comments) na blockchain Ethereum em 2018 com o objetivo de identificar bens únicos e que não podem ser divididos (REGNER; URBACH; SCHWEIZER, 2019). A ERC-721 permite implementar os NFTs através de contratos inteligentes, onde são registradas propriedades do bem e a quem pertence (ENTRIKEN et al., 2018). Dessa forma, além da prova de existência, os NFTs também permitem a atribuição de posse do ativo. Por conta disso, a ERC-721 prevê também a possibilidade de transferência de posse de NFTs.

Outra forma de criar uma prova de existência é através de carimbo do tempo. Ao invés de utilizar blockchain, obtém-se um certificado através de um serviço confiável de carimbo do tempo. Por isso, também pode ser chamado de carimbo do tempo confiável. Em um serviço confiável de carimbo do tempo, uma autoridade de carimbo do tempo atesta o recebimento de um documento em uma determinada data e hora. A autoridade de carimbo do tempo é uma terceira parte confiável que fornece data e hora para a existência de documentos. Quando um usuário solicita um carimbo do tempo a uma autoridade de carimbo do tempo, ela atribui uma data e hora ao documento, produz uma assinatura digital e retorna o documento assinado com o carimbo do tempo. A assinatura permite que a existência do documento possa ser posteriormente verificada. O carimbo do tempo confere as propriedades de integridade e tempestividade ao documento, assim como o NFT. A integridade garante que o documento não foi alterado e a tempestividade estabelece data e hora confiáveis para a existência do documento (MENEZES; OORSCHOT; VANSTONE, 2001).

O carimbo do tempo é regulamentado pela RFC3161 (Request for comments) (ZUCCHERATO et al., 2001), que estabelece padrões a serem seguidos pelas autoridades de carimbo do tempo. Esses padrões são comumente adotados por Infraestruturas de Chaves Públicas para o fornecimento de serviço confiável de carimbo do tempo, como a ICP-Brasil (2020b). O carimbo do tempo é aplicado por exemplo em registro de apólices de seguro, direitos autorais, ponto eletrônico e protocolos digitais (SANTIAGO, 2019). Um caso em que é indispensável o uso do carimbo do tempo é o diploma digital. Segundo Sergio Roberto de Lima e Silva Filho, da BRy Tecnologia (NSC, 2020), o carimbo do tempo comprova a data e hora que o diploma foi emitido.

A prova de existência de um documento pode ou não estar próxima do momento que o documento foi criado. É extremamente difícil saber o momento exato que um documento foi criado, no entanto após criar uma evidência de sua existência certamente esse também existe em qualquer momento posterior. O ancoramento temporal da prova de existência de um documento indica o momento da primeira evidência de sua existência, ou pelo menos, ao momento que o possuidor do documento desejou criar a prova de existência. Dessa forma, a acurácia da prova de existência é definida como o quão perto o ancoramento temporal está do momento que ela foi solicitada.

Algumas aplicações necessitam de ancoramento temporal com alta acurácia. Broby, Basu e Arulselvan (2019) apontam que leilões online e transações em mercado de ações necessitam de acurácia de no mínimo na ordem de milissegundos. Para o caso do carimbo do tempo, supondo que as autoridades de carimbo do tempo estejam sempre disponíveis e realizem o ancoramento prontamente, a acurácia está limitada à latência da rede de computadores. Mesmo no melhor cenário possível a latência em redes de computadores é da ordem de milissegundos e na prática os valores são da ordem de dezenas ou centenas de milissegundos (KUROSE, 2017; SINGLA et al., 2014). Sendo assim, qualquer forma de realizar prova de existência também estará limitada a mesma acurácia dos carimbos do tempo. Contudo, existem outros problemas associados aos serviços confiáveis de carimbo do tempo.

Segundo Oliveira (2020), os serviços confiáveis de carimbo do tempo são altamente centralizados e estão sujeitos a problemas de ponto único de falha, como ataques de negação de serviço. Com o intuito de criar serviços mais descentralizados e não depender de terceiras partes confiáveis, nos últimos anos surgiram vários trabalhos propondo a utilização de tecnologia blockchain para a realização de prova de existência.

A prova de existência baseada em NFTs utiliza tecnologia blockchain. Ainda existe a limitação da latência de rede, porém a blockchain dispõe de um modelo de arquitetura descentralizada. A blockchain foi inicialmente proposta como infraestrutura para a criptomoeda Bitcoin (NAKAMOTO, 2008). A blockchain do Bitcoin é um livro razão distribuído onde são registradas transações de criptomoedas. Tecnicamente, a blockchain é uma lista encadeada de blocos. Cada bloco contém um conjunto de transações e um cabeçalho com alguns dados como um ponteiro para o bloco anterior e um *timestamp* (marcação de tempo). O encadeamento dos blocos permite garantir a imutabilidade da cadeia. Se um bloco for modificado o ponteiro para o bloco se torna inconsistente, deixando a cadeia inválida do ponto de modificação em diante. Na blockchain do Bitcoin novos blocos são inseridos através do protocolo de prova de trabalho. O protocolo é executado por uma rede ponto-a-ponto de participantes chamados de mineradores. Os mineradores são responsáveis por validar transações e criar novos blocos. Não existem terceiras partes confiáveis ou pontos únicos de falha, os mineradores mantêm cópias da blockchain e são recompensados com criptomoedas.

Após o Bitcoin, surgiram outras plataformas baseadas em blockchain com outros protocolos de consenso e a possibilidade de criar redes privadas e permissionadas. Destacam-se

o Ethereum[1] e o Hyperledger Fabric[2] que introduziram o conceito de contratos inteligentes. Os contratos inteligentes são bytecodes (*scripts*) armazenados na blockchain. Quando invocados, estes scripts são executados pelos nós da rede, que atuam da forma prescrita no contrato e consentem sobre os resultados das operações (CHRISTIDIS; DEVETSIKIOTIS, 2016). Dessa forma, os contratos inteligentes permitem a criação de aplicações descentralizadas, como os NFTs.

O timestamp contido no cabeçalho dos blocos da blockchain indica a data e hora da criação do bloco. É justamente esse timestamp que permite o ancoramento temporal da prova de existência de um NFT. Além dos NFTs, outros trabalhos também propuseram a utilização do ancoramento temporal dos blocos para a criação de prova de existência, por exemplo os trabalhos de Gipp, Meuschke e Gernandt (2015) e Hepp et al. (2018). A utilização de blockchain para prova de existência tem como principal vantagem a não existência de uma terceira parte confiável. A confiança é dada pelo consenso da rede.

Na blockchain Ethereum, por exemplo, o timestamp do bloco é inserido pelo minerador e deve respeitar a um consenso estabelecido da seguinte forma. O timestamp do bloco deve ser maior que o timestamp do bloco anterior e deve ser menor que 15 minutos no futuro (ETHEREUM, 2021). Além disso, existem outros fatores que também podem impactar na acurácia da prova de existência: o tempo entre blocos e o fluxo de transações. O tempo entre blocos é definido como o intervalo de tempo entre a criação de blocos consecutivos, e impacta diretamente na precisão do ancoramento temporal. Na rede principal da blockchain Ethereum esse intervalo é em média aproximadamente 13 segundos (ETHERSCAN, 2021b). As transações quando são enviadas à rede entram em uma fila para serem executadas. Essa fila é ordenada pelas transações com as maiores taxas. As taxas são pagas para os mineradores pela execução das transações e mantenimento da rede. Como as transações com as maiores taxas são executadas primeiro, algumas transações podem permanecer na fila indefinidamente. Na rede principal da blockchain Ethereum, o tamanho médio da fila costuma ser de dezenas de milhares de transações pendentes, sendo que algumas delas podem permanecer indefinidamente da fila (ETHERSCAN, 2021e).

O restante do texto utiliza os termos prova de existência e carimbo do tempo como sinônimos, mesmo que o carimbo do tempo não esteja no padrão RFC3161. Quando for o caso de carimbo do tempo no padrão RFC3161 será mencionado.

1.1 PROBLEMA DE PESQUISA

A prova de existência em blockchain permite resolver os problemas causados pela centralização do carimbo do tempo RFC3161 fornecido por serviços de carimbo do tempo confiável. Contudo, conjectura-se que com essa prática a acurácia dos carimbos do tempo pode ser

[1] https://ethereum.org/en/

[2] https://www.hyperledger.org/use/fabric

superior a algumas dezenas de segundos. Além disso, pressupõe-se ser possível que carimbos do tempo em blockchain sofram atrasos ou avanços nas marcações de tempo.

Por esse motivo surge uma preocupação quanto a acurácia dos carimbos do tempo que utilizam blockchain. *É possível produzir carimbo do tempo descentralizado com acurácia utilizando blockchains?*. Essa é a pergunta de pesquisa do trabalho. Para responder essa pergunta outras secundárias devem ser feitas:

- Q1: Como são feitos os carimbos do tempo em blockchain?
- Q2: O que determina a acurácia de um carimbo do tempo em blockchain?
- Q3: Qual a acurácia dos carimbos do tempo em blockchain?
- Q4: É possível atrasar ou avançar a marcação de tempo de um carimbo do tempo em blockchain?
- Q5: Como é possível obter carimbos do tempo em blockchain com acurácia?

Ao longo deste trabalho buscou-se encontrar as respostas para cada uma dessas perguntas.

1.2 HIPÓTESE DE PESQUISA

A estimativa de que a acurácia dos carimbos do tempo em blockchain pode ser superior a algumas dezenas de segundos é baseada no tempo de confirmações das transações em blockchain públicas. Acredita-se que processar a solicitação de carimbo do tempo assim que ela é enviada à rede permitirá obter uma acurácia maior. Para essa suposição considera-se a utilização de contratos inteligentes. Portanto, têm-se as seguintes hipóteses:

- H1: Os carimbos do tempo em blockchain não garantem acurácia da ordem de segundos ou menores que isso.
- H2: Contratos inteligentes podem permitir atestar carimbo do tempo em blockchain de forma descentralizada e com maior acurácia.

Estas hipóteses estão interligadas com o objetivo do trabalho. O objetivo do trabalho busca demonstrar a veracidade das hipóteses apresentadas.

1.3 OBJETIVO GERAL

Avaliar a acurácia dos carimbos do tempo em blockchain e desenvolver um modelo com maior acurácia utilizando contratos inteligentes.

1.3.1 Objetivos Específicos

- Descobrir como são feitos os carimbo do tempo em blockchain.
- Determinar a acurácia dos carimbos do tempo em uma blockchain pública.
- Desenvolver um modelo de carimbo do tempo em blockchain com acurácia maior que os modelos encontrados.
- Determinar a acurácia e uma estimativa de custos para o modelo proposto.

1.4 JUSTIFICATIVA

Muitos trabalhos propuseram a utilização de blockchain para a criação de prova de existência. Contudo, esses trabalhos podem estar confiando deliberadamente nos carimbos do tempo providos pela blockchain ou não estão atentos à acurácia destes.

Como já mencionado anteriormente, estima-se que na blockchain Ethereum a acurácia dos carimbos do tempo pode chegar a algumas dezenas de segundos. Além disso, pressupõe-se que os carimbos do tempo realizados em blockchains podem sofrer avanços ou atrasos na marcação do tempo.

A utilização de blockchain é justificada pela descentralização frente à centralização dos serviços confiáveis de carimbo do tempo. Acredita-se que a tecnologia blockchain permitirá fornecer garantia de tempestividade aliado às vantagens de sistemas distribuídos. Carimbos do tempo descentralizados podem ser mais confiáveis do que carimbos do tempo padrão RFC3161 ao promover a auditabilidade pública e transparência.

1.5 DELIMITAÇÃO DO ESCOPO

Este trabalho propôs-se a pesquisar sobre a acurácia dos carimbo do tempo em blockchains públicas, com foco na blockchain Ethereum. São de interesse do trabalho propostas de ancoramento temporal para provas de existências descentralizadas. Não são de interesse trabalhos relacionados a protocolos de sincronização e sistemas de tempo real que confiam em terceiras partes para o fornecimento de tempo.

1.6 METODOLOGIA

A caracterização da pesquisa deste trabalho é de gênero empírico, com objetivo exploratório, abordagem quantitativa e natureza aplicada. A pesquisa seguiu procedimentos experimentais, com uso de hipóteses e coleta de dados. A fonte de informação da pesquisa foi pesquisa bibliográfica e coleta de dados em cenários reais por meio de medições.

Durante a realização do trabalho buscou-se alcançar os objetivos através de etapas bem definidas. Os estudos preliminares foram fundamentais para delimitação do trabalho e das demais etapas. As etapas são apresentadas a seguir, com uma breve descrição de cada uma delas:

- Estudos preliminares.

  Pesquisa bibliográfica exploratória. Delimitação do problema. Determinação das perguntas de pesquisa e hipóteses. Definição dos objetivos e métodos.

- Revisão sistemática da literatura para o tema tempestividade em blockchain.

  Determinação das *queries* e critérios de pesquisa. Busca de trabalhos relacionados nas principais bases de dados. Revisão e seleção dos trabalhos. Análise crítica e comparação dos trabalhos.

- Análise teórica da acurácia dos carimbos do tempo em blockchain.

  Estudo dos principais softwares utilizados na blockchain Ethereum. Análise de como são feitas as marcações de tempo nos blocos. Implicações do esquema de marcação sobre a acurácia teórica dos carimbos do tempo.

- Concepção de uma proposta de carimbo do tempo em blockchain.

  Definição dos participantes e do protocolo. Definição do esquema de carimbo do tempo e de um modelo de incentivo. Prototipação do modelo.

- Realização de experimentos para análise da acurácia dos carimbo do tempo em blockchain na prática.

  Definição do método de coleta de dados. Preparação do cenário. Execução dos experimentos. Coleta dos dados.

- Análise de resultados dos experimentos.

  Utilização de técnicas de estatística descritiva e inferencial sobre os dados do experimento. Discussão dos resultados dos experimentos.

1.7 CONTRIBUIÇÕES

Este trabalho contribuiu para o tema tempestividade em blockchain com uma análise da acurácia dos carimbos do tempo em blockchain. O trabalho inclui um estudo sobre as principais propostas de carimbo do tempo em blockchain e como é feito o ancoramento temporal na blockchain Ethereum. A análise teórica mostra uma estimativa da acurácia do ancoramento temporal e os experimentos mostram a acurácia na prática. O trabalho também contribui com um modelo de carimbo do tempo utilizando contratos inteligentes com melhor acurácia em relação aos modelos encontrados.

1.8 ESTRUTURA DO TRABALHO

No Capítulo 2 traz-se uma revisão bibliográfica sobre os principais assuntos relacionados a este trabalho: funções de hash criptográfico (Seção 2.1), prova de comprometimento (Seção 2.2), assinatura digital (Seção 2.3), certificados digitais (Seção 2.4), carimbos do tempo RFC3161 (Seção 2.5), blockchain (Seção 2.6) e contratos inteligentes (Seção 2.7).

No Capítulo 3 conduziu-se uma revisão sistemática da literatura com o tema tempestividade em blockchain para a busca de trabalhos relacionados a carimbo do tempo em blockchain. Estabeleceu-se o protocolo de busca utilizado e trouxeram-se os trabalhos selecionados pelos critérios estabelecidos. Foi identificado que a maioria das propostas de carimbo do tempo em blockchain utiliza o tempo dos blocos como arcoramento temporal e que não se preocupam devidamente com a acurácia do tempo.

No Capítulo 4 investiga-se a acurácia dos carimbos do tempo em blockchain que utilizam os tempos dos blocos para ancoramento temporal. Nesses casos, aponta-se que a acurácia depende de três fatores: tempo entre blocos (Seção 4.1), consenso de tempo (Seção 4.2) e fluxo de transações (Seção 4.3). Foi identificado que a acurácia tanto para os blocos como para as transações é de algumas dezenas de segundos. Contudo, para as transações a acurácia está sujeita a flutuações muito maiores. A acurácia das transações depende fortemente do volume de transações pendentes na rede e das taxas pagas pelas transações.

No Capítulo 5 apresenta-se um modelo de carimbo do tempo descentralizado utilizando contratos inteligentes. São definidos os participantes (Seção 5.1), o protocolo (Seção 5.2), a seleção do melhor carimbo do tempo e um mecanismo de incentivo (Seção 5.3). Por fim, apresenta-se um protótipo implementado do sistema (Seção 5.4).

No Capítulo 6 apresentam-se os experimentos realizados. Na Seção 6.1 define-se o ambiente de coleta de dados, incluindo os equipamentos utilizados para tal. Na Seção 6.2 investiga-se empiricamente a acurácia das transações na Ethereum Main Network, incluindo o tempo de propagação, janelas de tempo e confirmação das transações. Na Seção 6.3 conduziu-se uma avaliação do protótipo para a proposta apresentada, incluindo a adequação dos prazos do protocolo, das janelas de tempo e da estratégia de seleção.

No Capítulo 7 investigou-se a ocorrência dos tempos de confirmação negativos encontrados no experimento da Seção 6.2. São apresentados os motivos que podem ter levado a essas ocorrências. Os argumentos são fundamentados por um experimento anterior que analisou os tempos dos blocos na Ethereum Main Network.

No Capítulo 8 avaliou-se os custos dos carimbos do tempo obtidos com o modelo proposto. Analisaram-se as parcelas do custo computacional e das taxas e depósitos. Os custos foram parametrizados em função do *gas price*, câmbio, número de provedores, prêmio de risco, valor do depósito e número anual de carimbos do tempo fornecidos pelo serviço. É mostrado um exemplo com os valores para o dia 11 de junho de 2021 em função do número anual de carimbos do tempo fornecidos pelo serviço, identificando a fração de cada parcela do custo.

No Capítulo 9 são apresentadas discussões sobre os principais fatores relacionados ao

modelo de carimbo do tempo proposto. São discutidos itens como a latência de rede, tempo de confirmação, janela de tempo, acurácia, custos e depósitos. Adicionalmente, discute-se sobre as vantagens do modelo proposto. Por fim, no Capítulo 10 são apresentadas as considerações finais do trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são apresentados os conceitos fundamentais e as tecnologias utilizadas neste trabalho. Na Seção 2.1 apresentam-se as funções de hash criptográfico. Na Seção 2.2 define-se o esquema de prova de comprometimento baseado em função de hash. Nas Seções 2.3 e 2.4 apresentam-se assinatura digital e certificados digitais, respectivamente. Na Seção 2.5 introduz-se o carimbo do tempo no padrão RFC3161. E, nas Seções 2.6 e 2.7 trazem-se as definições de blockchain e contratos inteligentes, respectivamente.

2.1 FUNÇÕES DE HASH CRIPTOGRÁFICO

Uma função de hash criptográfico $H : \{0,1\}^* \rightarrow \{0,1\}^n$ mapeia um conjunto de bits $\{0,1\}^*$ de tamanho arbitrário para um conjunto de bits $\{0,1\}^n$ de tamanho n. As funções de hash criptográfico também são referenciadas algumas vezes apenas como *funções de hash*. As funções de hash são fundamentais para muitos protocolos criptográficos. Para os protocolos adotados neste trabalho, a função de hash deve cumprir alguns requisitos. O primeiro requisito é chamado de *resistência à colisão* e define que deve ser computacionalmente inviável encontrar $x \neq x'$ em que $H(x) = H(x')$. O segundo requisito é chamado de *resistência à pré-imagem* e define que deve ser difícil encontrar x dado $H(x)$. E o segundo requisito é chamado de *resistência à segunda pré-imagem* e define que dado um x deve ser difícil encontrar x' em que $H(x) = H(x')$. A saída de uma função de hash, $h = H(x)$, é comumente chamada de *hash criptográfico* ou apenas *hash* ("h é o hash de x"). Para mais detalhes sobre funções de hash pode-se consultar o livro de Stallings (2017). Uma escolha comum para função de hash é SHA-256 (STANDARDS; TECHNOLOGY, 2002) ou Keccak-256 (BERTONI et al., 2018), nas quais a saída tem tamanho $n = 256$.

2.2 PROVA DE COMPROMETIMENTO BASEADA EM FUNÇÃO DE HASH

Um esquema de prova de comprometimento baseado em função de hash é uma forma digital de esconder temporariamente um valor que não pode ser mudado (HALEVI; MICALI, 1996). O esquema consiste de duas fases. Na primeira fase, o *emissor* envia um valor bloqueado dentro de uma "caixa" para o *receptor*. Na segunda fase, o emissor envia a chave para o receptor abrir a caixa e descobrir o valor.

Uma prova de comprometimento pode ser construída de forma simples como a seguir. Dado $H(\cdot)$ uma função de hash criptográfico resistente a colisão. Na primeira fase, o emissor envia ao receptor um *commitment* $c = H(v||r)$ para um valor v, onde r é uma sequência de bits suficientemente longa e $v||r$ denota a concatenação binária de v e r. A função de r é dar mais variabilidade ao resultado, dificultando a descoberta do valor v. Isso porque se v for um timestamp, poderia-se encontrar v por tentativa e erro sabendo aproxidademente a data e hora do evento, caso não fosse utilizado o r. Na segunda fase, o emissor revela v e r para o receptor.

O receptor verifica se o emissor não mudou o valor de v computando $c' = H(v||r)$ e compara c' com c. Se c' e c forem iguais a prova de comprometimento é válida. Para mais informações sobre esquemas de prova de comprometimento, recomenda-se a leitura de (HALEVI; MICALI, 1996).

2.3 ASSINATURA DIGITAL

Os esquemas clássicos de assinaturas digitais consistem de três algoritmos. Especificamente, *GenerateKeys()*, *Sign(*d, S_k*)* e *Verify(*σ, P_k*)*. O algoritmo *GenerateKeys* cria uma chave privada e uma chave pública (S_k, P_k) para encriptação e decriptação de dados. O algoritmo *Sign* aceita como parâmetro um dado d e uma chave S_k. O dado d é comumente submetido a uma função de hash criptográfico $H(\cdot)$ e o resultado $h = H(d)$ é encriptado utilizando S_k e um algoritmo *Enc*, onde $sig = Enc_{S_k}(h)$. Dessa forma, a saída *sig* é concatenada com o dado original d gerando a assinatura $\sigma = (sig||d)$. Para verificar se σ é válido, deve-se executar o algoritmo *Verify*, que aceita como parâmetro σ e a chave P_k. Se a saída for verdadeira, a assinatura é válida, caso contrário, a assinatura é inválida. (STALLINGS, 2017)

As assinaturas digitais permitem verificar integridade de dados, isto é, permitem verificar se o dado original foi modificado. Além disso, as assinaturas digitais também permitem verificar autenticidade e não-repúdio, isto é, permitem identificar quem assinou um dado e impedem que o assinante negue a assinatura de um dado. Exemplos de esquemas de assinatura digital são o RSA e o ECDSA (STALLINGS, 2017).

2.4 CERTIFICADOS DIGITAIS

Certificados digitais permitem criar infraestruturas de chaves públicas. Uma terceira parte confiável chamada *autoridade certificadora* assina um certificado atestando que uma chave pública específica pertence a um determinado indivíduo. Ao confiar na autoridade certificadora pode-se confiar na autenticidade das assinaturas digitais emitidas pelos certificados derivados, a menos que o certificado tenha expirado ou a autoridade certificadora tenha sido comprometida. Considera-se que uma autoridade certificadora foi comprometida quando a sua chave privada foi descoberta. Os serviços confiáveis de carimbo do tempo (por exemplo, ICP-Brasil) utilizam infraestrutura de chaves públicas baseada em certificados digitais (ICP-BRASIL, 2020a).

2.5 CARIMBOS DO TEMPO RFC3161

Os carimbos do tempo são utilizados para identificar a data e hora em que um dado existia. Os carimbos do tempo emitidos por serviços confiáveis de carimbo do tempo são também chamados de carimbos do tempo confiáveis. Os carimbos do tempo confiáveis são compostos por uma tripla $H(v)$, t, σ, onde $H(\cdot)$ é uma função de hash resistente à colisão, $H(v)$ é o

hash do valor v a ser carimbado, t é a data e hora[1] quando v existia e σ é a assinatura digital da concatenação $H(v)||t$. A data e hora t, assim como a assinatura σ, são dadas por uma terceira parte confiável chamada Autoridade de Carimbo do Tempo (ACT). Confia-se que a ACT possui um relógio sincronizado com uma fonte de tempo oficial (por exemplo, o National Institute of Standards and Technology National Institute of Standards and Technology (2020)), e que utiliza a data e hora t do momento em que assina o carimbo do tempo (ICP-BRASIL, 2020b).

Qualquer carimbo do tempo emitido por uma ACT que foi comprometida não é mais confiável. Isso porque não é possível distinguir entre a) um carimbo do tempo válido emitido pela ACT antes de ter sido comprometida e b) um carimbo do tempo retroativo emitido pela ACT após ter sido comprometida. A RFC3171 (ZUCCHERATO et al., 2001) é o padrão para os carimbos do tempo emitidos pelos serviços confiáveis de carimbo do tempo, no qual outros protocolos de carimbo do tempo foram construídos sobre (BLAZIC; GONDROM; SALJIC, 2011; VIGIL et al., 2014).

2.6 BLOCKCHAIN

Uma blockchain é uma estrutura de dados distribuída composta por uma cadeia de blocos de dados, mantida por uma rede de nós utilizando um protocolo de consenso. Cada bloco pode incluir um ponteiro para o bloco anterior na cadeia, um *timestamp* e uma compilação de dados digitais. A Figura 1 ilustra simplificadamente uma estrutura blockchain. Esta tecnologia ganhou atenção com o Bitcoin (NAKAMOTO, 2008), onde implementa um livro razão distribuído para armazenar transações digitais de criptomoedas (transações de Bitcoins).

Para criar o encadeamento de dois blocos consecutivos B_i e B_{i-1}, onde $i >= 1$, cada bloco, exceto o primeiro (o bloco *gênesis*), armazena um ponteiro para o bloco anterior . O ponteiro é produzido com o hash criptográfico do bloco anterior. Para alterar um bloco no meio da cadeia (isso é, qualquer bloco exceto o bloco mais recente na cadeia), é necessário atualizar todos os blocos subsequentes para manter a consistência da cadeia. Em outras palavras, cada bloco B_i na cadeia permite verificar a integridade do bloco B_{i-1} e consequentemente de todos os blocos B_j com $j \leq i-1$. Além disso, o timestamp armazenado em cada bloco associado com o ponteiro de hash ajuda a estabelecer a ordem absoluta entre os blocos. Os timestamps dos blocos são discutidos mais detalhadamente no Capítulo 4.

Na proposta de Nakamoto (2008), a blockchain é compartilhada e replicada entre todos os nós de uma rede ponto-a-ponto, onde cada nó é conectado a um conjunto aleatório de nós (comumente chamados de vizinhos). Cada bloco em suas cadeias armazena uma compilação de dados digitais chamados de transações. Essas transações consistem em transferir Bitcoins de um nó para outro na rede. Cada nó é identificado por um endereço derivado de uma chave pública e controlado pela chave privada correspondente. A chave privada é utilizada para assinar

[1] As referências utilizam comumente o termo "data e hora", contudo a resolução de tempo não necessariamente é de 1 hora. O padrão mais comum, Unix Timestamp (https://www.unixtimestamp.com/), possui resolução de tempo de 1 segundo.

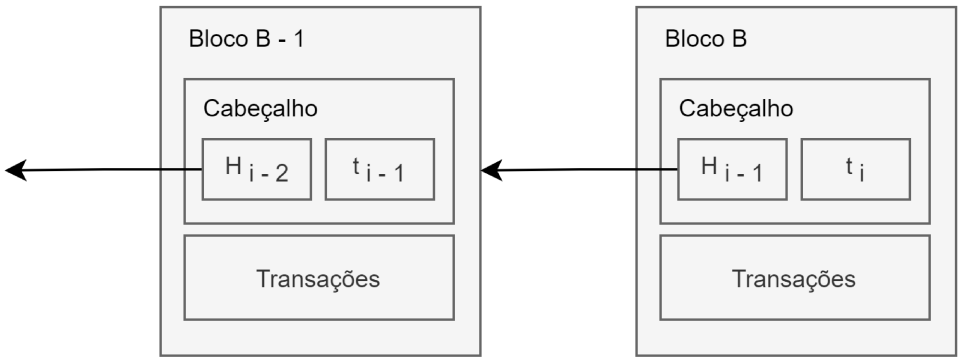

Figura 1 – Estrutura simplificada de uma blockchain.

as transações e a chave pública é utilizada para verificar as transações. É permitido aos nós possuírem múltiplos endereços, ou seja, diferentes pares de chaves. O conjunto de todas as chaves privadas de um nó é chamado de *carteira* de Bitcoin. Uma carteira de Bitcoin armazena virtualmente as moedas pertencentes ao nó.

Além disso, a blockchain armazena apenas transações validadas, ou seja, transações confirmadas pela maioria dos nós. Para armazenar uma nova transação na cadeia, o remetente (o nó que deseja transferir moedas para outro) envia uma requisição aos outros nós informando o endereço do destinatário e a quantidade de Bitcoins a ser transferido. Na sequência, os nós procedem com o processo de validação. Mais precisamente, eles verificam se a) o remetente dispõe de fundos suficientes para realizar a transação e b) se a assinatura digital da transação feita com a chave privada do remetente é válida.

Se a transação passar pelas verificações, um minerador pode adicioná-la a um novo bloco. No entanto, nesse sistema pode haver inconsistências entre as réplicas das blockchains. Dado que cada nó armazena uma cópia local da blockchain, é necessário um mecanismo no qual todos os nós concordam em quais blocos e transações são válidas.

Os nós participam de um algoritmo de consenso para manter as réplicas consistentes. No Bitcoin, os nós que desejam criar novos blocos são chamados de *mineradores*. Os mineradores montam uma compilação de transações enviadas em um novo bloco e executam o mecanismo de consenso chamado *Proof of Work* (PoW). O PoW consiste de um desafio criptográfico, onde os mineradores aplicam poder computacional para construir o novo bloco. Taxas são pagas aos mineradores para criarem novos blocos. Cada vez que um minerador cria corretamente um novo bloco, ele recebe alguns Bitcoins. Esse algoritmo de consenso pode ser entendido como uma competição baseada em rodadas. Cada vez que um bloco irá ser criado, os mineradores ingressam em uma nova rodada. Durante cada rodada, os mineradores coletam e verificam novas transações, definem o timestamp do próximo bloco que irão minerar e tentam resolver o desafio criptográfico. O desafio consiste em encontrar um *nonce* para o bloco que produza um hash a ele com determinada especificação, por exemplo, uma quantidade de zeros em sequência. Quando um minerador resolve o desafio criptográfico, ele envia o novo bloco

minerado para os outros nós e começa uma nova rodada de mineração. Após receber o novo bloco minerado, os outros mineradores também começam uma nova rodada. No entanto, os mineradores podem não estar sincronizados. Isso é, um minerador *A* pode estar em uma rodada diferente que o minerador *B*. Por exemplo, o minerador *A* pode estar tentando criar um novo bloco em uma cadeia não atualizada. Nesse caso, o algoritmo de consenso recomenda aos nós aceitarem a cadeia de blocos mais longa, em outras palavras, a cadeia que está na rodada mais recente.

Esse mecanismo atua também prevenindo os chamados *ataques Sybil* e *falhas Bizantinas*. Douceur (2002) explica que um ataque Sybil consiste de uma entidade maliciosa executando operações redundantemente em um sistema onde não existe uma autoridade confiável atestando correspondência entre entidades e identidades (por exemplo, usar várias identidades para votar múltiplas vezes em um protocolo de consenso baseado em votação). Com o PoW a participação está associada ao poder computacional, portanto não importa quantas identidades o indivíduo crie. Por sua vez, Mostéfaoui e Raynal (2016) descrevem uma falha Bizantina como a perda de um sistema devido a um de seus componentes desviar arbitrariamente do comportamento apropriado (por exemplo, um participante malicioso em uma rede ponto-a-ponto que envia mensagens conflitantes para impedir que os outros participantes cheguem a um consenso). Que nesse caso, são prevenidas pela adesão da cadeia mais longa, adotada pela maioria dos nós. Para mais informações sobre os diferentes algoritmos de consenso, recomenda-se a leitura do artigo de Alsunaidi e Alhaidari (2019).

O Bitcoin tornou-se rapidamente popular após o lançamento da rede em 2009 (Bitcoin Community, 2020), na qual inspirou dezenas de outras criptomoedas baseadas em blockchain nos anos seguintes (Mukhopadhyay et al., 2016). Em particular, a plataforma de blockchain Ethereum (WOOD et al., 2014). Similar ao Bitcoin, a plataforma Ethereum também provê meios para os nós transferirem fundos digitais, mas nesse caso, a criptomoeda é intitulada *Ether*. Além disso, o conceito central que difere a plataforma Ethereum da plataforma Bitcoin é a adoção dos contratos inteligentes, que serão vistos na próxima seção.

Ambas plataformas Bitcoin e Ethereum operam em redes públicas não permissionadas, onde cada nó integrante da rede pode acessar os dados armazenados nos blocos e propor novas transações. No entanto, outras plataformas blockchain, como Hyperledger Fabric (CACHIN et al., 2016), introduzem ambientes diferentes. Mais especificamente, no caso do Hyperledger Fabric, a blockchain é implantada em uma instância de uma rede privada permissionada. Portanto, as permissões de leitura e escrita podem ser regulamentadas por um nó ou um consórcio de nós. Para mais informações sobre as arquiteturas de blockchain, recomenda-se a leitura de (XU et al., 2017).

A blockchain tornou possível não apenas serviços financeiros descentralizados. Pesquisadores têm proposto aplicações de blockchain em diferentes áreas como educação (PALMA et al., 2019), combate a *fake news* (CHEN et al., 2020), armazenamento em nuvem (LI et al., 2020), computação em névoa (BANIATA; ANAQREH; KERTESZ, 2021), veículos inteligentes (OHAM et al., 2021) e Internet das Coisas (*IoT*) (ZHAO et al., 2020).

2.7 CONTRATOS INTELIGENTES

Uma das primeiras definições de contratos inteligentes foi apresentada por Szabo (1996) como cláusulas contratuais embarcadas em um fragmento de software. A Ethereum estendeu essa ideia generalizando para aplicações descentralizadas (DApps) executando na blockchain. Nesse contexto, qualquer nó na rede Ethereum pode criar contratos inteligentes. Os contratos inteligentes escritos através de instruções de alto-nível são traduzidos para uma codificação em bytes e armazenados como transações na blockchain Ethereum. Na blockchain Ethereum, um endereço também pode identificar contratos inteligentes. Para executar um contrato inteligente, é necessário enviar uma transação endereçada ao contrato específico definindo a função desejada e o conjunto de entradas. Isso porque a forma de interagir com o contrato é através da chamada de funções que são declaradas no código do contrato, sendo que cada função realiza uma rotina específica. Na sequência, os mineradores coletam essas transações e executam-a em um ambiente virtual local chamado Ethereum Virtual Machine (EVM).

Cada função em um contrato inteligente é representada como um conjunto de operações. Por exemplo, de acordo com o Apêndice G do *Ethereum Yellow Paper* (WOOD et al., 2014), para atribuir um valor a uma variável deve-se chamar a operação G_{sset}. Além disso, cada operação tem um custo associado em *gas*. A operação G_{sset} custa 20.000 *gas* para ser executada. Pode-se utilizar um compilador Ethereum para calcular o custo total das funções dos contratos inteligentes através da soma dos custos de todas as operações. Adicionalmente, o *gas* pode ser convertido para Ether e esse pode ser convertido, por exemplo, para dólar americano (U$) ou outra moeda fiduciária. Mais detalhes sobre os custos das operações são vistos no Capítulo 4.

3 ESTUDO DA LITERATURA

Este capítulo descreve um estudo da literatura sobre o tema tempestividade em blockchain. O estudo teve como objetivo obter conhecimento sobre os carimbos do tempo em blockchain e ajudar a responder as perguntas de pesquisa. A busca selecionou trabalhos com propostas de carimbo do tempo em blockchain ou que trazem alguma contribuição para o tema. Com a busca foi possível entender a relevância do tema e as dificuldades ainda encontradas.

3.1 REVISÃO SISTEMÁTICA

Uma revisão sistemática foi dirigida com seguinte pergunta de pesquisa: *Como são feitos os carimbos do tempo em blockchain?* O método utilizado foi o proposto por Kitchenham (2004).

A busca foi realizada nas principais bases de artigos científicos da área de ciência da computação. As palavras-chaves utilizadas foram *carimbo do tempo* e *blockchain*. Para carimbo do tempo foram aplicados na pesquisa os sinônimos ou palavras relacionadas:

proof of time, trusted time, timestamping, time stamping, time precision e *time accuracy*.

E para blockchain foram aplicadas na pesquisa as palavras relacionadas:

blockchain, bitcoin, ethereum e *hyperledger*.

A busca foi realizada em maio de 2021. Foram elegidos apenas artigos publicados em eventos e periódicos. O número de artigos encontrados em cada uma das bases de dados pode ser visto na Tabela 1. A última linha da tabela mostra o total de artigos não duplicados retornados. Com exceção da base de dados ArXiv, às demais foi possível exportar os metadados dos artigos e remover os artigos duplicados de forma automatizada. Para a base de dados ArXiv, por serem poucos artigos, o controle de duplicação foi feito manualmente. Não foram aplicados filtros de data de publicação. Todos os trabalhos foram publicados entre 2012 e 2021.

Base de Dados	Número de Artigos
ACM Digital Library	80
ArXiv	11
IEEE Xplore	14
Science Direct	991
Scopus	274
Springer Link	1649
Portal de Periódicos da Capes	197
Web of Science	34
Wiley	26
Total (não duplicados)	3087

Tabela 1 – Número de artigos retornados por cada base de dados na revisão sistemática da literatura.

Em um primeiro momento foram excluídos os artigos fora do escopo de carimbo do tempo e blockchain. A *string* de busca utilizada foi muito abrangente e trouxe muitos resultados. A maioria dos trabalhos foram desconsiderados apenas pelo título. Dos trabalhos que julgou-se que poderiam ter alguma contribuição para o tema foram, analisados também o resumo.

Observou-se que muitos trabalhos citam o termo carimbo do tempo ou termo relacionado, apesar de não fazer contribuição para a área. No conjunto de 3087 trabalhos retornados destacaram-se alguns temas. A Figura 2 mostra um gráfico com a quantidade de trabalhos, separados por tema, que ao menos mencionam carimbo do tempo.

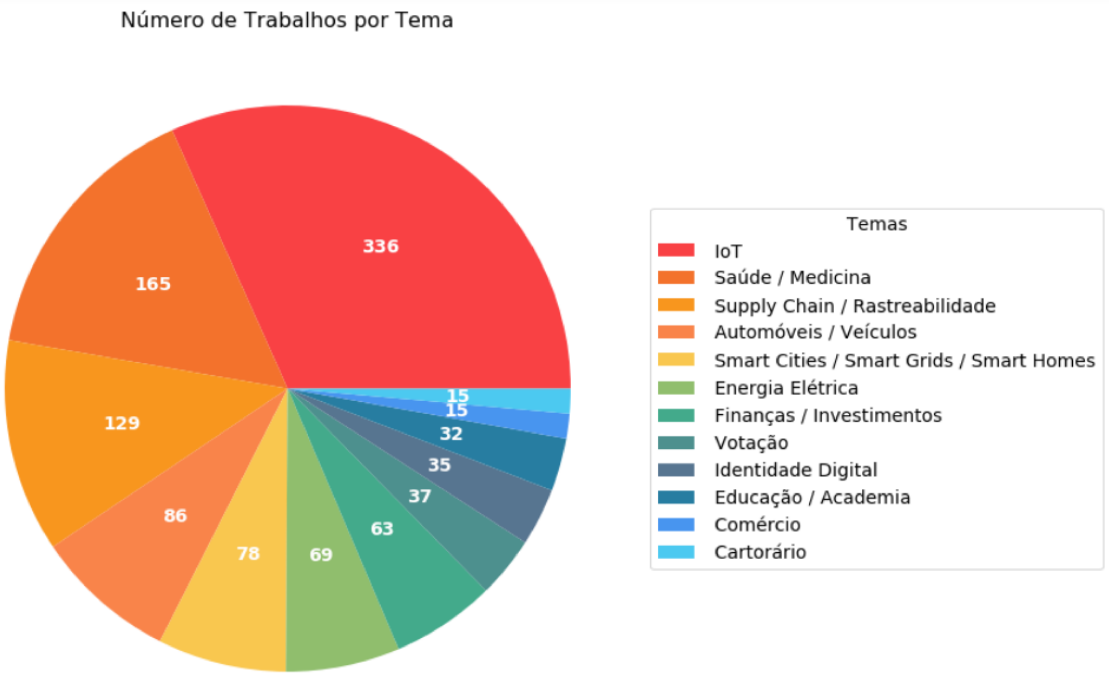

Figura 2 – Número de trabalhos nos temas que se destacaram na busca.

Em um segundo momento foram selecionados os artigos com aplicação mais direta do carimbo do tempo em blockchain. Foram escolhidos os trabalhos os quais considera-se que a acurácia do ancoramento do tempo é importante. A Figura 3 mostra um gráfico com a quantidade de trabalhos por tema.

A Figura 4 mostra um gráfico com o número de artigos publicados por ano dentro dos temas da Figura 3. Observa-se um número crescente nos últimos anos de trabalhos que potencialmente utilizam a propriedade de ancoramento temporal das blockchains. O ano de 2021 foi contabilizado até 5 de maio.

3.2 TRABALHOS RELACIONADOS

Nesta seção são trazidos mais detalhadamente os artigos com propostas de modelos de carimbo do tempo em blockchain. Também comparou-se os trabalhos relacionados com o modelo proposto neste trabalho. A Tabela 2 classifica os trabalhos relacionados em três categorias:

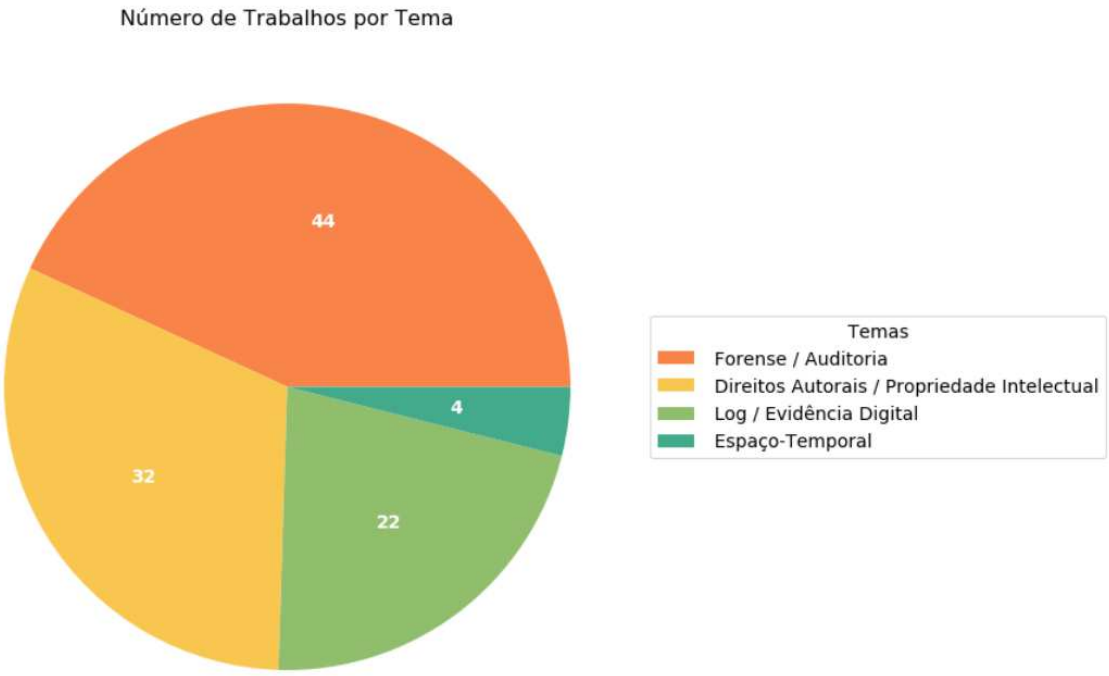

Figura 3 – Número de trabalhos nos temas que potencialmente usufruem do ancoramento temporal das blockchain.

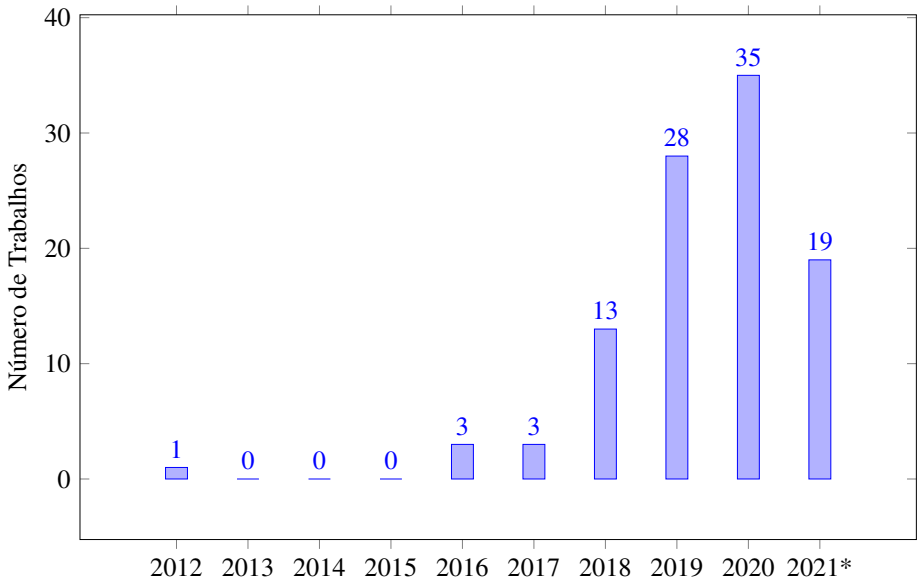

Figura 4 – Número de trabalhos publicados por ano que potencialmente usufruem do ancoramento temporal das blockchain.

1. Carimbos do tempo RFC3161 melhorados: propostas baseadas em blockchain que adicionam ou aperfeiçoam propriedades de segurança aos carimbos do tempo RFC3161;

2. Carimbos do tempo em blockchain: soluções que provém carimbos do tempo baseados no tempo que os mineradores atribuem aos blocos, e;

3. Carimbos do tempo em blockchain melhorados: soluções que adicionam algum grau de acurácia e segurança aos carimbos do tempo em blockchain.

	Acurácia				Propriedades	
Carimbos do tempo RFC3161 melhorados	*ms*	*s*	*m*	*h*	TPC	Plataforma
Stavrou e Voas (2017)			✓	✓	✓	Bitcoin*
Carimbos do tempo em blockchain	*ms*	*s*	*m*	*h*	TPC	Plataforma
Hepp et al. (2018)			✓	✓		Bitcoin*
Gipp, Meuschke e Gernandt (2015)			✓	✓		Bitcoin
Gipp et al. (2017)			✓	✓		Bitcoin
Kolydas (2019)		✓	✓			Ethereum
Jaquet-Chiffelle, Casey e Bourquenoud (2020)		✓	✓			DigiByte
Zhang et al. (2019b)		✓	✓			Ethereum
Zhang et al. (2019a)		✓	✓			Ethereum
Carimbos do tempo em blockchain melhorados	*ms*	*s*	*m*	*h*	TPC	Plataforma
Szalachowski (2018)			✓		✓	Bitcoin*
Guangkai, Chunpeng e Lu (2020)			✓		✓	Bitcoin
Zan e Xu (2020)		✓			✓	Permissionada
Landerreche, Schaffner e Stevens (2018)	-	-	-	-		-
Landerreche, Stevens e Schaffner (2020)	-	-	-	-		-
Este Trabalho (ESTEVAM et al., 2021)	✓					Ethereum*

Tabela 2 – Trabalhos relacionados comparados com este trabalho. Acurácia em: *ms* - milissegundos, *s* - segundos, *m* - minutos, *h* - horas. TPC (Terceira Parte Confiável).

A Tabela 2 também mostra a acurácia esperada para cada uma das propostas listadas, que depende fortemente da plataforma de blockchain escolhida. Por exemplo, propostas que utilizam a blockchain Bitcoin possuem acurácia estimada de minutos ou até horas. Por outro lado, na blockchain Ethereum, a acurácia estimada é de segundos no melhor cenário ou minutos, dependendo de fatores como as taxas oferecidas aos mineradores para confirmar as transações e a latência de rede. A acurácia na blockchain Ethereum será discutida mais detalhadamente no Capítulo 4.

Além disso, identifica-se se as propostas consideram ou não a participação de Terceiras Partes Confiáveis (TPC), o que inclui ACTs. Por último, é indicado com um asterisco (*) se os autores discutem o uso de diferentes plataformas em adição a plataforma utilizada na proposta original.

Foram encontrados alguns trabalhos relacionados à sincronização de relógios dos participantes em blockchain como (HARTL; ZSEBY; FABINI, 2019; FAN et al., 2018; FAN et al., 2019; BADERTSCHER et al., 2019). Nestes trabalhos, os autores afirmam que o problema da sincronização de tempo em um ambiente descentralizado é crítico e que pode influenciar na correta execução de protocolos de consenso. Esse tópico por si só é um desafio. Contudo, no contexto desse trabalho, manter os relógios sincronizados não é o suficiente. Apesar de a sincronização dos relógios ser essencial para obter carimbos do tempo com acurácia, não garante a correta utilização do tempo. Existe ainda um fator de confiança na aplicação desse tempo no carimbo do tempo. Por exemplo, um minerador, mesmo que com relógio sincronizado, pode aplicar um timestamp com tempo anterior ou posterior ao que está marcando o seu relógio. Portanto, o que se busca nesse contexto é como o tempo é aplicado no ancoramento temporal do carimbo do tempo.

Stavrou e Voas (2017) discutiram o problema de confiar em uma única ACT. Eles apon-

taram potenciais abusos que podem ser cometidos por uma entidade privada ou governamental e questionam sobre a validade de carimbos do tempo emitidos por uma ACT comprometida. Como uma alternativa, eles propõem um serviço de carimbo do tempo seguro combinando a confiança e acurácia de uma ACT centralizada com a natureza descentralizada das blockchains. Na proposta deles, uma ACT emite um carimbo do tempo RFC3161 com timestamp t e concatena t ao hash do bloco B_{i-1}, no qual os autores assumem ser o bloco mais recente na blockchain. Na sequência, a ACT publica essa concatenação na blockchain criando uma nova transação T_i, a qual espera-se ser confirmada no próximo bloco B_i. A concatenação produzida é registrada na blockchain e pode ser publicamente verificada. Dessa forma, pode-se inferir que t é posterior ao timestamp do bloco B_{i-1} e anterior ao timestamp do bloco B_i. No entanto, se a suposição dos autores de usar os blocos B_{i-1} e B_i não for verdade, então existe a possibilidade de ataques. Por exemplo, a ACT pode intencionalmente utilizar um bloco B_j mais antigo, assumindo $j < (i-1)$ como o bloco mais recente a fim de retroceder t. Além disso, considerando que é difícil garantir que a transação T_i contendo t é aceita no próximo bloco B_i, devido a latência e as taxas, T_i poderia ser aceita em um bloco futuro B_k, onde $k > i$. Portanto, a ACT pode atribuir a t qualquer valor entre B_{i-1} e B_k.

Os autores também discutem o cenário onde uma ACT é comprometida (por exemplo, se a chave privada da ACT é descoberta). Neste caso, a blockchain garante uma prova de existência contra violação dos carimbos do tempo criados antes do incidente, visto que pode-se distinguir entre carimbos do tempo criados antes e depois do incidente pelo timestamp do bloco.

A seguir são apresentados os trabalhos que contêm propostas de carimbo do tempo baseadas em blockchain. Ou seja, carimbos do tempo que utilizam o tempo que os mineradores atribuem aos blocos.

OriginStamp (HEPP et al., 2018; GIPP; MEUSCHKE; GERNANDT, 2015) é uma solução técnica que traz uma alternativa para os carimbos do tempo RFC3161. O serviço atua sobre o protocolo do Bitcoin. Para criar um carimbo do tempo, o usuário envia uma requisição para o serviço OriginStamp com o documento que deseja carimbar. Em seguida, o serviço computa o hash do documento enviado e o armazena para ser enviado para a blockchain. Uma vez por dia, o serviço coleta as requisições criadas nas últimas 24 horas e envia todas juntas à rede para reduzir os custos e o número de transações. Mais precisamente, eles armazenam os hashes coletados em uma árvore de Merkle balanceada (MERKLE, 1989). A raiz da árvore é inserida em uma nova transação T_i, que é eventualmente confirmada no bloco B_i. Se o usuário não estiver disposto a esperar até 24 horas no pior caso, também é possível o usuário solicitar um carimbo do tempo individual para um documento, que é encaminhado assim que o serviço recebe a solicitação. Neste caso, o serviço insere um único hash em uma nova transação e envia para a blockchain. Nesta proposta, a acurácia para os carimbos do tempo gerados dependem de a) se o serviço criou um carimbo do tempo único para a solicitação ou acumulou solicitações durante 24 horas antes de criar o carimbo do tempo, e b) a acurácia de tempo no ancoramento das transações na blockchain Bitcoin.

Nesse caso, os usuários confiam no serviço para enviar as transações assim que rece-

bem a solicitação de carimbo do tempo ou não mais do que 24 horas após a solicitação. Um exemplo de aplicação do OriginStamp é o CryptSubmit (GIPP et al., 2017), que traz rastreabilidade para sistemas de submissão de manuscritos. A ideia é ajudar os pesquisadores a se defenderem de potenciais vazamentos de dados ou fraudes dos revisores durante o processo de revisão em pares. Os autores propõem armazenar o hash dos manuscritos na blockchain para provar que o documento já existia em um certo ponto no tempo.

De forma semelhante, Kolydas (2019) e Jaquet-Chiffelle, Casey e Bourquenoud (2020) propõem o uso dos tempos dos blocos para o ancoramento dos carimbos do tempo nas blockchains Ethereum e DigiByte respectivamente. Nesses dois casos os carimbos são feitos diretamente para cada documento, sem a utilização de árvores de Merkle.

Chronos (ZHANG et al., 2019b) é um protocolo de carimbo do tempo que promete resolver a falta de acurácia dos esquemas de carimbos do tempo baseados em blockchain. A proposta adota a blockchain Ethereum para fazer o ancoramento temporal e utiliza um serviço de nuvem para fazer a coleta das requisições. O sistema carimba dados de terceiros assim que eles são registrados na nuvem. Existem três participantes na proposta:

1. Usuário: quem cria os arquivos e solicita os carimbos do tempo.
2. Servidor de Log Chronos: provê o serviço de carimbo do tempo e armazenamento para provedores de serviços em nuvem.
3. Auditor autenticado: entidade que pode acessar os dados armazenados no Servidor de Log Chronos e informar os carimbos do tempo para dados específicos.

Há quatro fases distintas no protocolo Chronos: *Setup*, *Store*, *TimeStamp* e *CheckStamp*. Na fase Setup são configurados os parâmetros de segurança, como as chaves dos usuários. Na fase Store os usuários geram e enviam novos arquivos assinados digitalmente para o Servidor de Log Chronos. Na fase TimeStamp, o Servidor de Log Chronos aplica uma função de hash na concatenação do arquivo, a assinatura e os últimos $\phi > 0$ blocos consecutivos da blockchain Ethereum (os autores recomendam $\phi = 12$ na rede Ethereum). O Chronos então gera uma transação com o hash computado e envia para a blockchain para ser armazenado em um bloco futuro B_i. Dessa forma, os autores afirmam que o Chronos pode atestar o intervalo em que um dado foi criado. Isto é, é possível inferir que a criação do dado é posterior ao bloco mais recente entre os últimos blocos e anterior ao bloco B_i. Além disso, os autores propõem o tempo limite de alguns minutos para esse intervalo. Na fase final, CheckStamp, quando solicitado, o auditor autenticado verifica o tempo quando o arquivo foi gerado e informa o timestamp.

Os autores estendem a proposta para o chamado Chronos+ (ZHANG et al., 2019a) para suportar carimbos do tempo em lotes, onde múltiplas requisições de diferentes usuários são tratadas juntas pelo serviço. Os autores também analisam as características dos serviços de dados em nuvem para determinar uma janela de tempo, a qual mede a praticidade dos serviços de carimbo do tempo em um cenário de nuvem. Em ambas as propostas, a acurácia dos carimbos

do tempo dependem fortemente do tempo decorrido entre o timestamp do bloco mais recente em ϕ e o timestamp do bloco B_i. Além disso, os usuários que solicitam os carimbos do tempo necessitam confiar no serviço Chronos.

Alguns outros pesquisadores realizaram propostas para melhorar os carimbos do tempo baseados em blockchain. Os trabalhos são apresentados a seguir.

Szalachowski (2018) propõem um mecanismo auxiliar baseado em serviços confiáveis de carimbo do tempo para aumentar a confiança dos carimbos do tempo na blockchain Bitcoin. A proposta não requer mudanças no protocolo original do Bitcoin. Existem dois tipos de participantes, uma ACT e Verificadores. Um Verificador é uma entidade que deseja verificar o tempo de um bloco em um futuro próximo. Cada participante na rede ponto-a-ponto pode atuar como um Verificador.

Primeiro, o Verificador observa a blockchain e seleciona o bloco mais recente B_{i-1}. O cabeçalho H_{i-1} do bloco é extraído e enviado para a ACT para gerar um carimbo do tempo t_{i-1}. Em seguida, o Verificador publica t_{i-1} enviando uma transação contendo t_{i-1} que será confirmada no próximo bloco B_i. Dessa forma, tem-se que B_i é posterior a t_{i-1}. O verificador então repete o procedimento gerando um carimbo do tempo t_i para o bloco B_i e publica t_i no próximo bloco B_{i+1}, criando a evidência que B_i foi criado entre os tempos de t_{i-1} e t_i. O autor sugere que a proposta possa ser estendida para suportar múltiplas ACTs, onde o Verificador poderia selecionar uma ACT diferente para gerar cada carimbo do tempo. A acurácia desta proposta depende do número de blocos entre t_{i-1} e t_i. No melhor cenário, o Verificador publica t_{i-1} no próximo bloco B_i e o mesmo ocorre para t_i. No entanto, não se pode garantir que isso sempre ocorra pois depende da confirmação das transações pelos mineradores, que normalmente priorizam as transações com as taxas mais altas. Além disso, nada pode-se dizer sobre o tempo que as transações em B_i foram criadas ou vistas pela primeira vez.

Seguindo a mesma linha, Guangkai, Chunpeng e Lu (2020) relatam que os tempos dos blocos na blockchain do Bitcoin podem ser manipulados em horas. Para mitigar esse problema, o trabalho sugere a utilização de ACTs para fornecer tempo com acurácia. Quando um nó está pronto para começar a mineração, envia para a ACT o número do bloco. A ACT retorna um carimbo do tempo assinado, com o número do bloco e o timestamp. O timestamp retornado pela ACT é inserido no bloco, para então ser minerado. Dessa forma, um verificador pode posteriormente consultar a validade do timestamp consultando a ACT. Essa abordagem permite atingir uma acurácia para o tempo dos blocos igual ao próprio tempo entre blocos, aproximadamente 10 minutos. Segundo o Autor, o *backdating* dos blocos é evitado auditando os carimbos na ACT.

Zan e Xu (2020) apresentam uma proposta com uma fonte confiável de tempo para sincronização do tempo em um consórcio blockchain permissionado. Por um lado abre mão da confiança distribuída inserindo terceiras partes confiáveis, mas por outro lado permite fornecer acurácia para uma série de aplicações *time-sensitive*.

Em um trabalho mais teórico, Landerreche, Schaffner e Stevens (2018) propuseram um serviço externo baseado em prova de trabalho sequencial. Teoricamente, a ideia pode ser utili-

zada por qualquer plataforma blockchain a fim de criar provas verificáveis criptograficamente para identificar a idade de registros.

Novamente, Landerreche, Stevens e Schaffner (2020) propõem um protocolo de carimbo do tempo baseado em *verifiable delay funcions*. O protocolo utiliza a técnica de computação-sequencial para prova de *elapsed time*. As *verifiable delay functions* permitem provar que um tempo foi despendido em computação não-paralelizável para o cálculo de um resultado mas que pode ser rapidamente verificável. O objetivo é trazer uma alternativa para o proof-of-work com menor custo e mesma garantia de robustez. Contudo, o trabalho é apenas teórico e não apresenta perspectivas de acurácia.

Ainda outros trabalhos estão relacionados a carimbo do tempo, mas não são propostas de modelo ou aplicações diretas. Lesk (2015) foi um dos primeiros a apresentar explicitamente a blockchain como um mecanismo de carimbo do tempo. Abid, Cheikhrouhou e Jmaiel (2020) desenvolveram um framework para a utilização de restrições temporais em contratos inteligentes na blockchain do Ethereum. Contudo, a fonte de tempo utilizada nas operações é o próprio tempo dos blocos. Nenhuma preocupação com a acurácia é levada em consideração. Por fim, Abadi et al. (2020) apresentam uma análise formal das primitivas de carimbo do tempo no contexto de blockchains através de um framework *Universally Composable* (UC). O trabalho examina as garantias contra ataques de *postdating* e *backdating*. Contudo, não considera a acurácia dos carimbos do tempo.

A proposta do presente trabalho (última linha da Tabela 2) (ESTEVAM et al., 2021) tem o intuito de melhorar os carimbos do tempo baseados em blockchain. No entanto, difere dos trabalhos relacionados apresentados acima por não confiar no tempo que os mineradores atribuem aos blocos, tampouco em Terceiras Partes Confiáveis. Ao invés disso, propõe-se combinar contratos inteligentes com diferentes fontes de tempo para prover carimbo do tempo exclusivamente para um dado. Na proposta, os carimbos do tempo são criados individualmente para cada transação. Os tempos são fornecidos por nós especiais da rede que fazem testemunhos do momento que viram as transações. Isso permite atingir uma acurácia da ordem de milissegundos, porque está limitada apenas à latência da rede.

4 ACURÁCIA DOS CARIMBOS DO TEMPO EM BLOCKCHAIN

A acurácia de um carimbo do tempo é definida pela diferença entre o momento da solicitação e o ancoramento temporal atribuído à ela. Este trabalho tem como foco os carimbos do tempo em blockchain que utilizam o timestamp dos blocos como ancoramento temporal, aqui chamado de *tempo do bloco*. Neste modelo, a acurácia é a diferença entre o momento que a transação é lançada na rede e o tempo do bloco em que a transação foi inserida. Analisou-se a acurácia na blockchain Ethereum, pois é a blockchain pública com maior valor de mercado entre as blockchains com suporte a contratos inteligentes (COINMARKETCAP, 2021).

Existem três fatores que influenciam a acurácia dos carimbos do tempo na blockchain Ethereum. São eles: tempo entre blocos (Seção 4.1), consenso de tempo (Seção 4.2) e fluxo de transações (Seção 4.3).

4.1 TEMPO ENTRE BLOCOS

O tempo entre blocos é definido como o intervalo de tempo entre a criação de blocos consecutivos. Na blockchain Ethereum um novo bloco é criado aproximadamente a cada 13 segundos em média (ETHERSCAN, 2021b). Isso determina a precisão da marcação de tempo dos blocos. A precisão é definida como o menor intervalo entre marcações de tempo, que também pode ser chamado de resolução de tempo. A precisão de tempo é limitada inferiormente pela unidade de tempo utilizada. O padrão utilizado pelo Ethereum é o Unix Timestamp[1], que conta o tempo em segundos desde 1º de janeiro de 1970. Contudo, por mais que a unidade de tempo do padrão adotado seja de segundos, a precisão do ancoramento temporal é limitada pelo tempo entre blocos porque não é possível obter uma marcação com tempo entre um bloco e outro.

Para ajudar a entender a precisão de tempo das transações pode-se imaginar o seguinte experimento mental. Bob está treinando para as olimpíadas na modalidade 100 metros rasos e pediu a ajuda da Alice para cronometrar o tempo dele. Nesta modalidade, os tempos costumam ser decididos na casa de décimos de segundos. Contudo, Bob e Alice possuem um cronômetro que marca apenas segundos inteiros. Mesmo assim eles decidiram anotar os tempos. A cada passada que Bob realiza, Alice marca o tempo que ele levou para percorrer a distância. Supõem-se que Bob não mantém uma performance constante e existe igual probabilidade de atingir qualquer tempo entre um intervalo mínimo e máximo, ou seja, a distribuição das amostras é uniforme dentro de um intervalo. Vendo os tempos anotados, Bob ficou chateado que não foi possível marcar os tempos com as casas decimais e quis saber qual o erro relacionado às marcações de tempo. Sabe-se que todas as marcações foram atribuídas ao segundo inteiro ligeiramente menor, algumas amostras tiveram erros de 0,01 segundos e outras 0,99 segundos. Como a distribuição das amostras é uniforme, a média dos erros é de 0,5 segundos. Ou seja,

[1] https://www.unixtimestamp.com/

a acurácia média é de 0,5 segundos, metade da precisão da marcação de tempo. Isso vale para qualquer evento com distribuição uniforme.

Atualmente, na blockchain Ethereum são confirmadas mais de um milhão de transações por dia (ETHERSCAN, 2021c). Isso corresponde a mais de 10 transações por segundo, cada bloco possui algumas centenas de transações. Portanto, podemos assumir que existe igual probabilidade de chegar novas transações a qualquer momento, ou seja, a distribuição de transações é uniforme. Se todas as transações quando lançadas na rede fossem inseridas no próximo bloco a ser criado então a acurácia média das transações seria de aproximadamente 6,5 segundos, metade da precisão de tempo dos blocos. A Figura 5 ilustra esse cenário. As transações recebidas entre t_{i-1} e t_i são inseridas no bloco B_i e o bloco B_i é criado no tempo t_i. O erro médio para o ancoramento das transações é $(t_{i-1} + t_i)/2$. Esse é o limite inferior da acurácia para o ancoramento temporal na blockchain Ethereum. Se esse fosse o único fator a influenciar a acurácia então poderíamos garantir que essa seria a acurácia média. Contudo há outros fatores que veremos a seguir.

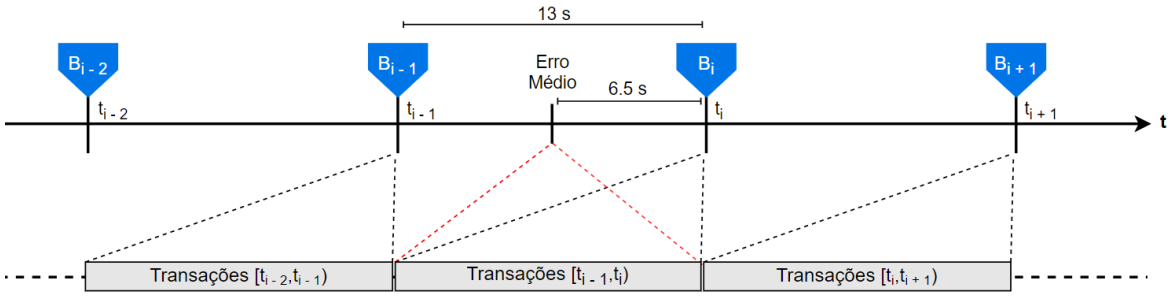

Figura 5 – Linha do tempo dos blocos e transações. As transações entre t_{i-1} e t_i são inseridas no bloco B_i. A acurácia média é de 6,5 segundos.

4.2 CONSENSO DE TEMPO

Na blockchain Ethereum existem algumas regras para os tempos dos blocos. Essas regras permitem atribuir algum grau de confiança ao tempo atribuído aos blocos, visto que elas devem ser cumpridas para o bloco ser inserido na cadeia. As regras são: 1) O tempo do bloco deve ser maior que o tempo do bloco anterior da cadeia; 2) O tempo do bloco deve ser menor que 15 minutos no futuro (ETHEREUM, 2021). A primeira regra apenas reforça a ordem relativa dos blocos, o que já é garantido pelo encadeamento dos blocos. A segunda regra garante tempo suficiente para que o bloco seja enviado para os mineradores pela rede. O tempo de 15 minutos permite abranger mineradores que eventualmente demorem para receber os blocos. Em uma rede ponto-a-ponto alguns participantes podem estar a vários saltos de distância do minerador que criou o bloco e também a rede pode estar sobrecarregada com alto número de transações pendentes sendo transmitidas entre os participantes. Contudo, nessas circunstâncias aumentam-

se as chances de desvios no tempo dos blocos. Vamos analisar as duas possibilidades de desvios de tempo: atraso e avanço.

O atraso do tempo é limitado pela primeira regra. A primeira regra apenas garante que o tempo do bloco será maior que o tempo do bloco anterior. Ou seja, basta o tempo do bloco ser um segundo maior que o tempo do bloco anterior para satisfazer a restrição. Um conluio de mineradores com intenção de atrasar os tempos dos blocos poderia adotar essa prática e atrasar os tempos dos blocos várias horas em um único dia. Contudo, o conluio pode ser quebrado por um único minerador honesto ao criar um bloco com tempo correto. Em uma blockchain pública como a Ethereum é pouco provável que esse tipo de conluio aconteça por conta da imprevisibilidade de quem criará o próximo bloco. Portanto, é razoável assumir que o atraso nos tempos dos blocos dificilmente irá ultrapassar algumas dezenas de segundos.

O avanço do tempo por sua vez é limitado pela segunda regra. A segunda regra garante que não sejam aceitos blocos com mais de 15 minutos no futuro. Esta regra é mais simples pois é um limite fixo de tempo. Um minerador com intenção de avançar o tempo de um bloco pode simplesmente atribuir um tempo de até 15 minutos no futuro, deixando apenas uma margem para o bloco ser transmitido aos participantes. Entretanto, analisando as implementações dos principais clientes[2] Ethereum (Geth, Parity e Open Ethereum) descobriu-se que na prática o limite de tolerância para o tempo dos blocos é de 15 segundos no futuro (GO-ETHEREUM, 2021; PARITY-ETHEREUM, 2021; OPEN-ETHEREUM, 2021). Os clientes Geth, Parity e Open Ethereum constituem cerca de 88 % da rede (ETHERSCAN, 2021g). Por serem softwares de código aberto é possível fazer alterações nos parâmetros, porém é provável que a maioria não o faça. Sendo assim, um bloco com um tempo de 15 minutos no futuro dificilmente será aceito pela rede. Mais do que isso, é razoável assumir que dificilmente blocos com mais de 15 segundos no futuro serão aceitos, por conta da implementação do clientes Ethereum.

A Figura 6 ilustra os cenários de atraso e avanço no tempo dos blocos. O momento da criação do bloco B_i é t_i, contudo ele pode receber um tempo t_i' ligeiramente maior que t_{i-1} (até 12 segundos no passado considerando que o bloco anterior está a 13 segundos e a marcação tem precisão de 1 segundo) ou t_i'' até 15 segundos à frente do momento atual. Ou seja, pode ser atribuido ao bloco qualquer tempo dentro desse intervalo.

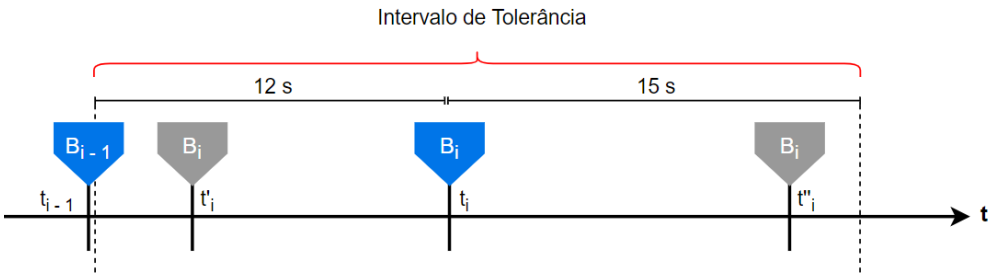

Figura 6 – Linha do tempo dos blocos. O Bloco B_i pode receber o tempo t_i' ou t_i''.

[2] Um cliente Ethereum é um software que implementa um nó Ethereum.

As regras do consenso de tempo mostram que a blockchain não garante alta acurácia (milissegundos ou menor que isso) para os tempos dos blocos. Levando em consideração os possíveis cenários ilustrados, estima-se que a acurácia dos tempos dos blocos seja de algumas dezenas de segundos. Em uma estimativa mais otimista pode-se dizer que a maioria dos blocos não sofreram um desvio maior que 12 segundos no passado ou 15 segundos no futuro, devido a como as regras são aplicadas na prática. Essa acurácia estimada de até 15 segundos refere-se aos tempos dos blocos. A acurácia das transações será vista a seguir.

4.3 FLUXO DE TRANSAÇÕES

A acurácia de uma transação é a diferença entre o momento que a transação foi enviada para a rede e o tempo do bloco em que a transação foi inserida. A acurácia das transações reflete diretamente a acurácia dos carimbos do tempo feitos na blockchain, pelo menos daqueles carimbos do tempo que utilizam o tempo do bloco para o ancoramento temporal. Isso porque é através das transações que as informações são registradas na blockchain.

Quando uma transação é enviada para a rede ela é sinalizada como pendente até ser inserida em um bloco. Após inserida em um bloco a transação é dita confirmada. Confirmar uma transação impõe custos de processamento e armazenamento para a rede. Para incentivar os mineradores a inserirem as transações nos blocos e evitar abusos à rede devem ser pagas taxas pelas transações. O número de transações que podem ser inseridas em um bloco é limitado, portanto os mineradores devem escolher quais transações inserir primeiro. Essa escolha é baseada na taxa paga e o custo de execução da transação. Desta forma, os mineradores priorizam as transações que trazem maior benefício, ou seja, maior razão taxa por custo de execução (ETHEREUM, 2021).

As taxas pagas pelas transações são quantificadas pelo total de recursos utilizados (processamento e armazenamento). O consumo de recursos é expresso em termos de *gas*. O *gas* representa operações realizadas ou armazenamento utilizado. Existe ainda uma precificação para o uso de recursos, o *gas price*. O *gas price* é o quanto de Ether o indivíduo está disposto a pagar por cada *gas* consumido. O total de taxas pago em Ether será $gas \times gas\ price$. Quanto maior o *gas price* oferecido mais rapidamente a transação será inserida em um bloco. Se o *gas price* oferecido for muito baixo a transação pode permanecer como pendente indefinidamente. (ETHEREUM, 2021)

O valor de *gas price* necessário para a transação ser confirmada é dinâmico, depende do número de transações pendentes na rede e o quanto os outros usuários estão dispostos a pagar para terem suas transações confirmadas. Existem alguns serviços que estimam o *gas price* necessário para que uma transação seja confirmada, como ETH Gas Station[3]. O ETH Gas Station (2021) utiliza um modelo de regressão estatística com base nos últimos 10000 blocos para estimar valores de *gas price*. Através de uma API[4], o serviço sugere valores em *gas price*

[3] https://ethgasstation.info/index.php

[4] Interface utilizada para comunicar com o serviço

para uma transação ser confirmada (vide Figura 7) em até 30 minutos (*SafeLow*), até 5 minutos (*Average*), até 2 minutos (*Fast*) e até 30 segundos (*Fastest*). Os valores para o *gas price* são retornados em *Gwei*. O *wei* é a menor fração de Ether. Um Ether equivale a $1x10^{18} wei$ e 1 *Gwei* equivale a $1x10^{9} wei$.

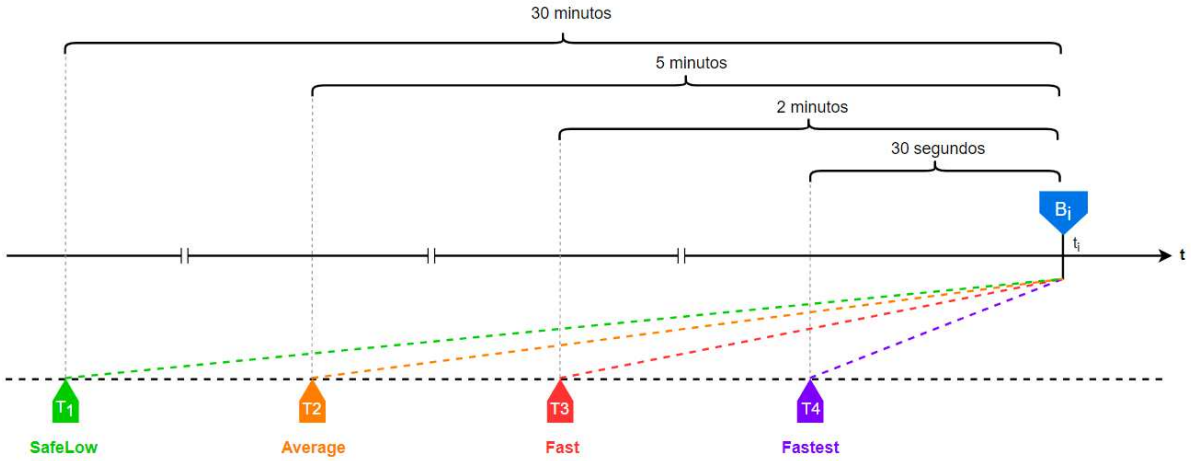

Figura 7 – Linha do tempo dos blocos e transações. Tempo de confimação para transações *SafeLow*, *Average*, *Fast* e *Fastest*.

O valor de *gas price* oferecido pela transação impacta diretamente na acurácia do ancoramento temporal da transação. Supondo utilizar sempre o *gas price* necessário para confirmar a transação em até 30 segundos, essa também será a acurácia estimada para a transação, acrescida da acurácia do bloco visto na seção anterior. Contudo, observando o histórico de transações dificilmente consegue-se estimar quanto tempo a transação permaneceu como pendente até ser confirmada. No melhor dos cenários a acurácia das transações é da ordem de algumas dezenas de segundos.

Esse limite para a acurácia elimina quaisquer chances de aplicações que necessitem de acurácia da ordem de segundos ou menos. Além disso, os valores de *gas price* para confirmar as transações mais rapidamente podem implicar custos que se tornam impraticáveis para algumas aplicações. Mais a frente iremos estimar a acurácia empírica das transações através de um experimento e discutir os custos na rede Ethereum.

5 UM NOVO SERVIÇO DE CARIMBO DO TEMPO DESCENTRALIZADO

Neste capítulo apresenta-se uma proposta de carimbo do tempo em blockchain utilizando contratos inteligentes. Na Seção 5.1, apresenta-se uma visão geral sobre os participantes e o protocolo. A Seção 5.2 traz uma explicação detalhada de como propõe-se que sejam feitos os carimbos do tempo na blockchain do Ethereum. Na Seção 5.3, discute-se um mecanismo para selecionar o melhor tempo para o carimbo. E na Seção 5.4, discute-se brevemente a implementação de um protótipo.

5.1 VISÃO GERAL

Nesta seção apresenta-se um modelo de carimbo do tempo baseado em blockchain com acurácia maior que as soluções existentes (que utilizam o ancoramento pelo tempo do bloco visto no Capítulo 4). Basicamente, propõe-se que o indivíduo interessado em obter o carimbo do tempo para um documento o envie para a rede Ethereum endereçada a um contrato especial. Ao fazer isso, o serviço irá criar um carimbo do tempo para a transação individualmente ao invés de ancorá-la em um bloco. O modelo é detalhado a seguir.

Existem três tipos de participantes no modelo. O primeiro tipo de participante são os indivíduos interessados em criar carimbos tempo para documentos na blockchain. Esses participantes são chamados de *requerentes*. O segundo tipo de participante consiste em nós na rede blockchain que são programados para prover carimbos do tempo. Esses participantes são chamados de *provedores*. E o terceiro tipo de participante é um contrato inteligente que atua como mediador entre os requerentes e os provedores.

A Figura 8 ilustra o processo de criação de carimbo do tempo. O processo é explicado a seguir. Primeiro, o requerente envia uma transação para o contrato inteligente contendo o dado que ele deseja carimbar. Junto, ele envia as taxas para recompensar os mineradores para confirmarem a transação e para os provedores criarem o carimbo do tempo (1 na Figura 3). Na sequência, os provedores enviam para o contrato inteligente uma transação contendo o *timestamp* do momento que eles viram a transação do requerente pela primeira vez (2). Então, o contrato inteligente realiza a seleção do melhor timestamp entre os timestamps recebidos de acordo com alguma estratégia de seleção (3). O contrato inteligente mantém o registro do dado carimbado e o timestamp selecionado. Por fim, o contrato distribui as taxas entre os provedores de acordo com alguma estratégia (4).

Nota-se que um provedor pode observar os timestamps dos outros provedores antes de enviar o seu timestamp para o contrato inteligente. Dessa forma, um provedor desonesto pode utilizar essa informação para manipular o seu timestamp a fim de obter maiores chances de ser selecionado pelo contrato inteligente. Além disso, uma entidade desonesta controlando vários provedores pode efetuar um ataque Sybil. Mais precisamente, a entidade pode aproveitar-se do contrato inteligente fornecendo a maioria dos timestamps para a solicitação de um requerente. Isso aumentaria as suas chances de ter um dos seus timestamps selecionado. A seção a seguir

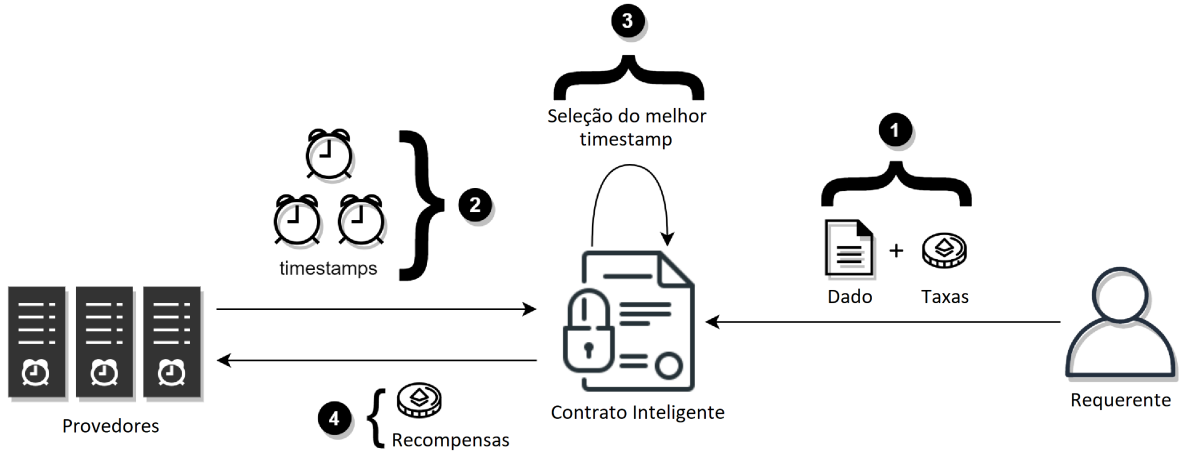

Figura 8 – Contrato inteligente atua como mediador entre requerentes e provedores de carimbo do tempo na blockchain.

propõe formas para solucionar esses problemas.

5.2 PROTOCOLO

Como visto antes, o protocolo deve prevenir entidades desonestas de obterem vantagens ou de conduzirem ataques Sybil. Para resolver esses problemas, o protocolo utiliza provas de comprometimento e exige que os provedores se registrem no contrato inteligente. Aqui as provas de comprometimento são chamadas de *commitments* e as revelações das provas são chamadas de *decommitments*. O protocolo é detalhado a seguir.

Inicialmente, os provedores devem registrar-se no contrato inteligente. O contrato inteligente apenas seleciona carimbos do tempo de provedores registrados. Para registrar-se no contrato inteligente, o provedor deve realizar um depósito $N > 0$ para o contrato inteligente. Refere-se ao conjunto de provedores registrados como P.

Relembra-se que no Ethereum os contratos inteligentes são scripts que apenas executam quando invocados por uma transação. Portanto, o requerente ou os provedores devem iniciar cada passo do protocolo. Se os passos não forem devidamente efetuados, a execução do protocolo poderá parar indefinidamente. Para incentivar a execução do protocolo são estabelecidos prazos e podem ser aplicadas punições aos participantes.

Os participantes do protocolo foram introduzidos na Seção 5.1. Para cada solicitação de carimbo do tempo participam o contrato inteligente, um requerente e um conjunto de provedores P. Assume-se $|P| > 0$. As etapas do protocolo são apresentadas a seguir e uma ilustração mostrando o fluxo principal pode ser vista na Figura 9.

1. Um requerente envia ao contrato inteligente uma transação T contendo um dado d a ser carimbado e o número mínimo $0 < I \leq |P|$ de timestamps esperados. A transação T deve pagar $M + S + R$ em taxas, onde $M, S > 0$ são as recompensas para os mineradores e

provedores, respectivamente, e $R > 0$ é um depósito de garantia feito pelo requerente. As taxas $S + R$ são mantidas pelo contrato inteligente.

2. A transação T dispara a execução do contrato inteligente, que define os prazos $D_c < D_d < D_s$. D_c é o prazo para realização dos commitments, D_d é o prazo para realização dos decommitments e D_s é o prazo para realização da seleção pelo requerente. Os prazos podem ser consultados publicamente no contrato inteligente.

3. Até o prazo D_c, cada provedor em um subconjunto $P' \subseteq P$ envia ao contrato inteligente um commitment $c = H(d||t||r)$, onde H é uma função de hash resistente a colisão, t é o timestamp do momento em que o provedor recebeu a transação T contendo d, e r é uma sequência de bits aleatória suficientemente longa.

4. A qualquer momento em $(D_c, D_d]$, cada provedor em $P' \subseteq P$ envia ao contrato inteligente o correspondente decommitment $\{d, t, r\}$.

5. A qualquer momento $(D_d, D_s]$, apenas o requerente poderá solicitar ao contrato inteligente o melhor timestamp selecionado para d e ter o seu depósito R devolvido. Após D_s, qualquer um na blockchain poderá fazer o mesmo (por exemplo, um provedor) e receber R.

6. Considera-se P'' o subconjunto de provedores que forneceram o decommitment corretamente, sendo $|P''| < |P'|$. Quando solicitado, o contrato inteligente coleta os timestamps $t_1, ..., t_{|P''|}$ para d cujos commitments puderem ser verificados com os decommitments correspondentes. Além disso, o contrato inteligente retorna R. O contrato inteligente seleciona o melhor timestamp para d e retorna R apenas uma vez, qualquer outra solicitação posterior é ignorada pelo contrato inteligente.

7. No caso de $|P''| < I$, o contrato inteligente reembolsa para o requerente as taxas $M + S$ utilizando os depósitos N feitos pelos provedores no momento do registro no contrato inteligente e aborta o protocolo.

8. O contrato inteligente seleciona o melhor timestamp entre $t_1, ..., t_{|P''|}$ e armazena na blockchain juntamente com o valor d.

9. O contrato inteligente compartilha a taxa S entre os provedores P'' que forneceram os timestamps $t_1, ..., t_{|P''|}$. O valor é adicionado aos depósitos N feitos pelos provedores.

Os provedores são desincentivados de executarem ataques Sybil através do pagamento do depósito N ao contrato inteligente pela permissão para participar do protocolo. O contrato inteligente não pode evitar que um provedor controle múltiplas identidades e submeta vários timestamps com a finalidade de aumentar suas chances de receber maiores recompensas. No entanto, pagar um depósito desencoraja os provedores de fazer isso. Na Seção 9.6 discute-se um pouco mais sobre os valores dos depósitos.

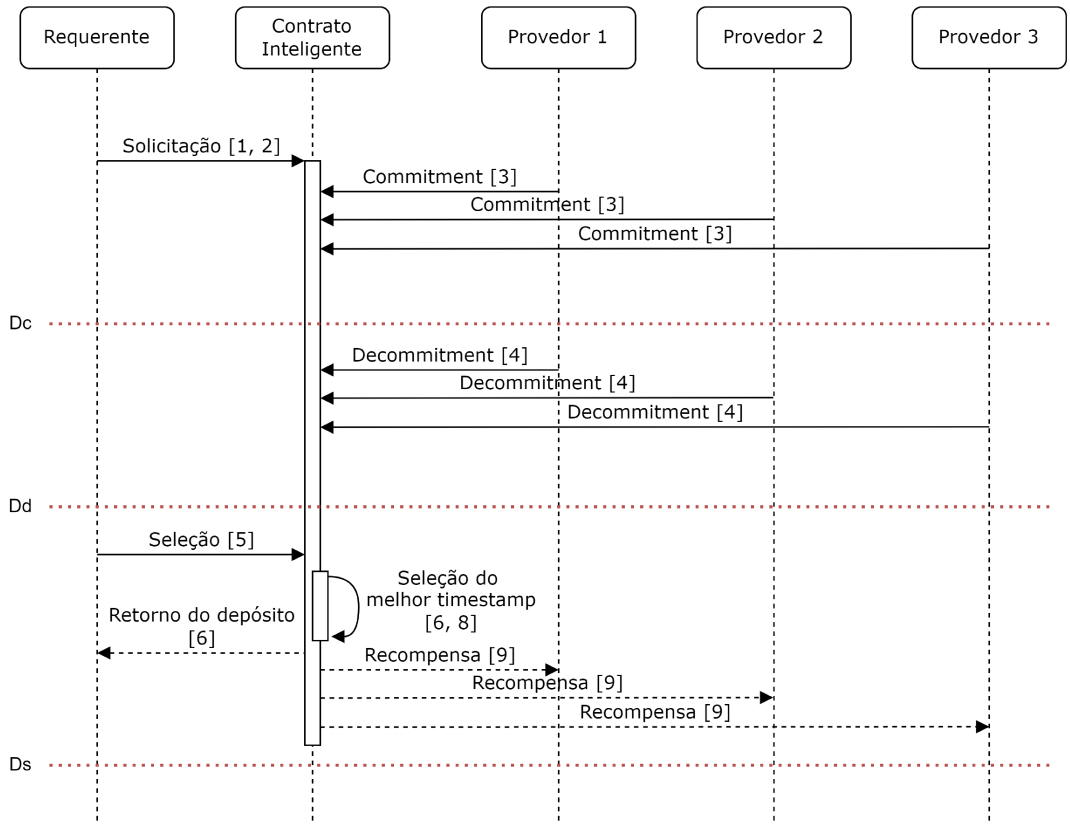

Figura 9 – Fluxo principal do protocolo com a indicação das etapas. Seleção realizada pelo requerente.

O uso de commitments e decommitments nos passos 3 e 4 previne os ataques de presságio. Ou seja, evita que um provedor primeiro descubra os timestamps dos seus concorrentes e então submeta um timestamp sob medida para ser selecionado como o melhor timestamp no passo 8.

A escolha dos prazos D_c, D_d e D_s no passo 2 é um limiar entre o tempo necessário para os provedores completarem o protocolo e as taxas que o requerente paga. Por um lado, quanto menores os prazos, mais rapidamente os provedores devem ter suas transações confirmadas. Ou seja, os provedores devem pagar taxas mais altas aos mineradores. Taxas estas que são cobertas pela recompensa paga pelo requerente. Por outro lado, aumentar os prazos reduz as taxas para confirmação das transações mas aumenta o tempo que deve-se esperar para ter-se uma solicitação de carimbo do tempo atendida. Os valores dos prazos são estimados empiricamente na Seção 6.2.

Como descrito no passo 7, se o número $|P''|$ de timestamps coletados pelo contrato inteligente e com decommitment correto for menor que o número I de timestamps que o requerente espera, então os seus custos por interagir com o contrato inteligente deverão ser reembolsados. O contrato inteligente paga o reembolso debitando dos depósitos pertencentes aos provedores registrados que falharam em fornecer um par de commitment e decommitment corretamente ou que não quiseram fornecer. Desta forma, incentiva-se os provedores registrados a participar corretamente do protocolo sempre que um requerente solicita um carimbo do tempo.

Em contrapartida, o requerente deve cobrir os custos dos P'' provedores quando eles atuarem corretamente com o contrato inteligente e o sistema atingir o número mínimo I de timestamps esperados. Além disso, a recompensa S que o requerente paga para os provedores deve ser maior que os custos para interagir com o contrato inteligente. A diferença deve ser o suficiente para que os provedores possam ter lucro ao participar do serviço de carimbo do tempo. Caso contrário, eles podem solicitar a retirada de seus depósitos e sair do serviço. Sendo assim, o sistema é regulatório, pois o próprio provedor avalia se compensa fornecer o timestamp para uma solicitação baseada na taxa paga. E isso também ajuda a evitar que a solicitação receba mais timestamps do que o desejado. No Capítulo 8 irá retomar-se a discussão sobre o valor de S e outras taxas e depósitos mencionados. Na próxima seção, explica-se como selecionar o melhor timestamp e compartilhar as recompensas entre os provedores.

5.3 SELECIONANDO O TIMESTAMP E COMPARTILHANDO AS RECOMPENSAS

Nesta seção, discute-se as estratégias para selecionar o melhor timestamp entre os timestamps fornecidos pelos provedores e para compartilhar as taxas entre esses provedores.

A seleção do timestamp se refere ao passo 8 do protocolo da Seção 5.2. O contrato inteligente seleciona um timestamp a partir do conjunto de timestamps fornecidos pelos provedores. Quanto mais perto os timestamps estiverem do momento em que o requerente solicita do carimbo do tempo, maior será a acurácia do serviço de carimbo do tempo. Portanto, o timestamp ideal que o contrato inteligente deve selecionar é o que estiver mais perto do momento da solicitação. A Figura 10 mostra um exemplo. No tempo t_R, o requerente solicita o carimbo do tempo ao contrato inteligente. Os provedores $P_1,...,P_5$ fornecem seus timestamps $t_{P_1},...,t_{P_5}$. O timestamp t_{P_1} é o melhor timestamp.

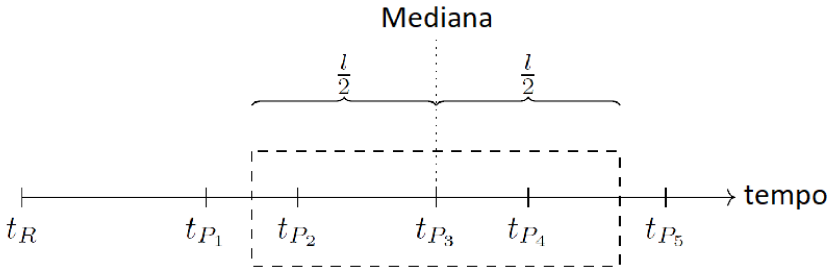

Figura 10 – Os timestamps na linha do tempo. O tempo t_P é o momento em que o requerente solicita o carimbo do tempo. Os timestamps $t_{P_1},...,t_{P_5}$ são os tempos que os provedores $P_1,...,P_5$ receberam a solicitação, respectivamente. Os timestamps fora dos limites da janela de tempo de tamanho l (retângulo tracejado) são considerados discrepantes.

No entanto, deve-se considerar que pode haver timestamps discrepantes. Os timestamps discrepantes podem acontecer porque as transações, antes de serem processadas, são propagadas não uniformemente na rede blockchain. Por conta disso, o momento em que os provedores recebem as transações podem diferir consideravelmente. Além disso, os provedores podem estar com os seus relógios dessincronizados ou podem mentir deliberadamente sobre o

momento que eles recebem a transação. Para o serviço funcionar adequadamente, assume-se que a maioria dos provedores são confiáveis e que a maioria dos timestamps que eles fornecem são não-discrepantes (o desvio aceito será determinado empiricamente). Nesse modelo, a confiança de que os provedores atuarão corretamente é colocada sobre o incentivo financeiro, em conjunto com a estratégia de seleção. Antes de tentar prejudicar o sistema, o provedor deve avaliar o custo-benefício de fazê-lo. Na Seção 9.6 discute-se novamente sobre isso.

Considerando que o contrato inteligente pode coletar alguns timestamps discrepantes no conjunto de timestamps (passo 6 do protocolo da Seção 5.2), uma estratégia consiste em selecionar a mediana do conjunto. Por exemplo, o timestamp t_{P_3} na Figura 10. Apesar de simples, essa estratégia falha em selecionar o melhor timestamp ou outro timestamp mais próximo que não seja discrepante.

Portanto, propõe-se um meio-termo entre selecionar o menor timestamp e evitar timestamps discrepantes. Dado uma janela de tempo como um intervalo de tamanho $l > 0$ centrada na mediana do conjunto de timestamps coletados pelo contrato inteligente para uma solicitação, e dado um subconjunto de timestamps que caem dentro da janela de tempo contendo apenas timestamps não-discrepantes. Para esse subconjunto, o contrato inteligente seleciona o menor deles como o melhor timestamp. A Figura 10 ilustra a janela de tempo como um retângulo tracejado centrado em t_{P_3}. O timestamp t_{P_2} é selecionado como o melhor timestamp, enquanto t_{P_1} e t_{P_5} são considerados discrepantes. Dessa forma, o menor (mais cedo) timestamp não-discrepante é selecionado, promovendo um carimbo do tempo com mais acurácia.

Para essa estratégia, sugere-se que o contrato inteligente selecione o melhor timestamp a partir de um conjunto com número de elementos ímpar e maior que um. Maior que um porque o serviço se propõe a ser descentralizado ao invés de confiar em apenas um timestamp fornecido. E um número ímpar de elementos para garantir que pelo menos um timestamp irá cair dentro da janela de tempo (precisamente, o timestamp mediano). Dessa forma, o contrato inteligente sempre conseguirá selecionar o melhor timestamp, segundo a política. Se o número de timestamps for par, então a janela de tempo não estará centrada em nenhum timestamp e poderá acontecer de nenhum timestamp cair dentro da janela de tempo. Por exemplo, se os timestamps estiverem mais de $l/2$ distantes da mediana. Portanto, o tamanho mínimo para o conjunto de timestamps é três. Para o caso de um número de timestamps maior que três e par, a janela é centrada no timestamp ligeiramente inferior a mediana (que no pior caso tem o mesmo efeito de descartar o maior timestamp).

A distribuição de recompensas para os provedores (passo 9 do protocolo da Seção 5.2) é baseada na estratégia da janela de seleção. Propõe-se compartilhar as recompensas uniformemente entre os provedores que tiveram seus timestamps dentro da janela de seleção (janela de tempo). Caso contrário, pagar apenas ao provedor que teve o timestamp selecionado poderia incentivar os provedores a manipular os seus timestamps a fim de ter o menor timestamp dentro da janela.

Outra opção também considera os provedores que forneceram timestamps discrepantes. Mais precisamente, pagar apenas o suficiente para cobrir os custos totais ou parciais por

interagir com o contrato inteligente. Esta abordagem se preocupa em ser justa e promover qualidade de serviço. Quanto a ser justo, evita-se punir provedores honestos que receberam transações mais rapidamente ou mais lentamente que outros por conta de anomalias na rede blockchain. Quanto a qualidade de serviço, incentiva os provedores a manter o maior número de conexões possíveis com os vizinhos para receber as transações mais rapidamente.

O ponto chave é estimar o tamanho l para a janela de tempo. Por um lado, quanto maior a janela, maiores serão as chances de selecionar timestamps discrepantes. Por outro lado, quanto menor a janela, menores as chaves de selecionar o melhor timestamp não-discrepante. Definiu-se empiricamente o tamanho da janela de tempo na Seção 6.2.2.

5.4 PROTÓTIPO

Neste trabalho foi desenvolvido um protótipo para a implementação do protocolo apresentado na Seção 5.2. A implementação consistiu de um contrato inteligente e dois scripts.

O contrato inteligente contém um conjunto de funções que os participantes podem chamar a fim de interagir com o contrato inteligente e participar do protocolo. Mais precisamente, um requerente solicita um carimbo do tempo ao contrato inteligente (passo 1 do protocolo) chamando a função *requestTimestamp*. Ao chamar a função, o requerente passa um parâmetro que é o dado a ser carimbado. Esta função também dispara o passo 2, onde são definidos os prazos. Um provedor envia um commitment (passo 3) e um decommitment (passo 4) ao contrato inteligente chamando as funções *addCommitment* e *addDecommitment*, respectivamente. Conforme os prazos definidos, ou o requerente ou um dos provedores pode solicitar ao contrato inteligente a seleção de um timestamp chamando a função *selectTimestamp*. Nesta função também, o timestamp é então armazenado juntamente com o dado carimbado e as recompensas são distribuídas (passos 5-9). A Tabela 3 sumariza essas funções e os respectivos passos do protocolo que elas executam.

Função	Passos do Protocolo
requestTimestamp	1-2
addCommitment	3
addDecommitment	4
selectTimestamp	5-9

Tabela 3 – As funções implementadas no protótipo e os respectivos passos do protocolo que as funções executam.

Para armazenar o timestamp selecionando juntamente com o valor carimbado (passo 8) a função *selectTimestamp* utiliza memória não-volátil para que qualquer um possa verificar os dados posteriormente. A função armazena o timestamp em um dicionário para tornar as consultas mais eficientes, no qual o dado carimbado mapeia ao timestamp selecionado.

Os dois scripts são destinados um para os requerentes e outro para os provedores. Os scripts permitem conectar-se a Ethereum Main Network e enviar as transações necessárias para disparar a execução das funções citadas acima. Adicionalmente, o script criado para os

provedores permite a eles monitorarem quando um requerente envia uma transação chamando a função *requestTimestamp* na Ethereum Main Network. Dessa forma, os provedores podem fornecer o timestamp do momento que eles receberam a transação pela primeira vez.

O contrato inteligente foi escrito em linguagem Solidity (Ethereum Foundation, 2021a) e os dois scripts foram escritos em linguagem JavaScript. Os scripts utilizam a biblioteca Web3js (Ethereum Foundation, 2021b) para acessar rede blockchain e rodam em ambiente de execução NodeJS (OpenJS Foundation, 2021).

A implementação do protótipo foi utilizada para conduzir os experimentos e estimar os custos de execução do protocolo proposto na Ethereum Main Network. Os experimentos e a estimativa de custos são apresentados nas próximas seções.

6 EXPERIMENTOS

Este capítulo descreve dois experimentos que foram conduzidos neste trabalho. O primeiro experimento (Seção 6.2) teve como objetivo encontrar os tempos de propagação e confirmação das transações na Ethereum Main Network. Os resultados do primeiro experimento foram utilizados como parâmetros para a avaliação da proposta no segundo experimento (Seção 6.3). A Seção 6.1 descreve a configuração do ambiente utilizado na coleta de dados. Optou-se por realizar os experimentos na rede principal da Ethereum para poder obter dados reais sobre a propagação e confirmação das transações.

O primeiro experimento foi executado entre 23 e 30 de novembro de 2019 e o segundo experimento foi executado no dia 3 de dezembro de 2019. Os altos custos para executá-los (centenas de dólares) dificultam uma replicação mais recente. Contudo, acredita-se que apenas os custos dos carimbos podem ter sido afetados desde então, sendo que esses podem ser convertidos para o equivalente atual. Os custos do modelo proposto são apresentados no Capítulo 8.

6.1 CONFIGURAÇÃO DO AMBIENTE

A execução dos experimentos contou com seis nós Ethereum conectados na Main Network. Os nós foram implantados em instâncias Amazon EC2 com sistema operacional Amazon Linux 2. A Tabela 4 mostra a configuração das máquinas e os softwares utilizados. Foram implantados um nó do tipo *Light* e cinco nós do tipo *Full*. Para cada tipo foram utilizados perfis de hardware diferentes. E, foi utilizado o serviço de NTP da UFSC[1] para realizar a sincronização dos relógios das máquinas.

Nó	Máquina	Cliente Ethereum
Light	EC2 t2.small: um processador virtual, 2GB RAM, performance de rede entre baixo e moderado.	Go-ethereum 1.9.7 stable
Full	EC2 m4.large: dois processadores virtuais, 8GB RAM, alta performance de rede.	Parity 2.5.10 stable

Tabela 4 – Configurações das instâncias Amazon EC2 utilizadas nos experimentos.

O nó Light foi utilizado para enviar transações à rede. Para enviar transações à rede não é necessário manter uma cópia da blockchain. Além disso, é desejável manter o maior número de conexões possíveis com nós vizinhos para propagar as transações o mais rápido possível. Kim et al. (2018) mostram que o Go-Ethereum envia as transações para todos os nós vizinhos, enquanto o Parity envia apenas para $\sqrt{n}$ vizinhos. Por isso, o Go-Ethereum foi escolhido para

[1] https://setic.ufsc.br/servicos/basico-de-rede/servico-de-ntp/

o nó Light. Também optou-se por nó Light por não manter uma cópia da blockchain, o que permite utilizar uma máquina com menos recursos e consequentemente com um custo menor.

Os nós Full foram utilizados para observar transações na rede. Ao contrário do nó Light, para observar as transações na rede tanto o Go-Ethereum quanto o Parity requerem manter uma cópia da blockchain. De acordo com (ETHERSCAN, 2021d), o Parity demanda menos espaço de armazenamento que o Go-Ethereum. Portanto, para os nós Full escolheu-se utilizar o Parity com a opção de manter uma cópia da blockchain. Mesmo assim, foi necessária uma máquina com maior performance para fazer o download da blockchain e receber todo o tráfego da rede.

As instâncias das máquinas EC2 foram alocadas em diferentes regiões geográficas a fim de assemelhar-se com a distribuição da Ethereum Main Network. A instância que implantou o nó Light foi alocada em Ohio nos Estados Unidos. As outras instâncias, que implantaram os nós Full, foram alocadas em Virgínia do Norte nos Estados Unidos, São Paulo no Brasil, Frankfurt na Alemanha, Seoul na Coréia do Sul e Sidney na Austrália. Foram alocados dois nós nos Estados Unidos por dois motivos. Primeiro, analisar se a proximidade geográfica influencia na propagação das transações, apesar de que Kim et al. (2018) apontam que os nós vizinhos são selecionados aleatoriamente. E segundo, porque o país concentra o maior número de nós da Ethereum Main Network (GMBH, 2021).

Os nós devem estar conectados a pelo menos um outro nó para enviar ou receber transações. Além disso, quanto mais conexões estabelecidas mais rápido as transações são enviadas e recebidas. A Tabela 5 mostra o número de vizinhos mantidos pelos nós no primeiro experimento, o mais longo deles. Como pode ser visto, nenhum dos nós foi isolado da rede durante o experimento. No entanto, o nó em Ohio, que utilizou um cliente Go-Ethereum para enviar transações, teve uma média menor e variou mais. Mesmo assim, o número médio de vizinhos ainda é maior que a raiz quadrada dos outros nós, o que corrobora a escolha do Go-Ethereum como cliente para enviar as transações.

Localização do Nó	Média de vizinhos	Desvio Padrão	Mínimo	Máximo
Ohio	18,19	3,12	12	27
Virgínia do Norte	37,81	1,72	34	42
São Paulo	37,08	1,24	34	41
Frankfurt	38,38	1,39	35	42
Seoul	39,17	1,43	36	43
Sidney	39,24	1,01	36	43

Tabela 5 – O número de vizinhos de cada nó durante o primeiro experimento.

6.2 ACURÁCIA DAS TRANSAÇÕES

O objetivo desse experimento é encontrar a acurácia empírica das transações na Ethereum Main Network. A acurácia do ancoramento das transações é o que define a acurácia dos

carimbos do tempo em blockchain, como visto no Capítulo 4. Foram analisados quanto tempo leva a propagação das transações (Seção 6.2.1) e quanto tempo os mineradores levam para confirmar as transações (Seção 6.2.3). Adicionalmente, analisou-se também a diferença dos tempos de propagação entre os nós para uma mesma transação (Seção 6.2.2). Foram utilizados os nós conectados na Ethereum Main Network apresentados na Seção 6.1 para obtenção dos dados. Os detalhes do experimento são apresentados a seguir.

O nó Light foi utilizado como emissor e os nós Full foram utilizados como receptores. O emissor enviou transações regularmente para um endereço específico na rede e anotou o momento em que enviou cada transação. Os nós receptores ficaram observando quando novas transações eram enviadas para o endereço específico e anotaram o momento em que viram a transação pela primeira vez.

O emissor enviou 395 transações durante os sete dias de experimento. O envio das transações era disparado com um intervalo de 25 minutos. No entanto, as transações poderiam aguardar até 5 minutos para serem enviadas. A opção de aguardar antes de enviar uma transação era aplicada conforme o *gas price* sugerido pelo ETH Gas Station no momento. A definição do *gas price* e o ETH Gas Station foram apresentados na Seção 4.3. Foi utilizado o *gas price average*, o sugerido para as transações serem confirmadas em até 5 minutos. Contudo, quando o valor sugerido era maior que 6 Gwei então aguardava-se até 5 minutos para o caso de *gas price* diminuir. Durante esse período, verificava-se o *gas price* sugerido a cada 10 segundos, caso o *gas price average* diminuísse para 6 Gwei ou menos então enviava-se a transação. Se isso não acontecesse durante o período de 5 minutos então a transação era enviada com o *gas price* de 6 Gwei.

Esse esquema foi utilizado para diminuir os custos do experimento. Não foram enviadas transações com gas price maior que 6 Gwei. O limite de 5 minutos foi definido para não aguardar-se indefinidamente e comprometer o envio da próxima transação. Nesse caso, aguardar até 5 minutos para enviar a transação não afetou a duração total do experimento. Isso porque o intervalo médio entre as transações continuou em aproximadamente 25 minutos. Por exemplo, ao atrasar uma transação em 5 minutos, o emissor começaria a tentar enviar a próxima transação 20 minutos depois. Isto é, o período em que o emissor fica aguardando para enviar uma transação não atrasa o início da tentativa de enviar a próxima transação.

Para enviar e observar as transações foram utilizados os scripts mencionados na Seção 5.4, mas sem adotar o protocolo completo. Apenas utilizou-se o envio de transações, papel exercido pelo requerente, e o recebimento de transação, papel exercido pelo provedor.

Como os nós são conectados a uma rede ponto-a-ponto, as transações podem enfrentar diferentes latências e números de saltos, portanto amostras discrepantes podem ocorrer. Quando isso ocorreu, as amostras foram identificadas como discrepantes de acordo com o Método do Intervalo Interquartil (FORSYTH, 2018).

6.2.1 Propagação das Transações

Nesta seção são apresentados os dados coletados referentes aos tempos de propagação das transações, ou seja, quanto é despendido pela propagação das transações na rede. Para encontrar os tempos de propagação, computou-se a diferença entre o momento que o emissor enviou a transação para rede e o momento que os receptores receberam a transação pela primeira vez. Os tempos foram aferidos em milissegundos. A Tabela 6 mostra os tempos computados, já desconsiderando as amostras discrepantes. A coluna mais à direita mostra a porcentagem das transações que o nó recebeu antes dos outros nós.

Localização do Nó	Média (ms)	Desvio Padrão	Transações Recebidas Primeiro (%)
Frankfurt	151,36	71,45	56,63
Virgínia do Norte	157,52	78,45	34,18
Seoul	198,84	70,23	7,91
São Paulo	206,17	76,40	0,77
Sidney	228,99	63,95	0,51
Todas localizações	188,35	78,06	-

Tabela 6 – O tempo necessário para cada nó receber as transações. A coluna mais à direita mostra a porcentagem das transações que o nó recebeu antes dos outros nós.

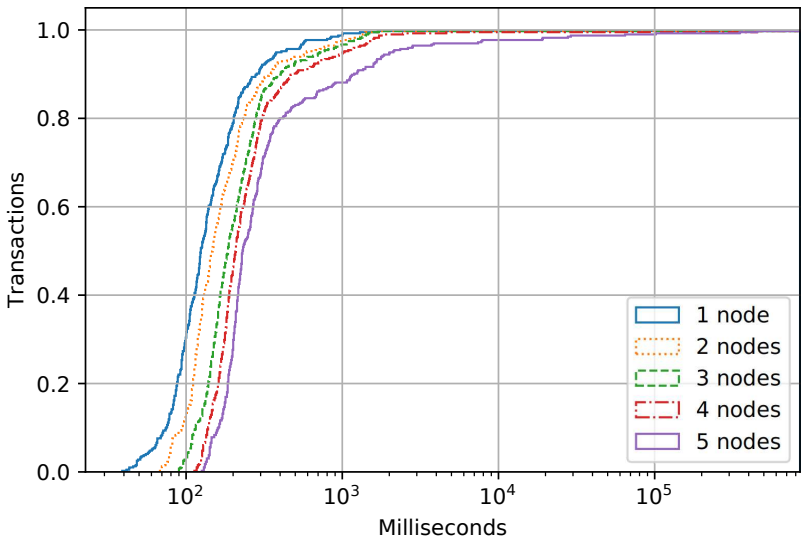

Figura 11 – Proporção de transações pelo tempo que levaram para alcançar os nós receptores.

Como pode ser visto na Tabela 6, foram necessários 188,35 milissegundos em média para os nós receberem as transações na Ethereum Main Network. No entanto, esses tempos podem variar significativamente devido à propagação não uniforme das transações na rede ponto-a-ponto, como pode ser visto na Figura 11. Especificamente, o desvio padrão corresponde a mais de 40% da média. A variância observada na velocidade em que as transações são propagadas dificulta estimar-se precisamente quanto tempo leva entre o momento que o requerente

solicita o carimbo do tempo e o momento em que o serviço de carimbo do tempo atende a solicitação. No entanto, esse resultado impõe um limite inferior na casa de milissegundos para a acurácia que os serviços de carimbo do tempo podem fornecer na blockchain.

A Tabela 6 também mostra que estar geograficamente perto do nó onde a transação foi gerada não garante que a transação seja recebida mais rapidamente. Isso porque a propagação depende das conexões estabelecidas na rede ponto-a-ponto. O nó em Frankfurt, mais longe de Ohio do que a Virgínia do Norte, teve um tempo médio menor no recebimento das transações e foi o primeiro a receber mais da metade das transações. Por outro lado, o nó na Virgínia do Norte recebeu as transações mais rapidamente do que Seoul, São Paulo e Sidney, que estão mais longe de Ohio do que a Virgínia do Norte. Três transações foram recebidas em Norte Virgínia e Frankfurt simultaneamente antes de chegarem aos outros nós. Isso é, os dois nós marcaram os mesmos tempos com a precisão adotada. Essas três transações não foram contabilizadas na coluna mais à direita da Tabela 6.

Valores discrepantes ocorreram devido à propagação não uniforme das transações na rede ponto-a-ponto. Segundo o Método do Intervalo Interquartil (FORSYTH, 2018), 9,72% das amostras dos tempos de propagação foram valores discrepantes. Além disso, o tempo máximo para uma transação ser recebida por um nó foi de 32,6 minutos, o que é mais de 10.000 vezes maior que o limite para não ser considerado discrepante. Em contraste, o tempo mínimo que uma transação levou para ser recebida por um nó foi de 39 milissegundos.

Para entender-se melhor como as discrepância afetam o recebimento das transações pelos nós, na Figura 11 é possível observar os tempos que as transações levaram para serem recebidas por um, dois, três, quatro e cinco nós. Em torno de 98% das transações levaram até um segundo para serem recebidas por pelo menos um nó, o que é mais de cinco vezes maior que a média de 188,35 milissegundos.

Para os tempos apresentados não distinguiu-se entre as taxas pagas pelo nó emissor para confirmação das transações pelos mineradores. Conforme descrito anteriormente, o emissor procurou pagar o necessário para as transações serem confirmadas em até 5 minutos, mas sem pagar mais do que 6 Gwei. Visto que os valores da taxa oscilam e em alguns momentos acima de 6 Gwei, nem todas as transações foram enviadas com a taxa total. Portando, pode-se dividir as transações de acordo com o *gas price* pago em dois tipos: 1) transações com *taxa total*, isto é, transações enviadas com o *gas price average*, e; 2) transações com *taxa parcial*, isto é, transações enviadas com *gas price* de 6 Gwei quando o *gas price average* era mais que 6 Gwei. Entre as transações, 71,65% foram enviadas com taxa total, enquanto o restante foram enviadas com taxa parcial. Para as transações com taxa parcial, o tempo de propagação médio foi 48,89% maior e o desvio padrão foi 9,48% maior. Isso é um indício de que quando os valores das taxas aumentam, a rede está sobrecarregada com transações pendentes.

6.2.2 Janelas de Tempo

A partir dos dados de propagação, comparou-se o momento em que os diferentes nós receberam as transações. Mais precisamente, computou-se a diferença entre o primeiro nó receber uma transação até o último nó receber a mesma transação. Refere-se a essa diferença como *janela de tempo*. Este dado importante para a decisão do tamanho da janela de tempo do método de seleção do timestamp na proposta. O tempo médio foi de 101,70 milissegundos, com desvio padrão de 29,47 milisegundos. Para este caso, os valores discrepantes foram 17,47% das janelas computadas. A Figura 12 ilustra a função de distribuição acumulada das janelas de tempo. Em torno de 90% das janelas não são maiores que 1 segundo.

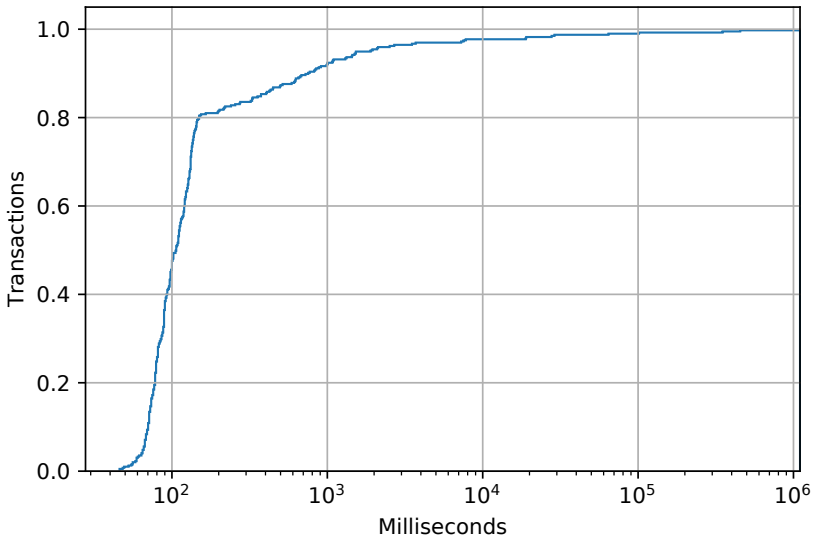

Figura 12 – Proporção de transações pela diferença de tempo entre o primeiro e o último nó receber uma mesma transação.

6.2.3 Confirmação das Transações

Nesta seção são apresentados os dados coletados quanto aos tempos de confirmação das transações, ou seja, quanto tempo as transações permaneceram como pendentes até os mineradores inserirem as transações nos blocos e os blocos serem minerados. Os tempos foram computados como a diferença entre o momento que o emissor enviou a transação para a rede e o tempo do bloco no qual a transação foi inserida. Aqui, diferenciam-se as transações que pagaram taxa total e taxa parcial, como explicado no final da Seção 6.2.1. A Tabela 7 apresenta os tempos de confirmação em segundos para os dois tipos de transações. Os valores discrepantes foram tempos menores que -101,00 segundos ou maiores que 187,00 segundos, sendo um total de 13,67% das amostras. Os valores negativos encontrados estão relacionados a um padrão de anômalias encontrado, que será visto mais à frente.

Taxa	Média (s)	Desvio Padrão	Mínimo (s)	Máximo (s)
Total	28,53	37,22	-33,15	1942,81
Parcial	39,46	44,14	-6,64	6422,10

Tabela 7 – O tempo que os mineradores levaram para confirmar as transações. As transações com taxa total foram enviadas com o *gas price* sugerido para serem confirmadas em até 5 minutos (*gas price average*), enquanto as transações com taxa parcial foram enviadas com *gas price* abaixo do sugerido.

A Tabela 7 mostra que os tempos de confirmação variaram significativamente. Especificamente, o desvio padrão é maior que a própria média. As transações com taxa total são aquelas enviadas com o *gas price average*, enquanto as transações com taxa parcial são aquelas enviadas com *gas price* de 6 Gwei quando o *gas price average* é maior que do isso. O segundo caso acontece provavelmente quando a rede está com um número maior de transações pendentes. Pelo fato de os mineradores priorizarem as transações com taxas mais altas e pela rede estar com uma carga maior, as transações com taxa parcial levaram mais tempo em média para serem confirmadas. Contudo, ambos os tipos de transações levaram menos do que os 5 minutos esperados para serem confirmadas, até mesmo as transações com taxa parcial.

Neste experimento um padrão de anomalias chamou a atenção. Especificamente, 12,41% dos tempos de confirmação foram negativos. Mais precisamente, nesses casos, o timestamp do bloco no qual a transação foi inserida era anterior ao momento que a transação foi enviada para a rede. Essas ocorrências sugerem que o minerador pode não ter atualizado o timestamp do bloco após inserir as transações durante o período em que esteve minerando o bloco. O Capítulo 7 traz algumas evidências para fundamentar essa hipótese. No entanto, percebe-se que o timestamp que os mineradores atribuem aos blocos podem ser utilizados para criar carimbos do tempo no passado.

O tempo de confirmação compreende o tempo de propagação e o tempo de mineração. No entanto, visto que os tempos de propagação são da ordem de milissegundos e os tempos de confirmação são da ordem de segundos, entende-se que o tempo de propagação é uma fração do tempo de confirmação que pode ser negligenciada.

A Figura 13 mostra a função de distribuição acumulada dos tempos de confirmação. Em torno de 91% das transações que pagaram taxa total foram confirmadas em menos de 180 segundos, o que é menos do que os 5 minutos esperados. Em contraste, apenas 80% das transações que pagaram taxa parcial foram confirmadas em até 5 minutos. No entanto, esses tempos devem ser analisados com cautela porque os tempos de confirmação das transações podem ter sido reduzidos em até algumas dezenas de segundos, como visto pelos casos anômalos. Como a maioria das transações que pagaram a taxa total foram confirmadas dentro do tempo esperado, no segundo experimento (Seção 6.3) foram utilizadas apenas transações pagando a taxa total.

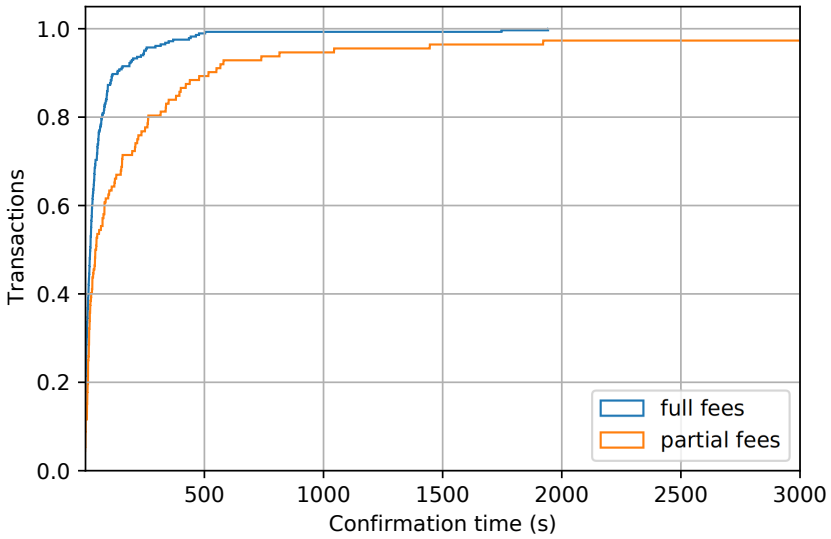

Figura 13 – Proporção de transações pelo tempo de confirmação das transações pagando a taxa total (*full fees*) ou uma taxa parcial (*partial fees*).

6.3 AVALIAÇÃO DO PROTÓTIPO

Nesta seção avalia-se o protótipo implementado para a proposta apresentada no Capítulo 5 na Ethereum Main Network. A diferença em relação ao experimento anterior é que neste momento o protocolo da Seção 5.2 foi executado por completo. O nó emissor atuou como requerente e os nós receptores atuaram como provedores. Além disso, o contrato inteligente também foi implantado. Os passos a seguir mostram como o experimento foi realizado:

1. o papel do requerente foi executado pelo nó emissor, enviando transações com solicitações de carimbo do tempo para o contrato inteligente;

2. o papel dos provedores foram executados pelos nós receptores, ao receber uma transação com uma solicitação de carimbo do tempo enviavam um commitment ao contrato inteligente com o tempo que eles viram a transação pela primeira vez;

3. os provedores revelavam os tempos com os respectivos decommitments; e,

4. o nó requerente solicitava a seleção do carimbo do tempo para o contrato inteligente, o qual utilizou a estratégia da janela de tempo apresentada na Seção 5.3.

Os parâmetros do contrato inteligente foram definidos conforme os resultados do experimento anterior. Especificamente, os prazos D_c, D_d e D_s do protocolo foram definidos como 180 segundos, o que corresponde ao tempo de confirmação de 91% das transações visto na Seção 6.2.3. E para a seleção do melhor timestamp foi definido uma janela de tempo de 1 segundo, que compreende pelo menos 90% das janelas de tempo obtidas na Seção 6.2.2.

O experimento foi executado em um período de 24 horas. O nó requerente foi programado para disparar uma transação por hora solicitando um carimbo do tempo para o contrato inteligente. A taxa atribuída às transações foi sempre a taxa total, o recomendado para a transação ser confirmada em até 5 minutos. Por conta disso, em alguns momentos o requerente não possuía fundos suficientes para enviar a transação. Quando isso ocorria, o requerente aguardava até 5 minutos para o caso do valor da taxa diminuir. Contudo, ao contrário do experimento anterior, não foram enviadas transações com taxa parcial. Portanto, se o valor da taxa não diminuísse, a transação era abortada.

O requerente foi programado para disparar 24 transações com solicitações de carimbo do tempo para o contrato inteligente. Das 24 transações, 21 foram enviadas com sucesso. As outras três transações foram abortadas porque o requerente não possuía a quantidade necessária de fundos para enviar a transação.

6.3.1 Prazos do Protocolo

Com os dados do experimento avaliou-se se o prazo estabelecido para a conclusão dos commitments, decommitments e seleção do timestamp foi adequado. Das 21 solicitações de carimbo do tempo enviadas, todos os nós conseguiram completar o protocolo. Isto é, os cinco nós provedores forneceram os commitments quando receberam a solicitação e os respectivos decommitments. Além disso, o requerente pôde solicitar a seleção do melhor timestamp no momento destinado a isso. Esse resultado mostrou que o período de 180 segundos para os prazos de commitments, decommitments e seleção do timestamp foram suficientes.

6.3.2 Janela de Tempo

Também avaliou-se se o tamanho estabelecido como 1 segundo para a janela de tempo foi adequado para o enquadramento dos timestamps. Para cada transação os provedores forneceram 5 timestamps, um timestamp cada, que foram coletados pelo contrato inteligente. A Figura 14 mostra que pelo menos três timestamp caíram dentro da janela de tempo em 20 das 21 solicitações. Em uma das solicitações houve timestamps discrepantes, e nesse caso a estratégia pôde identificá-los, sendo que apenas dois timestamps caíram dentro da janela. Esse resultado demonstra que a janela de tempo com tamanho de 1 segundo foi adequada para enquadrar a maioria dos timestamps na maioria das vezes.

Contudo, o tamanho da janela foi estabelecido de acordo com a observação do experimento que ocorreu em uma semana e o teste de adequação foi executado na semana seguinte. Se a atividade da rede fosse significativamente maior na semana seguinte (por exemplo, por conta de uma *Initial Coin Offering*), o tamanho da janela poderia não ter sido suficiente para os nós provedores receberem a transação e fornecerem tempos que enquadrassem-se na janela. Por outro lado, a janela de tempo não depende do número de provedores ou solicitações. Portanto,

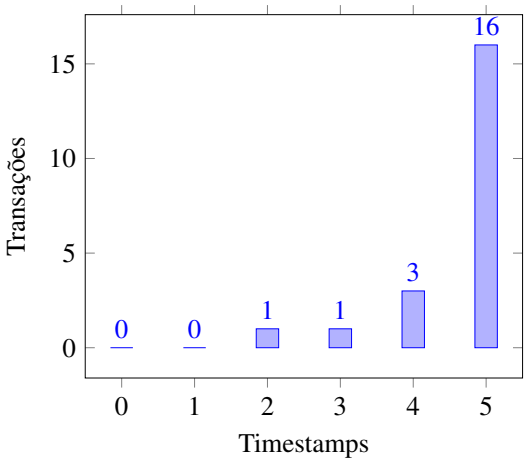

Figura 14 – Distribuição de frequências dos timestamps que enquadraram-se na janela de tempo de 1 segundo.

uma janela de tempo com o mesmo tamanho poderia ser adequada para um número maior de provedores ou solicitações simultâneas de carimbo do tempo no contrato inteligente.

6.3.3 Estratégia de Seleção

A acurácia do carimbo do tempo fornecida pelo serviço depende da seleção do melhor timestamp, ou seja, o timestamp mais próximo do momento da solicitação. Para analisar a acúracia computou-se a diferença entre o momento da solicitação e o timestamp selecionado pelo contrato inteligente. Na média, a acurácia do timestamp selecionado foi de 121,10 milissegundos com desvio padrão de 37 milissegundos.

Por fim, contou-se o número de timestamps selecionados como o melhor timestamp para cada um dos nós. A Figura 15 mostra que a Virgínia do Norte e Frankfurt forneceram a maioria dos timestamps selecionados. Nota-se que, ao contrário do experimento anterior, a Virgínia do Norte recebeu as transações primeiro mais vezes do que Frankfurt, fornecendo o melhor timestamp.

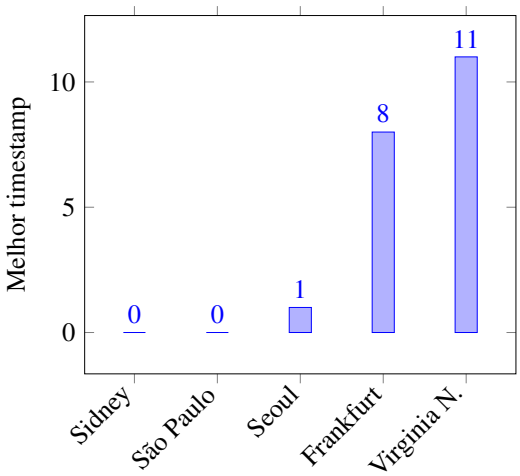

Figura 15 – Número de vezes que cada localização forneceu o melhor timestamp.

7 INVESTIGANDO TEMPOS DE CONFIRMAÇÃO NEGATIVOS

Na Seção 6.2.3 foi reportado o fato inesperado de ocorrerem transações com tempos de confirmação negativos. Isto é, os blocos que as transações foram inseridas possuiam um timestamp anterior ao momento em que as transações foram criadas. Isso aconteceu em 12,41% das transações do primeiro experimento. Esse é um grave problema pois permite criar carimbos do tempo no passado com os serviços de carimbo do tempo que utilizam os tempos dos blocos como ancoramento temporal. Este capítulo fornece algumas evidências que podem explicar porque isso ocorre.

Primeiramente, assume-se que os mineradores não atrasam os tempos dos blocos intencionalmente. Isto é, assume-se que os mineradores não manipulam os timestamps dos blocos (por exemplo, escolhendo um timestamp no passado que seja apenas suficientemente maior que o timestamp do bloco anterior). Mas ao invés disso, assume-se que os mineradores atribuem o timestamp do momento atual. Assumir isso é plausível considerando que a maioria dos minerados são supostamente honestos. Sendo assim, presume-se que os mineradores tenham inadvertidamente ancorado transações no passado. Isso pode ter acontecido se eles adicionaram novas transações em um bloco durante a mineração após ter atribuído o timestamp. Mais especificamente, está pressupondo-se que, quando uma nova rodada de mineração está para começar, os mineradores atribuem ao bloco um timestamp com o momento atual e começam a minerar. Contudo, durante a mineração adicionam novam transações ao bloco mas não atualizam o timestamp do bloco. A seguir, fornece-se alguns dados que fundamentam essa afirmação.

Observando as transações do experimento da Seção 6.2, distingui-se as transações com tempos de confirmação negativos conforme a taxa paga e o número de segundos no passado. A Figura 16 mostra que a maioria das transações com tempos de confirmação negativos pagaram a taxa total (full fees), ou seja, uma taxa relativamente mais alta. As transações que pagaram a taxa total correspondem a 87,76% das ocorrências. Além disso, a média das diferenças entre o momento em que as transações foram criadas e o tempo dos blocos que as transações foram inseridas não é maior que 13 segundos, o qual é justamente o tempo médio que os mineradores levam para criarem novos blocos. Essa descoberta sugere que os mineradores adicionaram novas transações ao bloco sendo minerado não apenas antes de começar a mineração, mas também enquanto eles estavam minerando (especialmente se as novas transações possuíam taxas mais altas). Caso contrário, toda transação recebida durante a mineração de um bloco teria que esperar até o final da rodada para então serem inseridas no próximo bloco a ser minerado, com um timestamp atualizado posterior ao recebimento das transações. Se esse fosse o caso, não teriam sido observados blocos com timestamps anteriores ao momento em que as transações foram criadas.

Em sequência, buscou-se investigar se os mineradores realmente não atualizam o timestamp durante a mineração. Para verificar essa afirmação comparou-se o momento em que um nó recebeu um novo bloco com o timestamp do próximo bloco (isto é, um bloco pai e um bloco filho, respectivamente). Inicialmente, assume-se que o tempo de propagação dos blocos

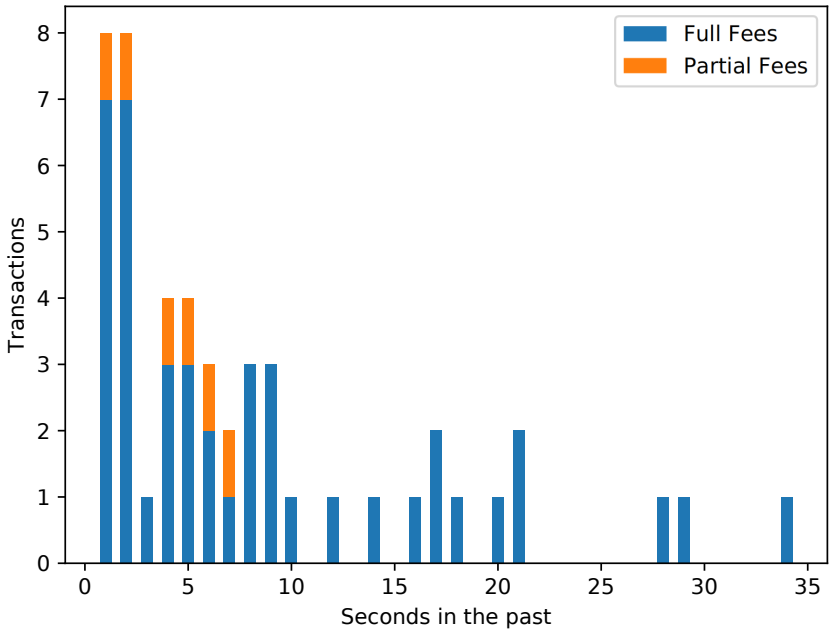

Figura 16 – Distribuição das transações com tempos de confirmação negativos observadas no experimento da Seção 6.2.

na rede é negligenciável e que os mineradores recomeçam o processo de mineração assim que criam um bloco ou ao recebem um novo bloco de outro minerador. Portanto, se o momento em que um nó recebeu um bloco pai é próximo ao timestamp do bloco filho, então significa que o minerador do bloco filho atribuiu o timestamp ao bloco logo após ter recebido o bloco pai e não atualizou o timestamp enquanto estava minerando o bloco filho.

Nos experimentos do Capítulo 6 não foram coletados os tempos em que os nós recebiam novos blocos. Por conta disso, foram utilizados dados de um experimento anterior conduzido em um ambiente similar. O experimento foi realizado entre os dias 16 e 22 de Julho de 2019. Foram utilizados quatro nós light com cliente Ethereum parity 2.5.5 stable. Os nós foram localizados em Ohio nos Estados Unidos, Frankfurt na Alemanha, Seoul na Coréia do Sul e Florianópolis no Brasil. Os nós anotaram o momento em que recebiam um novo bloco juntamente com o número e o timestamp do bloco. Todos os nós coletaram o mesmo conjunto de blocos. O conjunto de blocos coletados foi de 37703 blocos.

Para cada bloco foi computado a diferença entre o momento em que o bloco foi recebido por um dos quatro nós e o timestamp do próximo bloco. Foi escolhido o menor tempo entre os quatro nós para minimizar a defasagem causada pela latência de rede. A Figura 17 ilustra um histograma das diferenças computadas.

Para 97,51% dos blocos a diferença ficou entre -5 e 5 segundos. Essa diferença é condizente com a latência de rede esperada em uma rede ponto-a-ponto, como pode ser visto na Seção 6.2.1. Além disso, fornece evidência que os mineradores não atualizam os timestamps dos blocos durante a mineração. Se os mineradores estivessem atualizando os timestamps dos

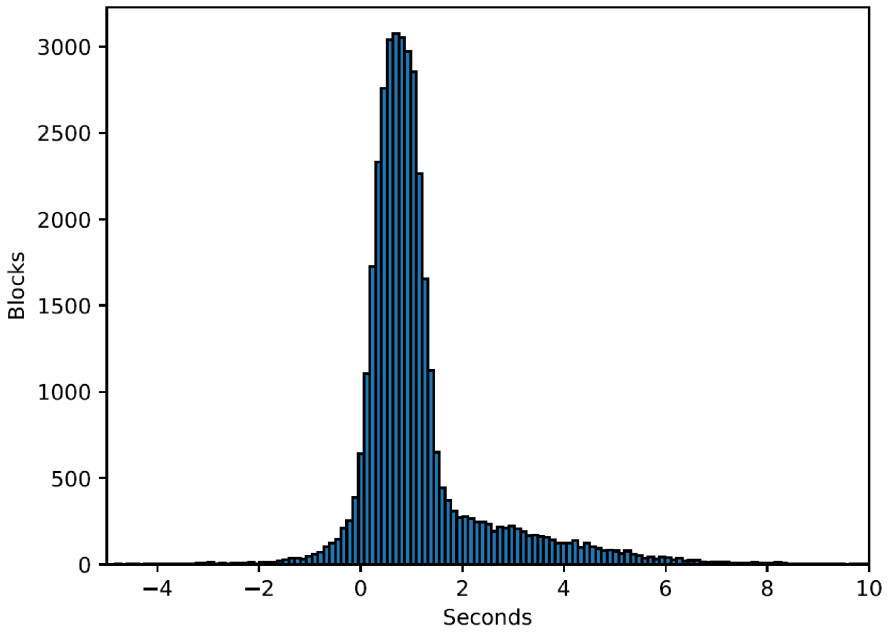

Figura 17 – Distribuição de frequência dos blocos pela diferença entre o momento em que o primeiro dos quatro nós recebeu um bloco e o timestamp do bloco seguinte.

blocos a todo momento então essa diferença deveria ser próxima ao tempo de mineração dos blocos, em torno de 13 segundos em média. Isso indica que os mineradores definem os timestamps dos blocos antes de começarem a mineração e não atualizam o timestamp durante a mineração. Logo, descoberto o momento em que os mineradores atribuem o timestamp e o fato de que novas transações podem ocasionalmente serem inseridas no bloco durante a mineração explica porque ocorreram transações inseridas em blocos com tempo no passado. E isso também explica que o fato mostrado pode ocorrer mesmo os mineradores atuando honestamente.

8 AVALIAÇÃO DE CUSTOS

O custo dos carimbos do tempo é um fator importante para as aplicações que os utilizam. O custo de um carimbo do tempo para um registro de direitos autorais pode ser irrelevante em comparação ao benefício que traz, por exemplo no caso do NFT mostrado no início da introdução. Contudo, para aplicações com grandes volumes de dados, por exemplo IoT, o custo de um carimbo do tempo deve ser considerado. Não cabe aqui discutir sobre a viabilidade das aplicações em custear o carimbo do tempo, porém traz-se uma estimativa dos custos dos carimbos do tempo providos pelo modelo proposto, para que depois possam ser analisados caso a caso.

O custo de um carimbo do tempo no modelo proposto é composto de duas parcelas: 1) os custos computacionais da rede Ethereum, e; 2) os custos para o mantenimento do serviço de carimbo do tempo. Na Seção 8.1 apresentam-se os custos computacionais na rede Ethereum. Este custo é referente a execução das operações e armazenamento dos dados pelos mineradores da rede, e na Seção 8.2 apresentam-se os valores das taxas adicionais que devem ser pagas como incentivo para os provedores participarem ativamente e corretamente do serviço de carimbo do tempo. Na Seção 8.2 também avaliam-se os valores dos depósitos que devem ser feitos pelo requerente e pelos provedores para prevenção de abusos e ataques.

8.1 CUSTOS COMPUTACIONAIS NA REDE ETHEREUM

O protocolo proposto na Seção 5.2 exige que uma série de transações sejam disparadas pelo requerente e pelos provedores. Estas transações são processadas pelos mineradores e geram custos computacionais. Estes custos são referentes a execução das operações descritas no contrato inteligente e armazenamento dos dados na blockchain.

Os custos de execução das operações e o armazenamento de dados são valores determinísticos descritos em *gas* no Yellow Paper Ethereum. O *gas* total de uma transação é a soma de todas as operações e armazenamentos de dados. O custo total em Ether de uma transação depende do *gas price* pago por ela. O total em Ether é igual ao *gas* $\times$ *gas price* (o total consumido em *gas* pela transação multiplicado pelo *gas price* pago). Por fim, o custo em moeda fiduciária depende do valor de câmbio, por exemplo entre o Ether e o Dólar.

A IDE Remix[1] pode ser utilizada para computar o total consumido em *gas* por uma transação. Contudo, apesar de os custos em *gas* serem determinísticos (teoricamente não mudarem com o tempo), o total em *gas* pode mudar sutilmente entre uma execução e outra. Isso acontece devido a algumas instruções utilizarem estruturas de dados, como mapas chave-valor e listas. Especialmente na primeira alocação em uma estrutura de dados o *gas* consumido é um pouco maior. Por conta disso, utilizou-se como base os dados do experimento da Seção 6.3. O experimento conduziu o protocolo da Seção 5.2 por completo, portanto permitiu aferir o *gas*

[1] https://remix.ethereum.org/

utilizado pelas quatro funções principais do contrato inteligente: *requestTimestamp*, *addCommitment*, *addDecommitment* e *selectTimestamp*. O custo em *gas* para cada uma das funções pode ser visto na Tabela 8.

Função	Média (*gas*)	Desvio Padrão
requestTimestamp	135.366	23
addCommitment	83.872	10.763
addDecommitment	63.192	4.879
selectTimestamp	174.593	16.225

Tabela 8 – Custo computacional em gas para a execução das funções.

O contrato inteligente possui outras funções que também adicionam custos ao serviço, como por exemplo para juntar-se ao serviço cadastrando-se como provedor e sacar fundos recebidos como recompensas. Contudo, estes tipos de funções não estão diretamente relacionados com a solicitação de carimbo do tempo. Um provedor pode juntar-se ao serviço e participar de várias solicitações de carimbos do tempo, diluindo os custos esporádicos. Por conta disso, o custo destas funções não foram contabilizados no custo do carimbo do tempo.

O custo em *gas* é atemporal, mas o custo em Ether e em moeda fiduciária podem variar conforme o *gas price* e o câmbio. O custo computacional final ainda depende do número de provedores participantes. As funções *requestTimestamp* e *selectTimestamp* são executadas pelo requerente apenas uma vez a cada solicitação. Enquanto as funções *addCommitment* e *addDecommitment* são executadas por cada provedor. Portanto, a fórmula final para o custo computacional C aproximado para o carimbo do tempo tem a seguinte forma

$$C(\textit{gas price}, \textit{câmbio}, I) = \textit{câmbio} \times \textit{gas price} \times (309.959 + I \times 147.064)$$

sendo I o número de provedores. O valor obtido será em moeda fiduciária, na qual o câmbio está convertendo.

Para exemplificar, calculou-se o valor em dólares para o dia 11 de junho de 2021. O *gas price* foi obtido através do *gas price average* sugerido pelo ETH Gas Station, que foi de 10 Gwei. E o valor do câmbio foi de 2470,70 U$/Ether. Com esses valores, o total para o requerente é de U$7,66, enquanto para cada um dos provedores é de U$3,63.

Além do custo computacional do carimbo do tempo, ainda devem ser adicionadas as taxas pagas para o serviço de carimbo do tempo, que serão vistas na próxima seção. Contudo, o custo computacional do carimbo do tempo pode ser entendido como um limite inferior. O custo computacional pode variar significativamente, por conta da flutuação do *gas price* e do câmbio com a moeda fiduciária. A discussão sobre o custo do carimbo do tempo será retomada no Capítulo 9.

8.2 TAXAS E DEPÓSITOS

As taxas pagas pelos requerentes para o serviço de carimbo do tempo são utilizadas para o mantenimento do serviço de carimbo do tempo. O valor das taxas é distribuído entre os provedores como recompensa por participarem do serviço. As taxas devem ser suficientes para custear as despesas dos provedores e um adicional de incentivo. Esse incentivo é o lucro propriamente dito que os provedores terão ao participar do serviço.

Os depósitos, por sua vez, são utilizados como uma garantia de participação honesta. O depósito feito pelo requerente a cada solicitação de carimbo do tempo tem o objetivo de evitar abusos, por exemplo, sobrecarregar o contrato com muitas solicitações. Assim como, também é um incentivo para que o requerente reivindique a seleção do carimbo do tempo após os commitments e decommitments. Ao reivindicar a seleção do carimbo do tempo, o requerente tem o seu depósito devolvido e dispara a distribuição de recompensas para os provedores. Após o prazo estabelecido qualquer outro participante poderá reivindicar a seleção do carimbo do tempo, caso ainda não tenha sido feita, o que também dispara a distribuição de recompensas para os provedores. Contudo, nesse caso, o requerente perde o depósito, que é entregue a quem reivindicou a seleção do carimbo do tempo.

Os provedores também devem realizar depósitos para participar do serviço. No caso dos provedores, o depósito é único, realizado durante o credenciamento do provedor no contrato inteligente. Após realizar o depósito o provedor poderá participar de um número ilimitado de solicitações de carimbo do tempo. Contudo, é permitido fornecer apenas um timestamp por solicitação de carimbo do tempo. Isso desincentiva a execução de ataques Sybil, pois para fornecer múltiplos timestamps é necessário realizar múltiplos depósitos. Além disso, ao estar credenciado no contrato inteligente mas não participar de uma solicitação de carimbo do tempo o provedor corre o risco de ter prejuízos. O prejuízo pode acontecer se uma solicitação não tiver um número suficiente de timestamps fornecidos. Nesse caso, os provedores que não enviaram seus timestamps devem pagar os custos computacionais daqueles que enviaram. Quando isso acontece, esse valor é debitado do depósito do provedor automaticamente pelo contrato inteligente. O provedor pode inclusive ficar desabilitado de fornecer timestamps se o depósito restante for inferior ao valor mínimo. Esse mecanismo é um incentivo para os provedores credenciados participarem ativamente das solicitações de carimbo do tempo.

Além disso, os provedores devem levar em consideração o custo para manter uma infraestrutura que permita fornecer os timestamps adequadamente. Uma opção é utilizar uma instância da Amazon EC2, como a m4.large que já foi testada para a função. Os experimentos do Capítulo 6 mostraram que uma instância da m4.large é suficiente para o trabalho. O custo médio entre as cinco regiões utilizadas no experimento do Capítulo 6 foi de U$640,53 por ano para cada instância (Amazon AWS Services, 2021).

Por fim, o último fator que os provedores devem considerar é o prêmio de risco. Como já foi dito, o depósito realizado pelos provedores tem como um dos objetivos evitar ataques Sybil. Ou seja, o valor deve ser o suficiente para evitar que o provedor crie múltiplas identidades.

Por conta disso, o depósito fica bloqueado no contrato inteligente durante todo o período em que o provedor estiver engajado no serviço. Esse valor poderia ser aplicado em um investimento de baixo risco, por exemplo um título do Tesouro Americano com retorno em torno de 1 à 2% no prazo de um ano. No entanto, o retorno de investimento no serviço depende da demanda por carimbo do tempo e do engajamento dos participantes. Portanto, é necessário haver um prêmio de risco, ou seja, um incentivo para os provedores participarem do serviço. Como o depósito está sujeito também a volatilidade da criptomoeda a qual fica atrelado, é justo ter um prêmio de risco próximo ao retorno de um investimento de renda variável. A NYSE (New York Stock Exchange), por exemplo, em maio de 2021 acumulou um retorno médio de 11,25% por ano nos últimos 5 anos (NYSE, 2021). Portanto, um valor atrativo para o prêmio de risco seria algo um pouco maior do que isso, em torno de 15% ao ano.

Tanto o custo com a infraestrutura quanto o prêmio de risco pago são diluídos conforme o número de carimbos do tempo realizados. Para o período de um ano com custo de U$640,53 com infraestrutura e prêmio de risco de 15% anual sobre o depósito, o custo final das taxas do serviço por carimbo em dólares seria de

$$T(N,E) = \frac{640,53 + N \times 0,15}{E}$$

sendo N o depósito do provedor e E o número anual de carimbos do tempo fornecidos pelo serviço. Por simplificação, esta fórmula considera que os provedores sempre fornecem seus timestamps quando há solicitações e que sempre acertam o timestamp dentro da janela de seleção.

Os depósitos pagos pelos requerentes não são considerados no preço final do carimbo do tempo, pois ficam bloqueados por pouco tempo, apenas entre a solicitação e seleção do carimbo. Portanto, o preço final (custo computacional + taxa destinada aos provedores) em dólares de um carimbo do tempo é dado por

$$P(\textit{gas price}, \textit{câmbio}, I, N, E) = C(\textit{gas price}, \textit{câmbio}, I) + I \times T(N,E)$$

$$= \textit{câmbio} \times \textit{gas price} \times (309.959 + I \times 147.064) + I \times \frac{640,53 + N \times 0,15}{E}$$

sendo I número de provedores, N o valor do depósito dos provedores e E o número de carimbos do tempo fornecidos anualmente pelo serviço.

A Tabela 9 mostra um exemplo de como o preço do carimbo do tempo é diluído conforme o número anual de carimbos do tempo fornecidos pelo serviço. A tabela também mostra a distribuição do custo entre os mineradores (custo computacional), infraestrutura e prêmio de risco. Para isso, considera-se o $I = 5$, N =U$10.000 e $E = 10, 100, ..., 1.000.000$ ao ano. Observa-se que quanto maior o número de carimbos, menor o custo do carimbo. Assim como, os custos com os mineradores passam a ser mais relevantes conforme o número de carimbos aumenta.

Carimbo do Tempo		Distribuição de Custos (% do Preço)		
Quant./Ano	Preço (U$)	Custo Computacinal	Infraestrutura	Prêmio de Risco
10	1096,09	2,36	29,22	68,43
100	132,85	19,44	24,11	56,45
1.000	36,53	70,70	8,77	20,53
10.000	26,89	96,02	1,19	2,79
100.000	25,93	99,59	0,12	0,29
1.000.000	25,84	99,96	0,01	0,03

Tabela 9 – Um exemplo do preço pago pelo requerente por um carimbo do tempo em função do número anual de carimbos do tempo fornecidos pelo serviço. Considera-se cinco provedores, depósito de U$ 10.000 por provedor, infraestrutura Amazon EC2 m4.large e prêmio de risco de 15% ao ano. O custo computacional é referente ao dia 11 de junho de 2021, com um *gas price* de 10 Gwei e câmbio 2470,70 U$/Ether.

9 DISCUSSÃO

Neste capítulo serão discutidos os principais fatores relacionados ao modelo de carimbo do tempo proposto. Os fatores apontados são a latência de rede, o tempo de confirmação, a janela de tempo, a acurácia, os custos e os depósitos. Também discute-se sobre as vantagens do modelo proposto em relação aos outros modelos de carimbos do tempo baseados em blockchain e aos carimbos do tempo padrão RFC 3161.

9.1 LATÊNCIA DE REDE

A latência da rede foi aferida através dos tempos de propagação das transações no experimento da Seção 6.2. O experimento apontou uma latência média de 188,35 milissegundos. Acredita-se que a latência em outros períodos não sofre grandes alterações em relação ao período do experimento. Isso porque a rede Ethereum possui uma dinâmica auto-regulatória de oferta (de recursos) e demanda (de transações). Se o volume de transações aumentar, a taxa média paga pelas transações também aumenta, forçando os usuários a procurarem alternativas. Por outro lado, se as taxas pagas pelas transações são maiores torna-se mais atrativo para os mineradores, incentivando o aumento de recursos da rede. O número de transações pendentes na rede Ethereum (ETHERSCAN, 2021f) permanece aproximadamente em algumas centenas de milhares de transações, enquanto o número de transações diárias (ETHERSCAN, 2021c) atinge patamares muito maiores. Isso mostra que a rede possui capacidade computacional para processar as transações que oferecem taxas justas. Parte das transações permanecem pendentes por não serem vantajosas para os mineradores ou por terem dependências de outras transações (na blockchain Ethereum as transações possuem um *nonce* que estabelece a ordem de execução das transações emitidas por um mesmo endereço).

Além disso, o valor encontrado para a latência da rede é próximo ao limite de latência das redes de computadores (KUROSE, 2017). O volume de dados gerados, por sua vez, não é um fator limitante. Diariamente são gerados algumas centenas de megabytes de dados na rede, considerando que um novo bloco é gerado aproximadamente a cada 13 segundos e cada bloco contém algumas dezenas de kilobytes (ETHERSCAN, 2021a). Contudo, é um volume muito pequeno comparado aos mais de 2.5 quintilhões de bytes gerados diariamente na Internet (AHMAD, 2018).

9.2 TEMPO DE CONFIRMAÇÃO

Com base nos resultados do experimento da Seção 6.2 estabeleceu-se o prazo de 180 segundos para a confirmação das transações no protocolo. Este prazo inclui 91% das transações do primeiro experimento. Na avaliação realizada no experimento da Seção 6.3, este prazo mostrou-se suficiente para a confirmação de todas as transações realizadas. Como já dito anteriormente, o tempo de confirmação depende diretamente da taxa paga pela transação. O valor

da taxa utilizada foi a sugerida pelo ETH Gas Station para confirmação em até cinco minutos. Apesar de o parâmetro utilizado ser de cinco minutos, o tempo médio aferido foi muito inferior (vide Tabela 7). Por isto, a escolha foi de 180 segundos ao invés dos 300 segundos sugeridos. Isso porque quanto maior o prazo estabelecido no contrato inteligente, maior a espera até o requerente obter o carimbo do tempo. Por outro lado, ao utilizar o ETH Gas Station para sugestão da taxa paga não é necessário preocupar-se com o tempo de confirmação, visto que o valor sugerido se adapta à demanda de transações na rede. Apesar disso, o contrato inteligente é parametrizado para ajustes nos prazos do protocolo.

9.3 JANELA DE TEMPO

Assim como o tempo de confirmação, a janela de tempo foi definida conforme os resultados do experimento da Seção 6.2. O tamanho escolhido para a janela foi de 1 segundo, que inclui 90% das transações do primeiro experimento. Novamente, uma avaliação foi realizada no experimento da Seção 6.3. O método selecionou o melhor timestamp em 20 dos 21 casos e em um dos casos identificou que os timestamps eram discrepantes. Nesse caso identificado, a amplitude das amostras foi de quase quatro segundos.

No experimento da Seção 6.3, os provedores foram programados para fornecer o timestamp do momento em que viram a solicitação pela primeira vez, sem manipular o tempo. Mesmo assim, como pode ser visto, aconteceu de timestamps não enquadrarem-se na janela. Isso mostra que a escolha do tamanho da janela requer atenção. Por um lado, quanto maior a janela maiores as chances de incluir timestamps discrepantes. Por outro lado, quanto menor a janela maiores as chances de não incluir os timestamps corretos. Uma estratégia possível é adaptar o tamanho da janela conforme algum critério. Por exemplo, aumentar o tamanho da janela caso muitas solicitações falhem por não atingirem timestamps dentro da janela. Contudo, é necessário também reduzir sistematicamente o tamanho da janela para que ela não aumente demasiadamente.

Essa estratégia poderia ter a seguinte forma. A cada 100 solicitações de carimbo do tempo o contrato inteligente verifica automaticamente o número de solicitações que falharam por não atingir o número necessário de timestamps dentro da janela. Se mais de 25% das solicitações falharam o contrato inteligente aumenta o tamanho da janela em 50 milissegundos. E, se menos de 5% das solicitações falharam o contrato inteligente diminui o tamanho da janela em 50 milissegundos.

Deve-se lembrar que os timestamps podem cair fora da janela por realmente serem discrepantes. Contudo, esses casos devem acontecer apenas em momentos de instabilidade da rede, voltando a atingir baixa latência quando a instabilidade passar. Por outro lado, os timestamps também podem cair fora da janela em tentativas de ataques. Para isso, a estratégia de seleção apresentada na Seção 5.3 desincentiva essa prática com aplicação de punições.

9.4 ACURÁCIA

A acurácia dos carimbos do tempo produzidos com o modelo proposto depende apenas do tempo de propagação das transações na rede. O experimento da Seção 6.3 aferiu uma acurácia de aproximadamente 121 milissegundos. Esse tempo é centenas de vezes menor do que a acurácia dos carimbos do tempo que utilizam os tempos dos blocos para ancoramento, como mostrado no Capítulo 4. Contudo, ainda é distante da acurácia de milissegundos almejada por Broby, Basu e Arulselvan (2019). Poderia-se considerar a possibilidade de a proposta ser implementada em uma rede blockchain permissionada. Nas blockchain permissionadas os participantes são identificados, estabelecendo conexões mais estáveis e com um número menor de saltos. Apesar disso, dificilmente irá alcançar-se acurácia de ordens menores que milissegundos, devido a latência imposta pelas redes de computadores (KUROSE, 2017). Adicionalmente, se esse limite fosse superado, um outro limite seria imposto pela capacidade computacional. Por exemplo, uma assinatura digital ECDSA de 256 bits, utilizada na plataforma Ethereum, consome cerca de 350 nanosegundos para ser computada em um computador com processador Intel Core i5 dual core de 2.6 GHz de acordo com o benchmark do OpenSSL (OpenSSL Software Foundation, 2018).

9.5 CUSTOS

O custo final do carimbo do tempo com o modelo proposto é de algumas dezenas de dólares, como pôde ser visto no Capítulo 8. A partir de mil carimbos do tempo fornecidos pelo serviço, o preço final pouco varia. Quanto maior o número de carimbos do tempo fornecidos pelo serviço mais relevante é a parcela referente ao custo computacional. Visto que o custo computacional é um valor fixo por solicitação. Por outro lado, o custo com a infraestrutura e o prêmio de risco é diluído conforme aumenta o número de carimbos do tempo fornecidos. Evidentemente, não se sabe inicialmente quantos carimbos do tempo serão produzidos durante um ano. Da mesma forma, os requerentes podem não querer se sujeitarem a pagar centenas de dólares pelo carimbo do tempo. Portanto, inicialmente os provedores devem arcar com parte dos custos com a infraestrutura e o prêmio de risco, na expectativa de terem retornos conforme fornecerem mais carimbos do tempo. Após um ano, por exemplo, pode-se tomar como base o histórico de carimbos do tempo emitidos no período para estimar o valor a ser pago pelo requerente. Vale lembrar que os valores indicados na Tabela 9 são referentes ao custo bruto do carimbo do tempo, cabe ao requerente oferecer um adicional extra para incentivar o serviço. Quanto maior o valor que o requerente está disposto a pagar pelo carimbo do tempo mais interesse terão os provedores em fornecer o carimbo do tempo. Da mesma forma que ocorre na plataforma Ethereum, onde os usuários adicionam às transações uma taxa de incentivo para os mineradores.

Contudo, se comparado aos carimbos do tempo RFC3161 o custo obtido com o modelo proposto é muito superior. Carimbos do tempo RF3161 podem ser obtidos por alguns centavos

de dólares (DigiStamp, Inc, 2020). Essa diferença representa o custo para obter-se um carimbo do tempo descentralizado em uma plataforma blockchain pública de grande abrangência como a Ethereum e a garantia de acurácia provida pelo modelo. Visto que a parcela mais significativa é destinada aos mineradores da rede (custo computacional na Tabela 9). Para diminuir os custos poderia-se reduzir o número de transações do protocolo. Isso pode ser feito de duas formas, ambas utilizando comunicação *off-chain* (fora da rede principal) entre os participantes.

Na primeira forma, os provedores enviam os decommitments de uma solicitação de carimbo do tempo para o requerente ao invés de enviar ao contrato inteligente (passo 4 do protocolo da Seção 5.2). E então o requerente envia ao contrato inteligente uma única transação contendo todos os decommiments. Como o Ethereum estabelece um custo mínimo fixo de 21.000 *gas* para cada transação a fim prevenir abusos na rede, esta estratégia permite reduzir custos. Mais precisamente, em testes empíricos com cinco provedores fornecendo timestamps o custo foi reduzido em aproximadamente 20%. O custo poderia ser reduzido ainda mais se o mesmo procedimento for aplicado aos commitments (passo 3 do protocolo). Contudo, isso abriria a possibilidade de o requerente boicotar provedores ignorando os commitments sem levantar suspeita.

A segunda forma consiste em utilizar uma estratégia baseada em *general state channels* (DZIEMBOWSKI; FAUST; HOSTÁKOVÁ, 2018), às vezes também chamado de *side-channels*. Nessa estratégia, os participantes aderem a um protocolo off-chain que executa uma cópia local do contrato inteligente e compartilham o estado do contrato inteligente trocando mensagens assinadas digitalmente com os outros participantes. A qualquer momento os participantes podem registrar o estado atual dos seus contratos inteligentes no contrato inteligente original na blockchain. Em caso de divergência entre os estados locais, o contrato inteligente original pode resolver a disputa. Isso poderia ser implementado com uma rede blockchain permissionada, como o Hyperledger Fabric[1]. Todo o processo desde a solicitação de carimbo do tempo até a seleção do melhor carimbo poderia ser feito na rede auxiliar. Apenas após a seleção do carimbo do tempo é que este seria registrado na blockchain principal.

Apesar de a abordagem reduzir significativamente os custos da proposta, deve-se atentar a possibilidade de conluio entre os participantes. Visto que os participantes podem executar o protocolo incorretamente e concordar com timestamps errados. Parte da confiança é perdida quando abre-se mão de executar todo o protocolo de forma pública e transparente. Quando a plataforma blockchain não utiliza estratégia de incentivo financeiro para a execução do protocolo, como é o caso das blockchain permissionadas, a confiança é depositada sobre os participantes. No caso de um serviço de carimbo do tempo como o proposto, os provedores que atuariam na rede auxiliar poderiam ser as próprias ACTs. Essa abordagem híbrida entre serviços confiáveis de carimbo do tempo e serviços de carimbo do tempo em blockchain poderia agregar as vantagens de cada modelo. Esse tópico pode ser tema de trabalhos futuros.

[1] https://www.hyperledger.org/use/fabric

9.6 DEPÓSITOS

Uma questão importante é quanto aos depósitos realizados, principalmente os que são pagos pelos provedores. O valor do depósito pago pelos provedores pouco afeta o custo final do carimbo do tempo quando o número de solicitações no serviço aumenta. No entanto, o valor do depósito implica na dificuldade de realizar-se um ataque Sybil. Mais especificamente, para criar multiplas identidades e fornecer vários timestamps em uma mesma solicitação é necessário que um provedor realize múltiplos depósitos, um para cada timestamp fornecido. Isso porque cada depósito dá o direito de fornecer apenas um timestamp.

De forma geral, funciona como um *proof-of-stake* (NGUYEN et al., 2019), ou seja, quanto mais recursos (financeiros nesse caso) o provedor empenhar, maior sua influência no sistema. Se o número de participantes for grande, a quantidade total de recursos empenhados no sistema também será. Portanto, individualmente os provedores terão uma influência proporcionalmente baixa. Mas se o número de participantes for pequeno, a descentralização do sistema pode ficar comprometida.

Na prática, para manter a segurança do sistema, o custo de forjar um carimbo do tempo falso deve ser maior do que o benefício que este pode trazer para um provedor desonesto. Para um participante fornecer um número de timestamps o suficiente para conseguir manipular o timestamp escolhido, a soma dos depósitos realizados deve ser maior que o ganho, por exemplo, de registrar uma arte digital com tempo no passado. Contudo, na maioria dos cenários (como registros de direitos autorais ou leilões online) é difícil prever o benefício que o provedor desonesto pode ter com um carimbo do tempo falso.

Para mitigar isso, poderia-se criar níveis de segurança para os carimbos do tempo. Provedores com depósitos maiores forneceriam carimbos do tempo mais confiáveis e poderiam receber maiores recompensas por isso. Dessa forma, ficaria a critério da aplicação exigir um determinado nível de segurança para o carimbo do tempo. Poderia-se também fornecer informações de quais provedores participaram de uma solicitação e quanto de recurso foi empenhado, para permitir uma avaliação posterior. Além disso, outros protocolos poderiam ser combinados com o esquema de *proof-of-stake* para adicionar imprevisibilidade. Por exemplo, determinar aleatóriamente quais provedores participaram de um solicitação.

Tem-se em mente que este é um ponto crítico, pois tem influência na descentralização do sistema. Apesar disso, o protocolo que gera o carimbo do tempo é executado sobre um contrato inteligente, portanto é independente do protocolo da rede blockchain em si. Dessa forma, é possível adotar diferentes esquemas para a correta participação dos envolvidos, como esquemas de reputação por exemplo.

9.7 VANTAGENS

Os experimentos realizados no Capítulo 6 indicam que os carimbos do tempo produzidos com o modelo proposto são potencialmente mais confiáveis do que os carimbos do tempo

que utilizam os tempos dos blocos. Comparando o tempo que uma transação foi criada e o timestamp aplicado pelo minerador que criou o bloco o qual a transação foi inserida, observou-se que o timestamp algumas vezes era anterior ao momento em que a transação foi criada. Isso acontece porque os mineradores podem adicionar transações a um bloco que está sendo minerado e não atualizar o timestamp desse bloco, como foi visto no Capítulo 7. Portanto, ao pagar uma taxa suficiente para a transação ser inserida no próximo bloco é possível ancorar uma transação com tempo no passado. Essa é uma falha grave para um serviço de carimbo do tempo. O modelo proposto não é suscetível a essa falha, pois não utiliza o tempo dos blocos. Ao invés de confiar nos tempos dos blocos, o modelo utiliza um consórcio de provedores de tempo que são incentivados a atuarem corretamente, sujeitos a sofrerem prejuízos caso não o façam. Além disso, o contrato inteligente possui um mecanismo para filtrar timestamps discrepantes e selecionar o melhor timestamp.

Entretanto, o modelo proposto possui um atraso maior para conclusão do protocolo. Enquanto os métodos que utilizam o tempo do bloco obtém o carimbo do tempo com uma única transação, no modelo proposto é necessário o tempo para confirmação de pelo menos três transações uma após a outra (commitment, decommitment e seleção). Portanto, enquanto o primeiro pode ser alcançado com o tempo necessário para a criação de um único bloco, o segundo necessita da criação de múltiplos blocos. Além disso, os métodos que utilizam o tempo do bloco também possuem um menor custo por utilizem apenas uma transação.

Em relação aos serviços de carimbo do tempo confiáveis, que oferecem carimbos do tempo RFC3161, o modelo proposto é vantajoso em dois aspectos. Os carimbos do tempo RFC3161 permanecem seguros apenas enquanto os algoritmos de assinatura, os comprimentos de chave e as funções de hash também permanecerem seguros (VIGIL et al., 2015). Enquanto, os carimbos do tempo baseados em blockchain dependem apenas da função de hash utilizada. Pelo fato de os comprimentos de chaves terem um tempo de vida menor que os algoritmos de funções de hash, espera-se que os carimbos do tempo baseados em blockchain permaneçam seguros por mais tempo que os carimbos do tempo RFC3161.

O segundo aspecto é referente a resistência ao comprometimento da chave. Se uma ACT que fornece carimbos do tempo RFC3161 é comprometida, qualquer carimbo do tempo que foi emitido pela ACT não é mais confiável. Esse problema acontece porque não é possível distinguir entre carimbos do tempo que foram emitidos antes da chave da ACT ter sido descoberta de um carimbo do tempo falso emitido depois de a chave da ACT ter sido descoberta. Da mesma forma, um problema similar surge quando a autoridade certificadora que fornece o certificado digital para a ACT é comprometida. Um certificado digital falso pode ser utilizado para assinar carimbos do tempo no passado em nome da ACT. Esse problema é bem conhecido e pode ser mitigado estabelecendo a ordem relativa dos carimbos do tempo através de uma lista encadeada baseada em hash, similar a uma blockchain. Mais precisamente, uma ACT adiciona a cada carimbo do tempo o hash do carimbo do tempo anterior (HABER; STORNETTA, 1991). Adicionalmente, alguns dos carimbos do tempo da lista encadeada podem ser publicados em um meio público de grande alcance, por exemplo um jornal (Surety, LLC, 2003). Desta forma,

se uma ACT for comprometida, ainda é possível confiar nos carimbos do tempo emitidos por ela por conta da ordem relativa, que permite ajudar a identificar quais carimbos do tempo não são retroativos. Maniatis e Baker (2002) propuseram que duas ou mais ACTs entrelaçam suas listas encadeadas, garantindo a ordem relativa de carimbos do tempo mesmo que reste apenas uma ACT não comprometida.

No modelo de carimbo do tempo proposto este problema não é preocupante. Se um provedor é comprometido mas a maioria dos provedores ainda é confiável então os carimbos do tempo não serão afetados. Além disso, o modelo proposto não necessita de garantias de ordem relativas ou um meio público de grande alcance, pois a própria blockchain provê isso. No entanto, embora foi mostrado que o timestamps dos blocos podem ser imprecisos, até certo ponto, podem ser utilizados para verificar os carimbos do tempo criados com o modelo proposto.

Por fim, o modelo proposto não depende de um algoritmo de consenso específico executado pela rede blockchain. Isso permite que o modelo possa ser implementado em outras plataformas blockchains, desde que suportem contratos inteligentes e *broadcast* de transações. No entanto, o *broadcast* de transações deve ser justo para que atinja todos os provedores com a mesma chance, sem priorizar um ou outro.

10 CONCLUSÕES

Este trabalho investigou a acurácia dos carimbos do tempo em blockchain. Adicionalmente, propõe-se um modelo de carimbo do tempo utilizando contratos inteligentes com maior acurácia. O estudo de trabalhos relacionados mostrou que muitas propostas de serviço de carimbo do tempo em blockchain confiam demasiadamente nos tempos dos blocos fornecidos pelos mineradores. A análise realizada indicou que com esse método a acurácia dos carimbos do tempo é de no mínimo algumas dezenas de segundos. Além disso, descobriu-se que os mineradores podem equivocadamente ancorar transações com tempo no passado, o que é uma falha grave para serviços de carimbo do tempo.

O modelo proposto contorna esses problemas ao não confiar nos tempos dos blocos para o ancoramento temporal dos carimbos do tempo. Ao invés disso, utiliza uma estratégia onde provedores de tempo declaram o momento em que receberam uma transação com a solicitação de carimbo do tempo. Os provedores são incentivados a atuarem corretamente através de incentivos financeiros. Aliado a isso, um mecanismo identifica timestamps discrepantes fornecidos pelos provedores e seleciona o melhor timestamp para a solicitação de carimbo do tempo. Os experimentos aferiram uma acurácia média de 121 milissegundos para os carimbos do tempo obtidos com o modelo proposto. Este resultado é muito superior à acurácia dos carimbos do tempo em blockchain que utilizam os tempos dos blocos. Além disso, é um valor próximo ao limite imposto pela latência das redes de computadores. Dificilmente serviços descentralizados obterão resultados significativamente melhores do que este com as redes de computadores atuais.

No entanto, obter um carimbo do tempo descentralizado e com acurácia impôs um custo considerável. O custo dos carimbos do tempo com o modelo proposto é significativamente maior que os carimbos do tempo RFC3161 e também que os carimbos do tempo em blockchain que utilizam os tempos dos blocos. Para mitigar isso foram levantadas algumas possibilidades. Uma delas é a utilização de redes blockchains permissionadas. Essa abordagem permite reduzir significativamente os custos mas abre mão de parte da confiança do modelo. Neste caso, a confiança passa a ser depositada sobre os participantes. Em trabalhos futuros, almeja-se aliar os benefícios dos carimbos do tempo em blockchain e dos serviços de carimbo do tempo confiáveis. Para isso, idealiza-se um modelo híbrido entre blockchains públicas e permissionadas, mantendo a descentralização e a acurácia e reduzindo os custos.

REFERÊNCIAS

ABADI, A. et al. Timed signatures and zero-knowledge proofs—timestamping in the blockchain era—. In: CONTI, M. et al. (Ed.). **Applied Cryptography and Network Security**. Cham: Springer International Publishing, 2020. p. 335–354. DOI:10.1007/978-3-030-57808-4_17.

ABID, A.; CHEIKHROUHOU, S.; JMAIEL, M. Modelling and executing time-aware processes in trustless blockchain environment. In: KALLEL, S. et al. (Ed.). **Risks and Security of Internet and Systems**. Cham: Springer International Publishing, 2020. p. 325–341. DOI:10.1007/978-3-030-41568-6_21.

AHMAD, I. **How Much Data Is Generated Every Minute? [Infographic]**. 2018. Acesso em: 13 de junho de 2021. Disponível em: https://www.socialmediatoday.com/news/how-much-data-is-generated-every-minute-infographic-1/525692/.

ALSUNAIDI, S. J.; ALHAIDARI, F. A. A survey of consensus algorithms for blockchain technology. In: **2019 International Conference on Computer and Information Sciences (ICCIS)**. Sakaka, Saudi Arabia: IEEE, 2019. p. 1–6. DOI:10.1109/iccisci.2019.8716424.

Amazon AWS Services. **Configure Amazon EC2**. 2021. Acesso em: 29 de maio de 2021. Disponível em: https://calculator.aws/#/createCalculator/EC2.

BADERTSCHER, C. et al. **Ouroboros chronos: Permissionless clock synchronization via proof-of-stake**. 2019. Acesso em: 20 de maio de 2021. Disponível em: https://eprint.iacr.org/2019/838.pdf.

BANIATA, H.; ANAQREH, A.; KERTESZ, A. PF-BTS: A privacy-aware fog-enhanced blockchain-assisted task scheduling. **Information Processing & Management**, Elsevier BV, v. 58, n. 1, p. 102393, jan. 2021. DOI:10.1016/j.ipm.2020.102393.

BERTONI, G. et al. **Keccak Team**. 2018. Acesso em: 19 de maio de 2021. Disponível em: https://keccak.team/index.html.

Bitcoin Community. **Bitcoin Core integration**. Bitcoin, 2020. Acesso em: 19 de maio de 2021. Disponível em: https://github.com/bitcoin/bitcoin.

BLAZIC, A. J.; GONDROM, T.; SALJIC, S. **Extensible Markup Language Evidence Record Syntax (XMLERS)**. RFC Editor, 2011. RFC 6283. (Request for Comments, 6283). Disponível em: https://rfc-editor.org/rfc/rfc6283.txt.

BROBY, D.; BASU, D.; ARULSELVAN, A. The role of precision timing in stock market price discovery when trading through distributed ledgers. **Journal of Business Thought**, Informatics Publishing Limited, v. 10, n. 1, p. 1–8, mar. 2019. DOI:10.18311/jbt/2019/23355.

CACHIN, C. et al. **Architecture of the hyperledger blockchain fabric**. 2016. 4 p. Acesso em: 19 de maio de 2021. Disponível em: https://www.zurich.ibm.com/dccl/papers/cachin_dccl.pdf.

CHEN, Q. et al. An incentive-aware blockchain-based solution for internet of fake media things. **Information Processing & Management**, Elsevier BV, p. 102370, ago. 2020. DOI:10.1016/j.ipm.2020.102370.

CHRISTIDIS, K.; DEVETSIKIOTIS, M. Blockchains and smart contracts for the internet of things. **IEEE Access**, v. 4, p. 2292–2303, 2016. DOI:10.1109/ACCESS.2016.256633.

COINMARKETCAP. **Today's Cryptocurrency Prices by Market Cap**. 2021. Acesso em: 20 de maio de 2021. Disponível em: https://coinmarketcap.com/.

DigiStamp, Inc. **DigiStamp: Price of Digital Timestamps from a Trusted Timestamp Authority**. 2020. Acesso em: 14 de junho de 2021. Disponível em: https://www.digistamp.com/subpage/price.

DOUCEUR, J. R. The sybil attack. In: **Peer-to-Peer Systems**. Berlin Heidelberg: Springer, 2002. p. 251–260. DOI:10.1007/3-540-45748-8_24.

DZIEMBOWSKI, S.; FAUST, S.; HOSTÁKOVÁ, K. General state channel networks. In: LIE, D. et al. (Ed.). **Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security**. Toronto Canada: ACM, 2018. p. 949–966. DOI:10.1145/3243734.3243856.

ENTRIKEN, W. et al. **ERC-721 Non-Fungible Token Standard**. 2018. Acesso em: 19 de maio de 2021. Disponível em: https://eips.ethereum.org/EIPS/eip-721.

ESTEVAM, G. et al. Accurate and decentralized timestamping using smart contracts on the ethereum blockchain. **Information Processing & Management**, v. 58, n. 3, p. 102471, 2021. ISSN 0306-4573. DOI:10.1016/j.ipm.2020.102471.

ETH Gas Station. **How does the calculator work?** 2021. Acesso em: 20 de maio de 2021. Disponível em: https://ethgasstation.info/FAQcalc.php.

ETHEREUM. **White Paper**. 2021. Acesso em: 19 de maio de 2021. Disponível em: https://ethereum.org/en/whitepaper/.

Ethereum Foundation. **Solidity**. 2021. Acesso em: 20 de maio de 2021. Disponível em: https://docs.soliditylang.org/en/develop/.

Ethereum Foundation. **web3.js - Ethereum JavaScript API**. 2021. Acesso em: 20 de maio de 2021. Disponível em: https://web3js.readthedocs.io/en/v1.3.4/.

ETHERSCAN. **Ethereum Average Block Size Chart**. 2021. Acesso em: 13 de junho de 2021. Disponível em: https://etherscan.io/chart/blocksize.

ETHERSCAN. **Ethereum Average Block Time Chart**. 2021. Acesso em: 19 de maio de 2021. Disponível em: https://etherscan.io/chart/blocktime.

ETHERSCAN. **Ethereum Daily Transactions Chart**. 2021. Acesso em: 20 de maio de 2021. Disponível em: https://etherscan.io/chart/tx.

ETHERSCAN. **Ethereum Full Node Sync (Default) Chart**. 2021. Acesso em: 21 de janeiro de 2020. Disponível em: https://etherscan.io/chartsync/chaindefault.

ETHERSCAN. **Ethereum Network Pending Transactions Chart - Time Series**. 2021. Acesso em: 19 de maio de 2021. Disponível em: https://etherscan.io/chart/pendingtx.

ETHERSCAN. **Ethereum Network Pending Transactions Chart - Time Series**. 2021. Acesso em: 13 de junho de 2021. Disponível em: https://etherscan.io/chart/pendingtx.

ETHERSCAN. **Ethereum Node Tracker**. 2021. Acesso em: 20 de maio de 2021. Disponível em: https://etherscan.io/nodetracker.

FAN, K. et al. Secure time synchronization scheme in IoT based on blockchain. In: **2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)**. Halifax, NS, Canada: IEEE, 2018. p. 1068–1068. DOI:10.1109/cybermatics_2018.2018.00196.

FAN, K. et al. A blockchain-based clock synchronization scheme in IoT. **Future Generation Computer Systems**, Elsevier BV, v. 101, p. 524–533, dez. 2019. DOI:10.1016/j.future.2019.06.007.

FORSYTH, D. **Probability and statistics for computer science**. Cham, Switzerland: Springer, 2018. ISBN 978-3-319-64409-7.

GIPP, B. et al. CryptSubmit: Introducing securely timestamped manuscript submission and peer review feedback using the blockchain. In: **2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL)**. Toronto, ON, Canada: IEEE, 2017. p. 1–4. DOI:10.1109/jcdl.2017.7991588.

GIPP, B.; MEUSCHKE, N.; GERNANDT, A. Decentralized trusted timestamping using the crypto currency bitcoin. **arXiv preprint arXiv:1502.04015**, 2015. Disponível em: https://arxiv.org/abs/1502.04015.

GMBH, B. **Ethereum Mainnet Statistics**. 2021. Acesso em: 26 de maio de 2020. Disponível em: https://www.ethernodes.org/countries.

GO-ETHEREUM. **go-ethereum/consensus/ethash/consensus.go**. 2021. Acesso em: 20 de maio de 2021. Disponível em: https://github.com/ethereum/go-ethereum/blob/c49a4165d074c2d08c362cfe1dac835b6a3e1251/consensus/ethash/consensus.go#L45.

GUANGKAI, M.; CHUNPENG, G.; LU, Z. Achieving reliable timestamp in the bitcoin platform. **Peer-to-Peer Networking and Applications**, 2020. DOI:10.1007/s12083-020-00905-6.

HABER, S.; STORNETTA, W. S. How to time-stamp a digital document. **Journal of Cryptology**, Springer Science and Business Media LLC, v. 3, n. 2, p. 99–111, jan. 1991. DOI:10.1007/bf00196791.

HALEVI, S.; MICALI, S. Practical and provably-secure commitment schemes from collision-free hashing. In: KOBLITZ, N. (Ed.). **Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, August 18-22, 1996, Proceedings**. Santa Barbara, California, USA: Springer, 1996. (Lecture Notes in Computer Science, v. 1109), p. 201–215.

HARTL, A.; ZSEBY, T.; FABINI, J. BeaconBlocks: Augmenting proof-of-stake with on-chain time synchronization. In: **2019 IEEE International Conference on Blockchain (Blockchain)**. Atlanta, GA, USA: IEEE, 2019. p. 353–360. DOI:10.1109/blockchain.2019.00055.

HEPP, T. et al. OriginStamp: A blockchain-backed system for decentralized trusted timestamping. **it - Information Technology**, Walter de Gruyter GmbH, v. 60, n. 5-6, p. 273–281, dez. 2018. DOI:10.1515/itit-2018-0020.

IGNACIO, B. **Beeple fatura US$ 69 milhões vendendo arte digital como NFT**. 2021. Acesso em: 19 de maio de 2021. Disponível em: https://tecnoblog.net/420763/beeple-fatura-us-69-milhoes-vendendo-arte-digital-como-nft/.

INFRAESTRUTURA DE CHAVES PUBLICAS BRASILEIRA. **CRITÉRIOS E PROCEDIMENTOS PARA CREDENCIAMENTO DAS ENTIDADES INTEGRANTES DA ICP-BRASIL**: Doc-icp-03 versão 6.2. 2020. Acesso em: 19 de maio de 2021. Disponível em: https://www.gov.br/iti/pt-br/centrais-de-conteudo/doc-icp-03-v-6-2-crit-e-proced-para-cred-das-ent-integ-da-icp-brasil-pdf.

INFRAESTRUTURA DE CHAVES PUBLICAS BRASILEIRA. **VISÃO GERAL DO SISTEMA DE CARIMBOS DO TEMPO NA ICP-BRASIL**: Doc-icp-11 versão 1.4. 2020. Acesso em: 19 de maio de 2021. Disponível em: https://www.gov.br/iti/pt-br/centrais-de-conteudo/doc-icp-11-v-1-3-visao-geral-do-sist-de-cts-na-icp-brasil-pdf.

JAQUET-CHIFFELLE, D.-O.; CASEY, E.; BOURQUENOUD, J. Tamperproof timestamped provenance ledger using blockchain technology. **Forensic Science International: Digital Investigation**, v. 33, p. 300977, 2020. DOI:10.1016/j.fsidi.2020.300977.

KIM, S. K. et al. Measuring ethereum network peers. In: **Proceedings of the Internet Measurement Conference 2018, October 31 - November 02, 2018**. Boston, MA, USA: IMC, 2018. p. 91–104.

KITCHENHAM, B. Procedures for performing systematic reviews. **Keele, UK, Keele Univ.**, v. 33, 08 2004.

KOLYDAS, T. Timestamping metadata using blockchain: A practical approach. In: GAROUFALLOU, E.; FALLUCCHI, F.; LUCA, E. W. D. (Ed.). **Metadata and Semantic Research**. Cham: Springer International Publishing, 2019. p. 451–458. DOI:10.1007/978-3-030-36599-8_42.

KUROSE, J. **Computer networking : a top-down approach**. Harlow, England Boston: Pearson, 2017. ISBN 978-0133594140.

LANDERRECHE, E.; SCHAFFNER, C.; STEVENS, M. **Cryptographic timestamping through sequential work**. 2018. Acesso em: 20 de maio de 2021. Disponível em: https://marc-stevens.nl/research/papers/preprint_LSS18_Cryptographic-timestamping.pdf.

LANDERRECHE, E.; STEVENS, M.; SCHAFFNER, C. Non-interactive cryptographic timestamping based on verifiable delay functions. In: BONNEAU, J.; HENINGER, N. (Ed.). **Financial Cryptography and Data Security**. Cham: Springer International Publishing, 2020. p. 541–558. DOI:10.1007/978-3-030-51280-4_29.

LESK, M. Ideas ahead of their time: Digital time stamping. **IEEE Security Privacy**, v. 13, n. 4, p. 76–79, 2015. DOI:10.1109/MSP.2015.69.

LI, J. et al. Blockchain-based public auditing for big data in cloud storage. **Information Processing & Management**, Elsevier BV, p. 102382, set. 2020. DOI:10.1016/j.ipm.2020.102382.

MANIATIS, P.; BAKER, M. Secure history preservation through timeline entanglement. In: BONEH, D. (Ed.). **Proceedings of the 11th USENIX Security Symposium, August 5-9, 2002**. San Francisco, CA, USA: USENIX, 2002. p. 297–312.

MENEZES, A. J.; OORSCHOT, P. C. van; VANSTONE, S. A. **Handbook of Applied Cryptography**. CRC Press, 2001. Disponível em: http://www.cacr.math.uwaterloo.ca/hac/.

MERKLE, R. C. A certified digital signature. In: BRASSARD, G. (Ed.). **Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, August 20-24, 1989, Proceedings**. Santa Barbara, California, USA: Springer, 1989. (Lecture Notes in Computer Science, v. 435), p. 218–238. DOI:10.1007/0-387-34805-0_21.

Mostéfaoui, A.; Raynal, M. Intrusion-tolerant broadcast and agreement abstractions in the presence of byzantine processes. **IEEE Transactions on Parallel and Distributed Systems**, v. 27, n. 4, p. 1085–1098, 2016. DOI:10.1109/TPDS.2015.2427797.

Mukhopadhyay, U. et al. A brief survey of cryptocurrency systems. In: **2016 14th Annual Conference on Privacy, Security and Trust (PST)**. Auckland, New Zealand: IEEE, 2016. p. 745–752. DOI:10.1109/PST.2016.7906988.

NAKAMOTO, S. **Bitcoin: A peer-to-peer electronic cash system**. 2008. Acesso em: 19 de maio de 2021. Disponível em: https://bitcoin.org/bitcoin.pdf.

National Institute of Standards and Technology. **The Official U.S. Time**. 2020. Online. Accessed 13-May-2020. Disponível em: https://timegov.boulder.nist.gov/.

NGUYEN, C. T. et al. Proof-of-stake consensus mechanisms for future blockchain networks: Fundamentals, applications and opportunities. **IEEE Access**, IEEE, online, v. 7, p. 85727–85745, 2019.

NSC, T. **UFSC é a primeira universidade brasileira a emitir diploma digital lançado pelo MEC**. 2020. Acesso em: 19 de maio de 2021. Disponível em: https://www.nsctotal.com.br/noticias/ufsc-e-a-primeira-universidade-brasileira-a-emitir-diploma-digital-lancado-pelo-mec.

NYSE. **NYSE Composite (DJ) NYA**. 2021. Acesso em: 29 de maio de 2021. Disponível em: https://www.nyse.com/quote/index/NYA.

OHAM, C. et al. B-FERL: Blockchain based framework for securing smart vehicles. **Information Processing & Management**, Elsevier BV, v. 58, n. 1, p. 102426, jan. 2021. DOI:10.1016/j.ipm.2020.102426.

OLIVEIRA, G. E. de. **Proposta de Carimbo do Tempo Descentralizado e Preciso para a ICP-Brasil utilizando Sistemas Embarcados e Criptografia Pós-Quântica**. 2020. Acesso em: 19 de maio de 2021. Disponível em: https://repositorio.ufsc.br/handle/123456789/212406.

OPEN-ETHEREUM. **openethereum/ethcore/verification/src/verification.rs**. 2021. Acesso em: 20 de maio de 2021. Disponível em: https://github.com/openethereum/openethereum/blob/5c3c9797985d86cfd8983e180575639cad735c11/ethcore/verification/src/verification.rs#L344.

OpenJS Foundation. **Node.js**. 2021. Acesso em: 20 de maio de 2021. Disponível em: https://nodejs.org/.

OpenSSL Software Foundation. **/docs/man1.1.1/man1/openssl-speed.html**. 2018. Acesso em: 13 de junho de 2021. Disponível em: https://www.openssl.org/docs/man1.1.1/man1/openssl-speed.html.

PALMA, L. M. et al. Blockchain and smart contracts for higher education registry in brazil. **Int. Journal of Network Management**, v. 29, n. 3, 2019. DOI:10.1002/nem.2061.

PARITY-ETHEREUM. **parity-ethereum/ethcore/verification/src/verification.rs**. 2021. Acesso em: 20 de maio de 2021. Disponível em: https://github.com/openethereum/parity-ethereum/blob/v2.7.2-stable/ethcore/verification/src/verification.rs#L339.

REGNER, F.; URBACH, N.; SCHWEIZER, A. Nfts in practice - non-fungible tokens as core component of a blockchain-based event ticketing application. In: **40th International Conference on Information Systems**. Munich: ICIS, 2019.

SANTIAGO, C. **Soluti Responde - ACT: o que é e como aplicar um carimbo do tempo?** 2019. Acesso em: 19 de maio de 2021. Disponível em: https://solutiresponde.com.br/act-o-que-e-e-como-aplicar-um-carimbo-do-tempo/.

SINGLA, A. et al. The internet at the speed of light. In: **Proceedings of the 13th ACM Workshop on Hot Topics in Networks**. New York, NY, USA: Association for Computing Machinery, 2014. (HotNets-XIII), p. 1–7. ISBN 9781450332569. Disponível em: https://doi.org/10.1145/2670518.2673876.

STALLINGS, W. **Cryptography and Network Security Principles and Practice**. Edinburgh Gate, Harlow, England: Person, 2017. ISBN 10:1-292-15858-1.

STANDARDS, N. I. of; TECHNOLOGY. **FIPS 180-2**. 2002. Acesso em: 19 de maio de 2021. Disponível em: https://csrc.nist.gov/csrc/media/publications/fips/180/2/archive/2002-08-01/documents/fips180-2.pdf.

STAVROU, A.; VOAS, J. Verified time. **Computer**, Institute of Electrical and Electronics Engineers (IEEE), v. 50, n. 3, p. 78–82, mar. 2017. DOI:10.1109/mc.2017.63.

Surety, LLC. **Ensuring Record Integrity with Absolute Proof^{SM}**. Online, 2003.

SZABO, N. Smart contracts: building blocks for digital markets. **EXTROPY: The Journal of Transhumanist Thought,(16)**, v. 18, p. 2, 1996.

SZALACHOWSKI, P. (short paper) towards more reliable bitcoin timestamps. In: **2018 Crypto Valley Conference on Blockchain Technology (CVCBT)**. Zug, Switzerland: IEEE, 2018. p. 101–104. DOI:10.1109/cvcbt.2018.00018.

VIGIL, M. A. G. et al. Integrity, authenticity, non-repudiation, and proof of existence for long-term archiving: A survey. **Comput. Secur.**, v. 50, p. 16–32, 2015. DOI:10.1016/j.cose.2014.12.004.

VIGIL, M. A. G. et al. An efficient time-stamping solution for long-term digital archiving. In: **IEEE 33rd International Performance Computing and Communications Conference, IPCCC 2014, December 5-7, 2014**. Austin, TX, USA: IEEE Computer Society, 2014. p. 1–8. DOI:10.1109/PCCC.2014.7017099.

WOOD, G. et al. Ethereum: A secure decentralised generalised transaction ledger. **Ethereum project yellow paper**, v. 151, n. 2014, p. 1–32, 2014. Acesso em: 19 de maio de 2021. Disponível em: https://gavwood.com/paper.pdf.

XU, X. et al. A taxonomy of blockchain-based systems for architecture design. In: **2017 IEEE International Conference on Software Architecture (ICSA)**. Gothenburg, Sweden: IEEE, 2017. p. 243–252. DOI:10.1109/icsa.2017.33.

ZAN, C.; XU, H.-C. A global clock model for the consortium blockchains. In: ZHENG, Z. et al. (Ed.). **Blockchain and Trustworthy Systems**. Singapore: Springer Singapore, 2020. p. 71–80. DOI:10.1007/978-981-15-2777-7_6.

ZHANG, Y. et al. Chronos+: An accurate blockchain-based time-stamping scheme for cloud storage. **IEEE Transactions on Services Computing**, Institute of Electrical and Electronics Engineers (IEEE), p. 1–1, 2019. DOI:10.1109/tsc.2019.2947476.

ZHANG, Y. et al. Chronos: Secure and accurate time-stamping scheme for digital files via blockchain. In: **ICC 2019 - 2019 IEEE International Conference on Communications (ICC)**. Shanghai, China: IEEE, 2019. p. 1–6. DOI:10.1109/icc.2019.8762071.

ZHAO, Q. et al. Blockchain-based privacy-preserving remote data integrity checking scheme for IoT information systems. **Information Processing & Management**, Elsevier BV, v. 57, n. 6, p. 102355, nov. 2020. DOI:10.1016/j.ipm.2020.102355.

ZUCCHERATO, R. et al. **Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP)**. RFC Editor, 2001. RFC 3161. (Request for Comments, 3161). Acesso em: 19 de maio de 2021. Disponível em: https://rfc-editor.org/rfc/rfc3161.txt.