UNIVERSIDADE FEDERAL FLUMINENSE

MARCELA TULER DE OLIVEIRA

Desenvolvimento de um Mecanismo de Consenso Baseado em Confiança para Cadeias de Blocos Privadas Permissionadas

NITERÓI

2018

UNIVERSIDADE FEDERAL FLUMINENSE

MARCELA TULER DE OLIVEIRA

Desenvolvimento de um Mecanismo de Consenso Baseado em Confiança para Cadeias de Blocos Privadas Permissionadas

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Engenharia Elétrica e de Telecomunicações da Universidade Federal Fluminense, como requisito parcial para obtenção do Grau de Mestre em Engenharia Elétrica e de Telecomunicação.

Orientador:
Prof. Ricardo Campanha Carrano, D.Sc.

Co-orientador:
Prof. Diogo Menezes Ferrazani Mattos, D.Sc.

NITERÓI

2018

Ficha catalográfica automática - SDC/BEE
Gerada com informações fornecidas pelo autor

D278d De Oliveira, Marcela Tuler
Desenvolvimento de um Mecanismo de Consenso Baseado em Confianca para Cadeias de Blocos Privadas Permissionadas : / Marcela Tuler De Oliveira ; Ricardo Campanha Carrano, orientador ; Diogo Menezes Ferrazani Mattos, coorientador. Niterói, 2018.
95 f. : il.

Dissertação (mestrado)-Universidade Federal Fluminense, Niterói, 2018.

DOI: http://dx.doi.org/10.22409/PPGEET.2018.m.13164261730

1. Cadeia de Blocos. 2. Mecanismos de Consenso. 3. Cadeias de Blocos Privadas Permissionadas. 4. Prontuário Médico Eletrônico. 5. Produção intelectual. I. Campanha Carrano, Ricardo, orientador. II. Menezes Ferrazani Mattos, Diogo , coorientador. III. Universidade Federal Fluminense. Escola de Engenharia. IV. Título.

CDD -

Bibliotecária responsável: Fabiana Menezes Santos da Silva - CRB7/5274

MARCELA TULER DE OLIVEIRA

Desenvolvimento de um Mecanismo de Consenso Baseado em Confiança para Cadeias de Blocos Privadas Permissionadas

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Engenharia Elétrica e de Telecomunicações da Universidade Federal Fluminense, como requisito parcial para obtenção do Grau de Mestre em Engenharia Elétrica e de Telecomunicação.

BANCA EXAMINADORA

Prof. Ricardo Campanha Carrano, D.Sc., UFF

Prof. Diogo M. Ferrazani Mattos, D.Sc., UFF

Prof. Natalia Castro Fernandes, D.Sc., UFF

Prof. Dianne Scherly Varela de Medeiros, D.Sc., UFF

Prof. Rodrigo de Souza Couto, D.Sc., UFRJ

Niterói
2018

À minha família.

Agradecimentos

Neste tempo de mestrado, de muito estudo, esforço e empenho, gostaria de agradecer primeiramente à Deus, por iluminar meu caminho e por todas as bênçãos e conquistas.

A algumas pessoas que me acompanharam e foram fundamentais para a realização de mais este sonho, expresso aqui a minha sincera gratidão e um pouquinho da importância que elas tiveram, e ainda têm, nesta conquista. Gostaria de agradecer à minha família e amigos pela paciência e apoio incondicional que me deram, especialmente ao meu filho Theo, que é a razão da minha alegria e me dá forças para seguir sempre em frente.

Não poderia deixar de agradecer ao meu orientador, Professor Ricardo C. Carrano, por toda a paciência e empenho com que sempre me orientou neste trabalho. Muito obrigada por ser um exemplo de profissional e uma das maiores influências para que eu siga na carreira acadêmica.

Desejo igualmente agradecer ao meu co-orientador, Professor Diogo M. F. Mattos, pelo conhecimento compartilhado e por todo o suporte que foi fundamental para o desenvolvimento desta dissertação e em todos os trabalhos que realizei durante o mestrado. Muito obrigada por ter me corrigido quando necessário, sem nunca me desmotivar.

Minha gratidão às professoras Dianne S. V. Medeiros e Natália C. Fernandes pelo apoio e aprendizado nos trabalhos realizados e a todos professores, colaboradores e alunos do Laboratório Mídiacom pelo conhecimento compartilhado e amizade construída.

Quero também agradecer aos amigos, Lúcio H. A. Reis e Gabriel R. Carrara pela parceria nas pesquisas e no aprendizado, cujos esforços e auxílio tornaram possível a concretização deste projeto.

Agradeço aos professores membros da banca examinadora Rodrigo S. Couto, Natália C. Fernandes e Dianne S. V. Medeiros, pelo interesse e disponibilidade.

Por fim, agradeço aos órgãos de fomento CNPq, CAPES, RNP e FAPERJ.

Resumo

Cadeia de blocos é a principal tendência para aplicações distribuídas sem uso de uma terceira entidade confiável e atribuindo requisitos de segurança como integridade, autenticidade, não repúdio e auditoria distribuída sobre os dados armazenados. Mesmo em rede de desconfiança mútua entre os pares, os mecanismos de consenso permitem que os nós concordem sobre a visão global da rede, dispensando intermediários na comunicação. A aplicação da tecnologia de cadeia de blocos para o armazenamento de prontuários médicos eletrônicos possibilitaria a descentralização da administração dos dados para um controle centrado no paciente. O trabalho avalia o desempenho de duas plataformas de cadeias de blocos privadas que utilizam Prova de Autoridade como mecanismo de consenso. As plataformas apresentaram fragilidade no consenso, pois concentram o controle da rede em um subgrupo de nós, o que ameaça os requisitos de segurança da cadeia. Para retomar os requisitos de segurança consequentes do consenso distribuído, esta dissertação propõe um modelo de confiança para o consenso, que oferece controle de acesso à rede, monitoramento de nós mineradores e exclusão de nós maliciosos. Para verificar se o mecanismo é escalável e resiliente às ações maliciosas, foram realizadas simulações em diversos cenários de rede e diferentes proporções de ações maliciosas. Por fim, foi proposta uma abordagem híbrida baseada em cadeias de blocos em conjunto a uma infraestrutura de chave pública, para proteger e armazenar registros médicos eletrônicos (Electronic Medical Records, EMR), descentralizando a administração dos dados, para um controle de acesso centrado no paciente com requisitos de privacidade respeitados. Na proposta, os EMRs são criptografados e o paciente compartilha a chave de decodificação somente com profissionais de saúde nos quais confia. A escalabilidade da abordagem é investigada e os resultados mostram que oferece disponibilização dos EMRs em tempo real e que a rede possui escalabilidade, pois, ao aumentar o número de nós, há aumento linear no tamanho da cadeia.

Palavras-chave: Cadeia de Blocos, Cadeias de Blocos Privadas Permissionadas, Mecanismo de Consenso Baseado em Confiança, Prontuário Médico Eletrônico.

Abstract

The blockchain technology is the main trend to provide distributed applications without the use of a trusted third party. The blockchain technology assures security requirements such as integrity, authenticity, non-repudiation, and distributed audit of stored data. Even in a network of mutual mistrust between peers, consensus mechanisms allow nodes to agree on the global view of the network, without the need for intermediaries in communication. The application of blockchain technology for the storage of electronic medical records would enable the decentralization of data administration to a patient-centered control. This work evaluates the performance of two platforms of private blockchain that use Proof of Authority as a consensus mechanism. The platforms presented fragility in the consensus, because they concentrate the control of the network in subgroups of nodes, which hampers the security requirements of the blockchain. In order to achieve security requirements resulting from the distributed consensus, this work proposes a consensus trust-based mechanism, which provides access control, monitoring of miners, and expulsion of malicious nodes. To verify if the mechanism is scalable and resilient to malicious actions, simulations were performed for various network scenarios and for different rates of malicious actions. Finally, a Hybrid blockchain and Public Key Infrastructure approach was proposed to protect and store electronic medical records (EMR), decentralizing data management to patient-centered access control with respected privacy requirements. In the proposal, the EMRs are encrypted and the patient shares the decryption key only with the healthcare professionals they trust. The scalability of the approach is investigated and the results show that it offers real-time availability of the EMRs and that the network scales, since, by increasing the number of nodes, there is a linear increase in the size of the blockchain.

Keywords: Blockchain, Private Permissioned Blockchain, Trust Based Consensus Mechanism, EMR blockchain.

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

CA	:	Autoridade de Certificação;
PKI	:	Infraestrutura de Chave Púlica;
$\text{Tx}_{\text{ingresso}}$	:	Transação de Solicitação de Ingresso;
$\text{Tx}_{\text{minerador}}$	:	Transação de Solicitação para ser Minerador;
$\text{Tx}_{\text{votação}}$	:	Transação de Votação por Expulsão de nó Malicioso;
$\text{Tx}_{\text{expulsão}}$	:	Transação de Expulsão de nó Malicioso;
J	:	Número de Juízes por nó;
J_E	:	Número de Juízes Esperado por nó;
N	:	Número de Nós na rede;
M	:	Número de Mineradores na rede;
M_E	:	Número de Mineradores Esperados na rede;
$N_{maliciosos}$	:	Número de Nós Maliciosos;
$M_{maliciosos}$	:	Número de Mineradores Maliciosos;
R	:	Nota de Reputação;
NI	:	Nota Inicial de Reputação;
L	:	Limiar de Confiança;
Up	:	Nota de Atualização de Reputação;
EMR	:	Prontuário Médico Eletrônico;
PoW	:	Prova de Trabalho;
PoA	:	Prova de Autoridade;
PoS	:	Prova de Participação;
PoC	:	Prova de Capacidade;
PBFT	:	Prática Byzantina de Tolerância a Falha;

Sumário

Capítulo 1

Introdução*

A tecnologia *blockchain*, ou cadeia de blocos, é a principal tendência para aplicações distribuídas sem uso de uma terceira entidade confiável e com requisitos de segurança como integridade, autenticidade, não repúdio e, principalmente, auditoria sobre os dados armazenados [44]. A ideia central da tecnologia de cadeia de blocos é distribuir a validação dos dados e a responsabilidade pela inserção de novos dados na cadeia sobre uma rede par a par que executa mecanismos de consenso e regras de validação dos dados. Assim, a tecnologia de cadeia de blocos é apontada como uma alternativa simples e segura para o desenvolvimento de aplicações em diversas áreas do conhecimento [32, 17].

A cadeia de blocos consiste em um histórico imutável de transações em uma estrutura de dados distribuída, em que cada nó da rede contém uma réplica de todos os blocos. Cada nó participante do sistema executa protocolos de consenso que validam as transações e as agrupam em blocos, que são encadeados usando uma referência ao bloco antecessor. A referência é um resumo criptográfico (*hash*), obtido através de algoritmos criptográficos unidirecionais. Essa propriedade torna improvável a recuperação dos dados originais a partir do resumo criptográfico gerado, garantindo a integridade do conteúdo e a segurança da cadeia. A cadeia de blocos é uma tecnologia capaz de atender muitas das necessidades dos sistemas de armazenamento de dados sensíveis, como prontuários médicos eletrônicos, oferecendo padronização e disponibilidade às informações do paciente, mais transparência e agilidade às consultas médicas e atribuindo responsabilidade e não repúdio à atividade médica, em um repositório distribuído e auditável. Contudo, aplicações de dados sensíveis

*Este capítulo é baseado no minicurso "*Blockchain* para Segurança em Redes Elétricas Inteligentes: Aplicações, Tendências e Desafios" [41]. Agradecimentos pela colaboração dos autores Diogo M.F. Mattos, Dianne S.V. Medeiros, Natalia C. Fernandes, Gabriel R. Carrara, Arthur A.Z. Soares, Luiz Claudio S. Magalhães, Diego Passos, Ricardo C. Carrano, Igor M. Moraes, Célio V. N. Albuquerque e Débora C. Muchaluat-Saade.

enfrentam o desafio de manter a privacidade do paciente [18, 29, 3, 59].

No desenvolvimento da tecnologia de cadeia de blocos, é possível diferenciar três momentos principais e definir duas gerações, conforme mostrado na Figura 1.1. A *blockchain 1.0*, que marca o primeiro momento, é uma tecnologia de armazenamento de transações na cadeia, de forma distribuída e através da criação de ativos na forma de *tokens*. A primeira aplicação a ganhar notoriedade no uso de cadeia de blocos foi a criptomoeda *Bitcoin* [44], introduzida em 2009. A ideia central é criar um livro razão seguro e confiável, distribuído em uma rede par a par. Um marco na evolução da tecnologia foi o uso da cadeia de blocos para gerir, de forma segura, a transferência de ativos sob a forma de criptomoeda, a partir de 2013. Em 2015, essa tecnologia passou a ser usada pelo sistema LINQ, da Nasdaq, para armazenar transações privadas[1]. A segunda geração dessa tecnologia, referenciada como *blockchain 2.0* [27], consiste na introdução dos contratos inteligentes. A plataforma *Ethereum* [55] foi a primeira a suportar os contratos inteligentes, ao permitir o armazenamento do código de execução automática na cadeia. A principal evolução entre a primeira e a segunda gerações foi que, a exemplo da *Bitcoin*, a linguagem de programação das cadeias de blocos da primeira geração era "orientada a pilha" e não permitia laços (*loops*) no código executável, a fim de evitar ciclos mortos (*deadlocks*) no sistema. Já em sistemas de segunda geração, como a *Ethereum*, são permitidas execuções de laços nos códigos, que consomem créditos - no caso da *Ethereum* quantificados em *Ether* - ao executar contratos inteligentes. Assim, no caso de um ciclo morto, a execução é interrompida quando se esgotam os créditos da conta usada [42].

Em aplicações distribuídas, os mecanismos de consenso são fundamentais para que os nós participantes concordem sobre o funcionamento da aplicação e a ordem e veracidade das informações que são armazenadas, dispensando a terceira entidade centralizadora. Na tecnologia de cadeia de blocos os mecanismos de consenso têm a função de determinar a ordem dos blocos e os nós que foram responsáveis pela geração, de forma que todos os nós da rede par a par concordem sobre o conteúdo da cadeia e tenham a réplica atualizada armazenada localmente. As redes públicas de cadeia de blocos costumam adotar mecanismos de consenso abertos, baseados em mineração, onde os mineradores competem entre si pela liderança do consenso, a partir de um alto gasto de poder computacional, i.e. consenso baseado em Prova de Trabalho (*Proof of Work, PoW*), do poder de posse sobre a criptomoeda, i.e. consenso baseado em Prova de Posse (*Proof of Stake, PoS*), ou de outros poderes de relevância para a eleição do minerador que não podem ser mo-

[1] Acessível em http://ir.nasdaq.com/news-releases/news-release-details/nasdaq-linq-enables-first-ever-private-securities-issuance.

nopolizados, i.e consenso baseado em Prova de Autoridade (*Proof of Autority, PoA*). Os vencedores da competição recebem um incentivo, geralmente em criptomoeda. O pagamento do incentivo é fundamental para a estratégia montada para tolerar ataques bizantinos, como ataques em conluio para subverter a rede. Por isso, a governança da rede costuma estabelecer um conjunto de regras mínimas para a sua existência, além de critérios de favorecimento mútuo dos usuários [27]. Atualmente, a Prova de Trabalho (PoW) é uma das poucas abordagens de consenso para redes públicas bem sucedidas e resilientes a ataques *Sybil* [20], em que o atacante subverte a reputação de uma rede par a par criando um grande número de identidades. Contudo, na rede *Bitcoin*, a criação de conglomerados de mineração tem colocado em risco o funcionamento do mecanismo na rede. Além disso, o alto consumo de energia e processamento para o funcionamento do mecanismo gera muitas críticas, por ser insustentável.

Cadeias de blocos privadas permitem a aplicação de mecanismos de consenso mais sustentáveis, baseados em votação, graças aos mecanismos de controle de acesso. No entanto, consensos baseados em práticas bizantinas de tolerância a falhas enfrentam problemas de escalabilidade, pois geram altas cargas de mensagens para a votação [17, 10]. Outra opção para redes privadas é o mecanismo de Prova de Autoridade (PoA), em que são determinados nós com permissão para gerar blocos da cadeia, com a função de minerador distribuída entre os nós de autoridade. No entanto, não existe um mecanismo de confiança associado ao PoA, para monitorar os nós de autoridade. Assim, o conceito básico de descentralização da tecnologia é violado, concentrando o poder da rede não em um nó, mas em um grupo de nós que pode agir maliciosamente.

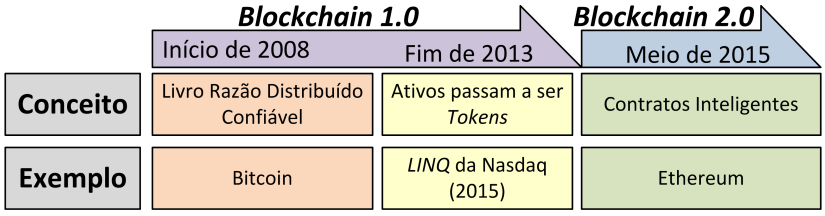


Figura 1.1: Gerações no desenvolvimento da tecnologia de cadeia de blocos. O conceito surge com o livro-razão confiável, distribuído e implantado pela *Bitcoin*, evolui para a transação de ativos na forma de *tokens* e se concretiza como uma segunda geração, após a introdução de contratos inteligentes na estrutura de dados da cadeia [41].

As redes públicas de cadeia de blocos costumam adotar mecanismos de consenso abertos, baseados em mineração, onde os mineradores competem entre si pela liderança do consenso, a partir de um alto gasto de poder computacional, do poder de posse sobre a criptomoeda ou de outros poderes de relevância para a eleição do minerador - e que

não podem ser monopolizados. Os vencedores da competição recebem um incentivo, geralmente em criptomoeda. O pagamento do incentivo é fundamental para a estratégia montada para tolerar ataques bizantinos, como ataques em conluio para subverter a rede. Por isso, a governança da rede costuma estabelecer um conjunto de regras mínimas para a sua existência, além de critérios de favorecimento mútuo dos usuários [27]. Atualmente, a Prova de Trabalho (*Proof of Work*, PoW) é uma das poucas abordagens de consenso para redes públicas bem sucedidas e resilientes a ataques *Sybil* [20]. Contudo, na rede *Bitcoin*, a criação de conglomerados de mineração tem colocado em risco o funcionamento do mecanismo na rede. Além disso, o alto consumo de energia e processamento para o funcionamento do mecanismo gera muitas críticas, por ser insustentável.

Cadeias de blocos privadas permitem a aplicação de mecanismos de consenso mais sustentáveis, baseados em votação, graças aos mecanismos de controle de acesso. No entanto, consensos baseados em práticas bizantinas de tolerância a falhas enfrentam problemas de escalabilidade, pois geram altas cargas de mensagens para a votação [17, 10]. Outra opção para redes privadas é o mecanismo de Prova de Autoridade (*Proof of Authority*, PoA), em que são determinados nós com permissão para gerar blocos da cadeia, com a função de minerador distribuída entre os nós de autoridade. No entanto, não existe um mecanismo de confiança associado ao PoA, para monitorar os nós de autoridade. Assim, o conceito básico de descentralização da tecnologia é violado, concentrando o poder da rede não em um nó, mas em um grupo de nós que pode agir maliciosamente.

1.1 Motivação

Alcançar o consenso é um problema fundamental em computação distribuída confiável, pois permite que participantes coordenem as suas ações de forma a alcançar decisões comuns e, assim, garantir a manutenção da consistência dos seus estados e o progresso do sistema, apesar da existência de falhas [28]. Nesse sentido, o consenso é fundamental para a tecnologia de cadeia de blocos, pois possibilita a obtenção de um acordo sobre quais informações serão agregadas à cadeia na rede par a par.

Existem diversos mecanismos de consenso que podem ser empregados em cadeias de blocos, para a decisão sobre a ordem dos blocos e sobre qual nó será responsável por gerar o bloco. No entanto, o consenso pode ser visto como um mecanismo amplo, que permite que a rede possa ser auto configurável, de acordo com a cooperação dos nós participantes. É de interesse comum dos participantes que a rede funcione de maneira segura, visto que

não existe uma instituição administrando e controlando o funcionamento da rede. Por isso, a participação de nós monitorando o comportamento dos demais nós da rede e os processos de segurança da tecnologia de cadeia de blocos permite que nós maliciosos sejam identificados e expulsos. Sendo assim, a partir de um modelo de confiança, o mecanismo de consenso pode trazer robustez à rede e resiliência a ataques.

1.2 Objetivos

Esta dissertação tem como principal objetivo propor um mecanismo de consenso, baseado em confiança, para redes de cadeia de blocos privadas permissionadas. A proposta mostra como a utilização da cooperatividade dos nós da rede é fundamental para o desenvolvimento de um sistema de controle, no qual o comportamento dos nós é monitorado por outros nós. O monitoramento da nota de reputação é constante na rede e, por meio de votação, o nó considerado malicioso à rede é expulso.

Para o funcionamento do mecanismo, é preciso definir critérios de rigidez a serem aplicados a ações identificadas como maliciosas, qual o limiar de confiança adotado e qual nota recebe um nó que ainda não foi avaliado pela rede. Além disso, determinar quantos nós são necessários para monitorar o comportamento de um nó malicioso, a ponto de identificá-lo e expulsá-lo com maior eficiência. Também são objetivos deste trabalho, apresentar a avaliação de desempenho de duas plataformas de desenvolvimento de cadeias de blocos privadas permissionadas em relação ao tempo de validação de uma transação, ao tempo de resposta às buscas na cadeia e ao tempo de mineração de transações, e propor um caso de uso da tecnologia de cadeia de blocos para aplicação em armazenamento de prontuários médicos eletrônicos em uma abordagem híbrida, que associa a aplicação da infraestruturada de chaves públicas à tecnologia de cadeia de blocos, para obedecer aos requisitos de armazenamento de dados médicos e oferecer privacidade e controle de acesso aos dados centrado no paciente.

1.3 Metodologia

Considerando a disponibilidade das informações na cadeia, o papel de cada nó na rede par a par e como ocorre o consenso na rede, a tecnologia de cadeia de blocos apresenta características distintas, que são importantes para definir o cenário de aplicação. Trabalhos anteriores classificam as cadeias como pública, privada, permissionada e híbrida [48, 30].

Christidis e Devetsikiotis classificam as cadeias de bloco quanto aos aspectos de controle de acesso ao conteúdo da cadeia e quanto às funções que os nós da rede exercem [13]. Por isso, neste trabalho é proposto adotar a taxonomia de rede de cadeias de blocos em diferentes tipos de visão de rede: pública não permissionada, pública permissionada, privada não permissionada e privada permissionada.

A partir desta classificação da taxonomia, é avaliado o desempenho de duas plataformas de desenvolvimento de cadeias de blocos privadas permissionadas, *Parity*[2] e *Multichain*[3]. A avaliação consiste na comparação entre as plataformas, analisando a vazão das transações, a aceitação de blocos e a latência de acesso à cadeia, com a aplicação de cargas de trabalho realísticas. As cargas de trabalho são geradas seguindo a distribuição de probabilidades da chegada de transações na cadeia de blocos do *Bitcoin*. Os resultados mostram que cada plataforma se destaca em critérios específicos. As decisões de projeto de cada plataforma resultam em restrições de funcionalidades, que devem ser tratadas por desenvolvedores para a criação de cadeias mais seguras e eficientes. Além disso, os resultados obtidos com a implementação das plataformas são utilizados como métricas para o desenvolvimento de um simulador de cadeia de blocos privada permissionada.

Apesar de o mecanismo de consenso PoA utilizado pelas plataformas testadas ser eficiente em relação ao funcionamento da tecnologia, as plataformas testadas não mantém um mecanismo de confiança entre os nós. Além disso, confiar que um grupo de nós de autoridade age corretamente sem ser em benefício próprio, contradiz os preceitos de segurança de uma cadeia de blocos, pois atribui a um grupo de nós a autoridade de intermediar as transações da rede. Por isso, é fundamental um mecanismo de confiança associado à Prova de Autoridade, para garantir que os nós mineradores são realmente confiáveis, a partir de um monitoramento constante de suas ações.

Por fim, é apresentado um caso de uso de aplicação de cadeias de blocos privadas permissionadas para registros médicos eletrônicos (EMR). Estas são informações altamente confidenciais, compartilhadas entre os pares envolvidos para manter o histórico do paciente atualizado. Proporcionar segurança, privacidade e disponibilidade a esses dados confidenciais é um desafio, porque normalmente após a publicação dos dados o paciente perde o controle sobre eles. Então, propõe-se utilizar a tecnologia de cadeia de blocos para o desenvolvimento de aplicações EMR seguras, onde o controle de acesso é centrado no paciente. Nesta proposta, o EMR é criptografado na cadeia de blocos, e o paciente compartilha a chave de decodificação somente com profissionais de saúde nos quais con-

[2]Disponível em https://parity.io/
[3]Disponível em https://multichain.com/

fia. A escalabilidade da abordagem é investigada e os resultados mostram que a rede se adapta bem, já que o aumento do número de nós na rede implica em um aumento linear no tamanho da cadeia armazenada. Os resultados revelam que o tempo para inserir um novo EMR na cadeia de blocos permanece baixo, mesmo quando o número de nós na rede aumenta.

1.4 Contribuições

A principal contribuição desta dissertação é a proposta de um mecanismo de consenso baseado em confiança para redes de cadeias de blocos privadas permissionadas, que oferece a auto organização da rede de forma escalável, o monitoramento dos nós mineradores e expulsão de nós maliciosos. Além disso, analisou-se o desempenho de duas plataformas de redes privadas e permissionadas, em que os resultados estão dispostos nesta dissertação e foram utilizados para o desenvolvimento de um simulador de cadeia de blocos privada permissionada. Por fim, é proposto um caso de uso de abordagem híbrida de cadeia de blocos privadas permissionadas e infraestrutura de chaves pública para registros médicos eletrônicos.

1.5 Organização da Dissertação

Esta dissertação está organizada da seguinte forma: o Capítulo 2 explica a tecnologia de cadeia de blocos e a taxonomia proposta. O Capítulo 3 mostra a comparação de desempenho entre plataformas privadas permissionadas. O Capítulo 4 expõe a proposta de um mecanismo de consenso baseado em confiança para cadeias de blocos privadas permissionadas. O Capítulo 5 apresenta os resultados das simulações do mecanismo de consenso proposto. O Capítulo 6 apresenta uma proposta de aplicação da tecnologia de cadeia de blocos para armazenamento de prontuários médicos. Finalmente, o Capítulo 7 trás as conclusões da dissertação.

Capítulo 2

Tecnologia de Cadeia de Blocos*

A tecnologia de cadeia de blocos consiste em uma rede par a par com uma estrutura de dados capaz de armazenar transações de forma ordenada e distribuída. Dessa forma, esta tecnologia é definida por dois elementos básicos: a estrutura de dados de encadeamento dos blocos e a rede par a par composta pelos nós participantes. O diferencial que a tecnologia de cadeia de blocos oferece em relação aos sistemas de banco de dados distribuídos é a não necessidade da terceira entidade centralizadora, âncora de confiança, para garantir a segurança entre transações na rede [44]. Ao ser introduzida uma terceira entidade centralizadora, é gerado um ponto único de falha, que prejudica a segurança e a privacidade das transações realizadas, quando são considerados os conflitos de interesses entre as partes envolvidas. Assim, a tecnologia tem sido amplamente empregada em diversas aplicações, a fim de garantir disponibilidade, integridade, privacidade e não repúdio, sem a necessidade de uma organização centralizadora controlando os dados.

A primeira geração de cadeia de blocos, representada pela *Bitcoin* [44], foi idealizada para transferências monetárias entre nós em uma rede pública, que representam transações de pequenas quantidades de dados em uma rede hostil, em que os nós não confiam uns nos outros. Posteriormente, a segunda geração de cadeia de blocos, representada pela rede *Ethereum*, propôs que sua estrutura de dados fosse usada para representar transações mais complexas, que executam uma determinada aplicação, os chamados contratos inteligentes, que são estruturas de computação de mensagens de objeto confiável e autoexecutáveis. Estes contratos são determinísticos, ou seja, uma mesma entrada sempre produzirá uma

*Este capítulo é baseado no minicurso "*Blockchain* para Segurança em Redes Elétricas Inteligentes: Aplicações, Tendências e Desafios" [41]. Agradecimentos pela colaboração dos autores Diogo M.F. Mattos, Dianne S.V. Medeiros, Natalia C. Fernandes, Gabriel R. Carrara, Arthur A.Z. Soares, Luiz Claudio S. Magalhães, Diego Passos, Ricardo C. Carrano, Igor M. Moraes, Célio V. N. Albuquerque e Débora C. Muchaluat-Saade.

mesma saída. Caso contrário, um contrato não determinístico geraria resultados aleatórios para os diferentes nós da rede [13], impossibilitando o alcance de um consenso sobre os dados que devem ser armazenados na cadeia de blocos. Como o contrato inteligente reside na cadeia de blocos, seu código pode ser inspecionado por todos os participantes da rede. Como todas as interações com um contrato ocorrem via mensagens assinadas, é possível rastrear todos os participantes envolvidos na operação do contrato. Sendo assim, a aplicação de contratos inteligentes possibilita a automatização de regras executáveis com o consentimento das várias partes envolvidas [55].

A segurança oferecida pela tecnologia de cadeia de blocos consiste em todos os nós participantes da rede par a par acessarem uma réplica idêntica da cadeia de blocos armazenada localmente, mesmo em um ambiente de desconfiança mútua entre participantes. Portanto, é necessário adotar mecanismos de validação e de consenso, para realizar a distribuição da réplica coerente dos dados, adotar mecanismos de assinatura digital e resumos criptográficos, para garantir a auditoria distribuída sobre as transações executadas na rede.

Para que as transações sejam adicionadas como parte da cadeia, são processadas por quatro camadas: transações, validação, geração de blocos e distribuição, mostradas na Figura 2.1.

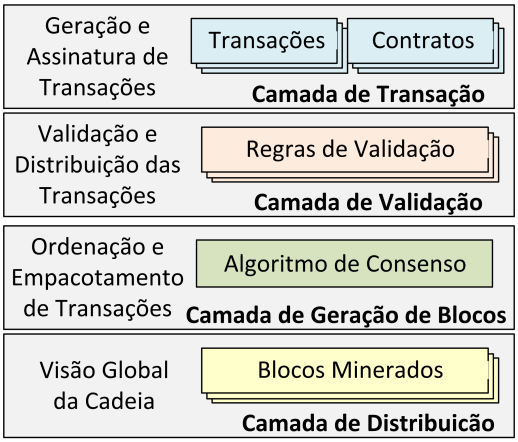


Figura 2.1: A tecnologia de cadeia de blocos em camadas. As transações dos usuários são geradas na camada de transação, validadas pela rede na camada de validação, inseridas em blocos na camada de geração de blocos e os blocos são distribuídos na camada de distribuição.

A camada de transações representa a geração das informações que se deseja armazenar em uma cadeia de blocos. Estas informações são, por natureza, não editáveis, isto é,

contratos, transferências bancárias, compra e venda etc. Assim como em bancos de dados tradicionais, na cadeia de blocos as transações seguem a semântica ACID (Atomicidade, Consistência, Isolamento e Durabilidade) [17]. A atomicidade requer que as transações sejam atômicas, ou seja, todas as operações devem ser executadas em conjunto, sem resultados parciais em caso de falha. A integridade define que a execução de uma transação deve levar a cadeia de blocos de um estado consistente a um outro estado consistente, ou seja, uma transação deve respeitar as regras de integridade dos dados. o Isolamento é garantido, pois, pelo armazenamento local da cadeia, a leitura e escrita de novas transações são ações isoladas, que não interferem umas nas outras. Por fim, a durabilidade assegura que as transações devem estar disponíveis em definitivo, mesmo em que haja perda do armazenamento localmente, os dados são disponibilizados por outros nós participantes. A interação direta dos usuários com a cadeia ocorre na camada de transações. O controle de acesso à rede acontece a partir da geração de um par de chaves assimétricas, para que o usuário possa assinar digitalmente uma transação na rede. O usuário que deseja se tornar um nó da rede par a par precisa gerar um par de chaves assimétricas, usando o mesmo algoritmo utilizado pelos outros nós da rede. O mecanismo de controle de acesso empregado na rede irá avaliar e conceder o acesso à chave pública do par gerado pelo usuário. Em redes públicas, não existe controle de acesso, o nó utiliza um par de chaves, geralmente disponível publicamente na Internet, para assinar as transações e ter acesso à cadeia. Os usuários são identificados somente pelas chaves públicas geradas ao ingressarem na rede, permitindo uma pseudoanonimização dos participantes [44]. Vale ressaltar que, por padrão, não há um esquema para identificação de usuários, já que são identificados apenas por suas chaves públicas. Não há um mecanismo que relacione uma chave pública com uma entidade conhecida, como é feito por uma Infraestrutura de Chaves Públicas (*Public Key Infrastructure* - PKI). Assim, o usuário assina suas transações com a chave privada e pode ser endereçado na rede por meio da chave pública. Seguindo critérios definidos na rede - como organização e linguagem de codificação pré-estabelecida para a elaboração da transação e assinatura - o usuário transmite a transação para todos os nós vizinhos. A Figura 2.2 mostra um exemplo de transação em que Alice deseja enviar um ativo monetário para Bob. Para isso, Alice utiliza sua chave pública como endereço do emitente, assina a transação com sua chave privada e endereça a transação à chave pública de Bob.

A camada de validação é determinada pela verificação das transações. Os nós vizinhos são responsáveis por verificar se as transações seguem os critérios pré-determinados pela rede. A validação analisa se a transação obedece a todas as regras da rede determinadas

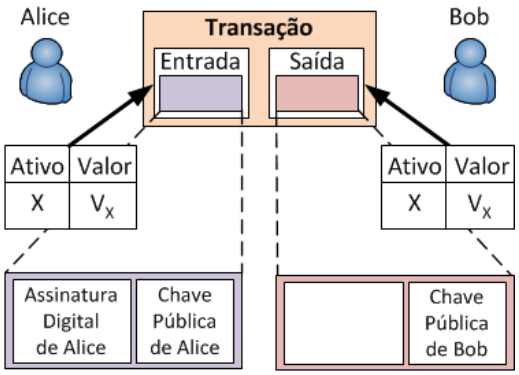


Figura 2.2: Exemplo de transação de transferência de ativos. Transação típica em que Alice transfere um ativo da sua posse para Bob. A transação é identificada pelas chaves públicas e validada pela assinatura digital de Alice.

no desenvolvimento da cadeia de blocos. Por exemplo, na *Bitcoin* a regra fundamental para executar uma transação é a disponibilidade da quantia enviada em posse da chave pública que a emite. As transações são transmitidas para os nós seguintes somente se são consideradas válidas. Caso a transação descumpra algum dos critérios da rede, deve ser descartada e não passada adiante. Esta é uma das camadas da tecnologia de cadeia de blocos responsável pela característica de auditoria distribuída.

Na camada de geração de blocos, as transações validadas na camada de validação estão disponíveis para formação do novo bloco da cadeia no repositório de transações válidas. Essas transações são coletadas, temporalmente ordenadas e empacotadas em um bloco candidato a ser inserido na cadeia, com a marcação de tempo correspondente (*timestamp*). A geração do bloco é chamada de processo de mineração, no qual o nó minerador assume a responsabilidade de gerar o bloco e introduzi-lo efetivamente na cadeia. Contudo, a escolha do nó minerador depende diretamente do mecanismo de consenso empregado na rede. Cada bloco minerado contém uma referência ao bloco antecessor, formando assim o encadeamento de blocos. Essa referência é feita através de resumos criptográficos (*hash*) [44, 12, 13], como ressaltado na Figura 2.3. O bloco inicial da cadeia é o bloco *genesis*, que armazena um valor determinado na formação da cadeia de blocos, que o identifica como o primeiro bloco da cadeia. A partir do *hash* do bloco *genesis* é gerado o encadeamento de *hash* entre blocos. O bloco B_n, com transações válidas, possui junto ao seu conteúdo o resumo criptográfico do bloco anterior B_{n-1}. O conteúdo completo do bloco B_n será usado para gerar o resumo criptográfico que será incluído como referência no próximo bloco B_{n+1}. Como o algoritmo que computa o resumo criptográfico é unidirecional, é improvável a recuperação dos dados originais a partir do

resumo gerado, assim como é improvável a geração de um novo conteúdo que gere o mesmo resumo, dado que a probabilidade de colisão do algoritmo empregado é muito baixa. Isso garante a integridade dos dados na cadeia. Caso haja uma mudança indevida no conteúdo de um dos blocos armazenados em um nó, tal mudança é evidenciada pela alteração do valor do *hash* desse bloco que, por sua vez, é propagada para todos os demais blocos da cadeia, devido ao encadeamento de valores dos *hashes* do blocos seguintes, a exemplo do $Bloco_{n-2}$ no *Nó 2* da Figura 2.3.

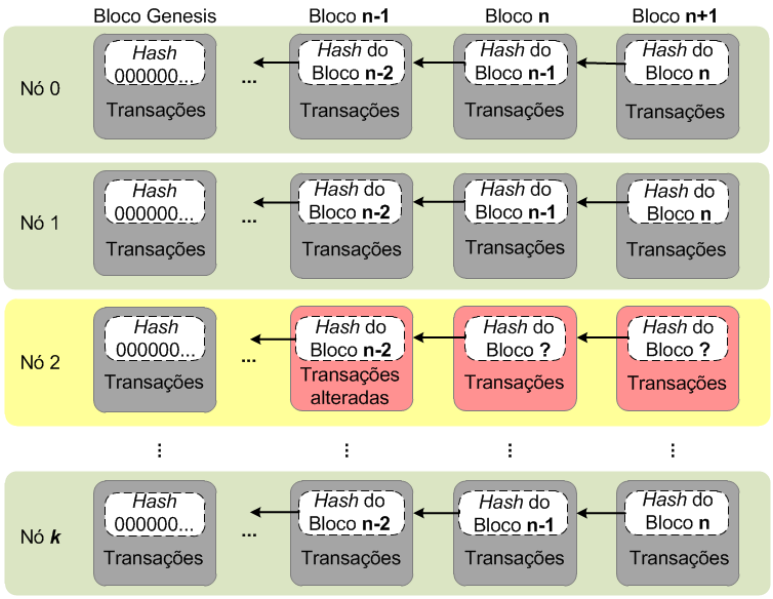


Figura 2.3: Visão esquemática da estrutura de dados em uma cadeia de blocos. O bloco *genesis* representa o primeiro bloco da cadeia. Cada bloco tem o resumo criptográfico do bloco anterior, gerando um encadeamento de resumos criptográficos. A alteração de um bloco gera a inconsistência de todos os blocos seguintes da cadeia [41].

Na camada de distribuição, o bloco minerado é adicionado à estrutura de dados da cadeia de blocos em cada nó da rede par a par. Destaca-se a necessidade de uma réplica atualizada da cadeia ser armazenada localmente, para que seja alcançado o consenso na rede. Assim sendo, as transações associadas aos blocos são executadas para atualizar a visão global da cadeia. Por exemplo, na execução de uma transação de transferência de criptomoedas, o valor da carteira dos envolvidos é atualizado para todos os nós da rede. Ressalta-se que a execução das transações determina uma mudança de estado global na cadeia, seja a transferência de ativos, seja a execução de um contrato inteligente. Contudo, o encadeamento de um novo bloco só ocorre nos nós vizinhos se o resumo criptográfico do bloco minerado estiver correto. Essa verificação é feita pela comparação entre o con-

teúdo do bloco e o resumo criptográfico apresentado. Caso contrário, o bloco minerado é descartado. Se todos os nós da rede possuírem o mesmo estado global da cadeia, com o mesmo conteúdo e blocos organizados na mesma ordem, os nós estão em consenso. Ao atingirem o consenso, todos os nós passam a ter acesso à mesma informação. A visão global distribuída da cadeia permite a disponibilidade e a auditoria das informações.

2.1 Taxonomia de Plataformas de Cadeia de Blocos

Christidis e Devetsikiotis classificam as cadeias de blocos segundo os aspectos de controle de acesso ao conteúdo da cadeia e quanto às permissões sobre as funções que exercem os nós da rede [13]. Outros trabalhos classificam as redes como pública, privada, permissionada e híbrida [48, 30]. Os conceitos de rede pública e privada são antagônicos e bem conhecidos, já os conceitos de rede permissionada e híbrida não são autoexplicativos. Por isso, diferente das classificações encontradas na literatura [48, 30], a taxonomia empregada neste trabalho propõe diferenciar as características de acesso à rede par a par, pública e privada, e as características de permissão ao participar da geração de blocos e consenso, permissionada e não permissionada. Assim, a taxonomia é dividida em quatro diferentes combinações de visão da rede, *pública não permissionada*, *pública permissionada*, *privada não permissionada* e *privada permissionada*, como evidenciado na Figura 2.4.

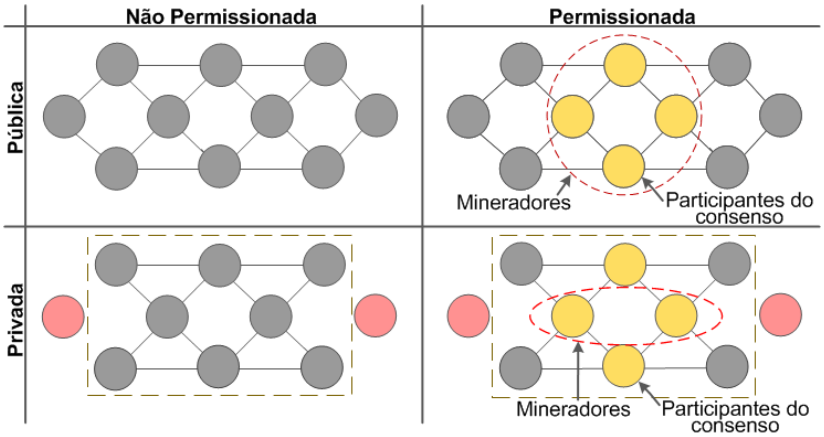


Figura 2.4: Taxonomia aplicada a redes de cadeia de blocos. A classificação entre pública e privada relaciona-se com a participação de nós na rede. A classificação entre permissionada e não permissionada relaciona-se com o papel desempenhado pelos nós da rede nos mecanismos de consenso e de geração de novos blocos [41].

Redes públicas e privadas se distinguem em relação ao controle de acesso à rede e ao conteúdo da cadeia, visto que, uma vez que o nó é participante da rede par a par, o nó

acessa a sua réplica da cadeia armazenada localmente. Em uma rede pública, de conteúdo aberto, não há qualquer mecanismo de controle de acesso e os nós podem ingressar e deixar a rede sem qualquer prejuízo para o mecanismo de consenso ou para a geração de novos blocos. Em redes privadas, em que o conteúdo é fechado, há medidas de controle de acesso e apenas nós autorizados podem acessar a rede par a par e ter acesso ao conteúdo da cadeia. Em paralelo, redes permissionadas e não permissionadas se diferenciam pelo critério de atividades desempenhadas na rede. Em redes não permissionadas, todos os nós desempenham o mesmo papel, podendo gerar transações, competir na mineração de blocos e participar do mecanismo de consenso. Em contraste, nas redes permissionadas os nós podem possuir papéis distintos, de acordo com a necessidade da aplicação. Por exemplo, em uma aplicação em que todos os nós podem criar transações, um grupo de nós da rede é responsável por realizar o consenso e apenas um subgrupo é autorizado a minerar novos blocos.

Em redes públicas não permissionadas de cadeias de blocos há desconfiança mútua entre os usuários e, por isso, os mecanismos de consenso são rígidos. A fim de evitar ataques de personificação (*Sybil Attack*) [20], o consenso em redes públicas não permissionadas é oneroso, exigindo-se a resolução de um desafio computacional como prova da participação no consenso, evitando, assim, que um nó apresente diversas identidades. Para que os nós mineradores participem desse mecanismo de consenso, é oferecido um incentivo econômico, na forma de criptomoeda. Isso é justificável pelas características principais de uma rede pública não permissionada, tais como conteúdo aberto e igualdade entre os nós. Todo nó pode ingressar e sair da rede. O participante da rede gera um par de chaves criptográficas para assinar e realizar transações quando ingressa pela primeira vez na rede. Além disso, qualquer nó pode ser um minerador e fazer parte do mecanismo de consenso da rede. Os problemas associados às redes públicas não permissionadas estão relacionados à escalabilidade e ao tempo efetivo desde a emissão da transação até a execução na cadeia. Essas redes constituem ambientes colaborativos e, portanto, dependem do comportamento benigno dos nós. Além disso, a competição entre os nós mineradores pelas recompensas da geração do bloco torna o processo mais lento e custoso. *Bitcoin* e *Ethereum* são exemplos de plataformas que oferecem configuração de rede pública não permissionada.

As redes públicas permissionadas foram desenvolvidas para aplicação de mecanismos de consenso menos custosos. A diferença entre as redes públicas não permissionadas e as permissionadas é a desigualdade de atuação dos nós na rede. Em uma rede pública permissionada, todos os dados são disponibilizados para auditoria pública e não se res-

tringe a entrada de novos nós. Contudo, um nó só participa da rede após a verificação adequada de sua identidade. Somente após essa verificação são alocadas as permissões que determinam quais atividades o nó pode executar na rede. Este tipo de rede é utilizada para gerenciar transações entre empresas ou em processos que envolvem várias entidades, permitindo que somente alguns nós de cada entidade fiquem responsáveis pela geração de blocos, aplicando mecanismos de consenso mais eficientes e sustentáveis. A Querum da A*Star Labs é um exemplo de plataforma que oferece a configuração de rede pública permissionada.

As redes privadas não permissionadas se diferenciam das redes públicas por restringirem a entrada de nós, portanto, só fornecem a réplica da cadeia a nós identificados por uma chave pública autorizada. Existe uma instituição, ou um conjunto delas, que determina quem são os nós autorizados a participar da rede. Necessário é destacar que o conceito de rede privada restringe-se ao controle de acesso aos dados da cadeia. Os nós que participam da rede podem ter funções iguais e exercem a mesma importância na rede, pois não se determinam permissões diferenciadas para os nós da rede. Uma vez autorizado a participar da rede, o nó pode gerar transações, gerar blocos e participar do consenso. Esta característica é interessante às aplicações em que nós, mesmo autorizados a participar da rede, oferecem um comportamento hostil. Nesses casos, são empregados mecanismos de consenso tolerantes a falhas Bizantinas (*Byzantine-Fault Tolerant* - BFT), exigindo que todos os nós participem do consenso. O *Hyperledger Fabric* é um exemplo de plataforma com opção de configuração de redes privadas não permissionadas.

As redes privadas permissionadas oferecem oportunidade de utilizar aplicações de mecanismos de consenso mais eficientes e menos custosos em termos de processamento. A característica privada limita a entrada e permanência de nós indesejáveis na rede. As permissões possibilitam configurar diferentes papéis para os nós participantes da rede como, por exemplo, oferecer flexibilidade para aplicações permitirem que apenas alguns nós façam parte do processo de consenso e apenas um subconjunto desses nós possa gerar o bloco seguinte. Dispositivos com mais capacidade de processamento e segurança exercem essas atividades na rede. A *Parity* e a *MultiChain* são exemplos de plataformas que permitem configurações de redes privadas permissionadas.

2.2 Consenso em Cadeias de Blocos

Os nós participantes de um sistema que utilize cadeia de blocos devem possuir uma visão global comum sobre a rede, para que não exista divergência entre as cópias das cadeias de blocos presentes em cada nó. Os blocos da cadeia são compostos por uma sequência de transações a serem executadas. Antes de serem executadas, os nós precisam alcançar um consenso, concordando com as transações inseridas no bloco e com a ordem em que serão executadas. O consenso consiste em regras para validação e difusão de transações e blocos, resolvendo potenciais conflitos [56] e alcançando uma consistência eventual da informação presente na rede. Ao se alcançar o consenso, garante-se a integridade, a consistência e a imutabilidade da cadeia de blocos.

O consenso é alcançado de forma distribuída, eliminando a necessidade de um agente central intermediário confiável. O tipo de mecanismo de consenso utilizado depende do tipo de rede de cadeia de blocos e do tipo de vetor de ataque esperado. São duas as principais classes de mecanismos de consenso: protocolos probabilísticos de consistência eventual e protocolos baseados em votação por maioria [13, 9]. Nos mecanismos baseados em consistência eventual, não é necessário saber o número de participantes disponíveis no consenso e eventualmente existe convergência sobre a cadeia de blocos, com base na disseminação da informação sobre o que cada participante enxerga como verdade. Já nos protocolos baseados em votação, é necessário conhecer todos os participantes do mecanismo. Dessa forma, consenso baseado em consistência eventual é adequado para cadeias de blocos públicas, enquanto os baseados em votação são mais adequados para cadeias de blocos privadas [13, 9]. Dentre os mecanismos de consenso utilizados nas redes de cadeia de blocos públicas, estão as Provas de Trabalho (*Proof of Work* - PoW), de Participação (*Proof of Stake* - PoS) e de Capacidade (*Proof of Capacity* - PoC).

Em redes de cadeia de blocos privadas, a necessidade de mecanismos de consenso custosos em termos computacionais, como a PoW, é reduzida [13] e, portanto, outros mecanismos de consenso podem ser utilizados, como a Prova de Autoridade (*Proof of Authority* - PoA), o Protocolo Prático de Tolerância a Falhas Bizantinas (*Practical Byzantine Fault Tolerance* - PBFT) [10] e os algoritmos Paxos [35], Raft [45] e Ripple [50].

2.2.1 Prova de Trabalho – PoW

Em redes públicas, tais como *Bitcoin* e *Ethereum*, uma única entidade pode participar da rede com múltiplas identidades para influenciar a votação sobre a validação de um

determinado bloco. A consequência imediata dessa possibilidade é o controle da rede por uma minoria [13]. Para desestimular essa prática, Nakamoto [44] propõe o uso de um mecanismo de consenso denominado Prova de Trabalho (*Proof of Work* - PoW). Na PoW, para que um bloco seja inserido na cadeia de blocos, os nós mineradores devem resolver um desafio não trivial, que exige grande poder de processamento para ser resolvido, mas cujo resultado pode ser facilmente verificado por qualquer nó da rede que seja participante do consenso. O bloco candidato a ser inserido na rede é composto pelas transações que foram submetidas, mas ainda não foram validadas. Quando o desafio é solucionado, o bloco é minerado, isto é, é inserido na cadeia de blocos global, e as transações são validadas e executadas. O nó minerador recebe uma recompensa por ter empenhado seu poder computacional para a resolução do desafio.

O desafio computacional exigido na PoW consiste em encontrar um número aleatório (*nonce*) que faz com que o resumo criptográfico (*hash*) do bloco tenha o número esperado de zeros iniciais para a dificuldade definida para aquela rede [44]. O *nonce* está contido no cabeçalho do bloco, juntamente com o resumo criptográfico do bloco anterior. Como mostra a figura 2.5. Quanto maior o número de zeros iniciais requeridos, mais complexo é o desafio computacional. Na *Bitcoin*, a rede ajusta a dificuldade do desafio a cada 2016 blocos, para levar em consideração mudanças no poder de processamento dos nós e para garantir que os blocos sejam gerados a uma taxa de aproximadamente 10 minutos. [13] e, atualmente, são exigidos resumos criptográficos que iniciam com 18 zeros.

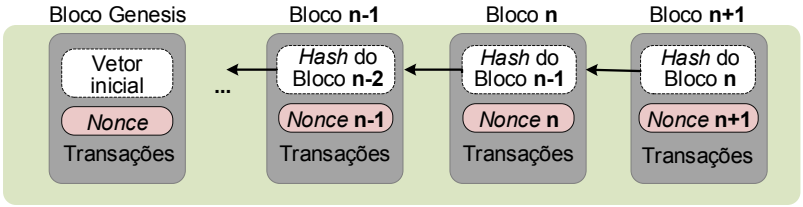


Figura 2.5: Cadeia de blocos em redes que utilizam consenso como prova de trabalho. A inserção do valor *nonce* possibilita alterar o resultado da operação de *hash* para atingir o desafio da rede.

Ao resolver o desafio computacional, o nó gera uma Prova de Trabalho e pode adicionar o novo bloco à cadeia de blocos, recebendo a recompensa pela resolução do desafio. Esse bloco é disseminado através da rede, para que os outros nós possam adicioná-lo a suas cópias da cadeia de blocos. Os outros nós que estavam trabalhando para solucionar o mesmo desafio param de tentar, uma vez que não haverá mais recompensa, caso resolvam o desafio após o bloco ter sido minerado [44]. Além disso, os blocos que os outros

nós estão tentando minerar agora referenciam o resumo criptográfico do bloco errado e podem ser compostos por transações que já foram mineradas, isto é, que estão no bloco recentemente inserido na cadeia pelo nó vencedor. Ao desistirem e aceitarem que o desafio foi solucionado por um minerador vencedor, o consenso é alcançado. Na PoW, é possível que vários nós reivindiquem o próximo bloco a ser adicionado à cadeia. Isso ocorre porque a PoW é um mecanismo de consenso probabilístico, no qual cada nó tem uma determinada probabilidade de concluir o desafio computacional antes de todos os outros nós da rede. Quando mais de um nó minera um bloco, ocorre uma ramificação da cadeia de blocos. Na PoW, a ramificação que carregar a maior quantidade de trabalho deve ser seguida. A ramificação da cadeia de blocos que crescer primeiro, através da inserção de novos blocos, será adotada como a cadeia correta, levando à poda dos outros ramos [44]. Os nós identificam como o último bloco da cadeia aquele que segue a cadeia mais longa e com o *timestamp* mais antigo. Isso permite que a rede alcance novamente o consenso sobre a ordem de ocorrência dos eventos e, então, obtenha uma visão global consistente da cadeia.

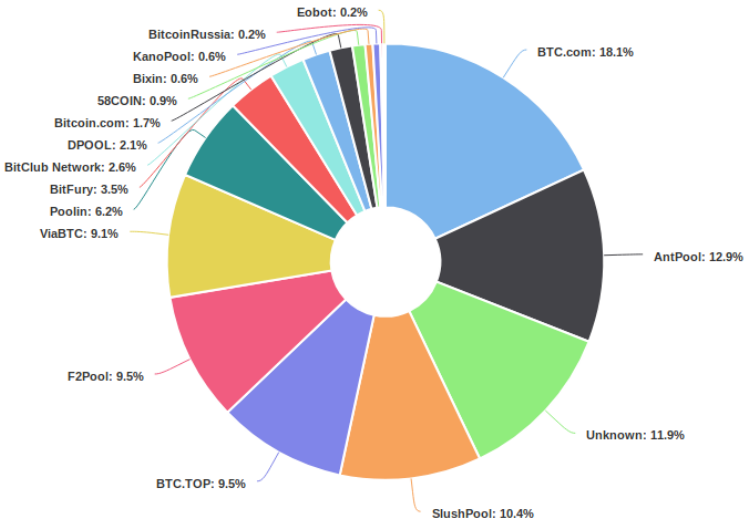


Figura 2.6: Distribuição do poder de geração de blocos na rede *Bitcoin* em 12 de Novembro de 2018[2].

Apesar de o mecanismo tender probabilisticamente à convergência, a Prova de Trabalho apresenta desvantagens, como a crítica à sustentabilidade do processo de mineração, em que há um gasto exacerbado de energia para criação de um bloco. Além disso, é obser-

[2]Dados disponíveis em https://www.blockchain.com/pt/pools.

vada alta latência para alcançar o consenso na rede. Como consequência, há baixa vazão na quantidade de transações validadas no tempo. O mecanismo de Prova de Trabalho pode ser comprometido, teoricamente, por um usuário que controle mais que 50% dos recursos computacionais da rede. Apesar da PoW desestimular que uma única entidade possua diversas identidades na rede, devido ao custo computacional para gerar a Prova de Trabalho, um grupo de nós mineradores pode compartilhar recursos para gerar blocos mais rapidamente, distorcendo a natureza descentralizada da rede. A Figura 2.6 mostra a estimativa de distribuição do poder de mineração entre os maiores aglomerados na rede *Bitcoin*, em que *BTC.com*, *SlushPool*, *AntPool* e *BTC.TOP* são, juntos, responsáveis por mais de 50% da taxa de *hashs* gerados.

2.2.2 Prova de Autoridade – PoA

No contexto das redes privadas, em vez da Prova de Trabalho, propõe-se o uso da Prova de Autoridade (*Proof of Authority* – PoA). Nas redes privadas, existe uma governança descentralizada que pré-determina o papel de alguns nós, ou cria métricas para automatizar a seleção do subgrupo de nós de autoridade. Assim, na Prova de Autoridade, a ideia é designar um conjunto de nós de autoridade com permissão para participar do consenso. Esses nós são encarregados da tarefa de gerar novos blocos e validar as transações. A PoA endossa um bloco como parte da cadeia, se ele for assinado por pelo menos um nó com autoridade. O modelo de incentivo no PoA destaca que é do interesse de um nó de autoridade manter sua reputação, para permanecer como nó de autoridade.

Plataformas que se baseiam em PoA como mecanismo de consenso aplicam um tipo de rotação entre os nós de autoridade, para que cada um tenha um tempo para gerar blocos alternadamente, sem disputa e sem desperdício de recursos por mais de um nó. Após o nó minerador da rodada minerar o último bloco, todos os nós de autoridade devem concordar e adicioná-lo ao final da cadeia. O mecanismo PoA deve ser usado em ambientes confiáveis, em que é possível prever o comportamento dos nós de autoridade. No entanto, para que a PoA mantenha a natureza distribuída da rede, é preciso adotar um modelo de confiança associado ao mecanismo de consenso. O modelo deve ser capaz de monitorar o comportamento dos nós de autoridade. Caso seja observada alguma falha do nó de autoridade, é preciso que a plataforma ofereça recursos para fiscalizar e retirar a autoridade desse nó e, como consequência, desconsiderar seus blocos minerados, retornando as transações para o conjunto de transações não mineradas [9].

[2]Dados disponíveis em https://www.blockchain.com/pt/pools.

2.2.3 Prova de Participação – PoS

A Prova de Participação (*Proof of Stake* - PoS) é uma variação da Prova de Autoridade aplicada em redes públicas como alternativa ao alto custo computacional da Prova de Trabalho para alcançar o consenso, preservando a natureza descentralizada da rede pública. Na PoW, a probabilidade de um nó conseguir minerar um bloco depende do poder computacional de cada nó. Já na PoS, essa probabilidade passa a depender da participação dos nós na rede. Os nós mineradores precisam acumular riqueza por mais tempo que os outros, representada pelo *coin age*. A dificuldade para encontrar esse resumo criptográfico do bloco é inversamente proporcional à riqueza acumulada (*coin age*) daquele nó, definida como a quantidade de recursos do nó multiplicada pelo período em que o nó reteve aquele recurso. Por exemplo, se Bob recebeu 10 recursos de Alice e manteve a posse desses recursos por 90 dias, Bob acumulou uma riqueza igual a 900 $recursos \times dias$. Quando Bob gasta esses 10 recursos provenientes de Alice, ele consome, ou destrói, o valor de riqueza acumulado devido a esses 10 recursos. A consequência imediata de se utilizar o conceito de riqueza acumulada na PoS é atribuir ao nó com maior participação - com maior riqueza acumulada - a oportunidade de gerar o bloco seguinte [52].

Na PoS, é necessário que cada transação possua um campo de marcação do tempo de geração, para que o valor da riqueza acumulada seja calculado. O conjunto de nós que deseja atuar como minerador precisa necessariamente bloquear seus recursos por um determinado tempo. Isso é feito através da construção de um bloco especial, *coinstake*, no qual o proprietário do recurso emite uma transação para si mesmo, adicionando determinada taxa de transação como recompensa [52]. No momento em que o minerador paga a si mesmo, o valor da sua riqueza acumulada (*coin age*) é consumido. Dessa forma, na próxima rodada de mineração, outros nós têm a chance de conseguir minerar o novo bloco. A primeira entrada do bloco *coinstake* é composta por um *kernel* que deve obedecer a um protocolo de geração de resumos criptográficos específico da rede.

As tentativas de geração do resumo criptográfico correto ocorrem à taxa de uma tentativa por unidade de riqueza acumulada. Assim, quanto mais recursos o nó disponibilizar para tentar encontrar o *kernel* correto, maior será a sua chance de reivindicar a oportunidade para minerar o bloco seguinte. Por exemplo, se Alice possui uma riqueza acumulada de 100 $recursos \times dias$, para a qual se espera a geração de um *kernel* em dois dias, então Bob, que possui uma riqueza acumulado de 200 $recursos \times dias$, pode esperar que o *kernel* seja gerado na metade do tempo. Com a PoS, a probabilidade de gerar o *kernel* independe do poder computacional dos nós da rede [34]. Caso exista uma ramificação

da cadeia, será declarada como principal aquela que possuir a maior soma de riqueza acumulada consumida [34]. Existe a possibilidade de as ramificações crescerem na PoS, quando a implementação não fornece incentivo para que os nós adicionem blocos à cadeia correta, originando o problema conhecido como "nada a perder"(*nothing-at-stake*) [33]. Dessa forma, os nós adicionam blocos a múltiplos ramos, para maximizar a probabilidade de receber uma recompensa, prejudicando a obtenção de uma visão global única da cadeia de blocos.

2.2.4 Prova de Capacidade – PoC

O protocolo de consenso de Prova de Capacidade foi desenvolvido como alternativa à Prova de Trabalho. Introduzido com a criptomoeda *Burstcoin*[3], o algoritmo permite utilizar o espaço de armazenamento de memória secundária, ao invés do poder computacional bruto, para determinar o nó minerador. Assim, a PoC é baseada no espaço disponível do disco rígido do nó. O mecanismo funciona configurando o disco rígido para reservar um espaço de armazenamento em um processo chamado "plotagem"(*plotting*). Com a plotagem no disco rígido, os cálculos são feitos de antemão e as possíveis soluções são armazenadas em disco. Algumas dessas soluções, ou parcelas das soluções, permitem alcançar a solução final mais rapidamente do que outras e o nó que alcançar primeiro a solução final para o bloco mais recente será recompensado pela mineração. Isso essencialmente permite que o nó gere uma receita passiva utilizando o espaço de armazenamento disponível. Em outras palavras, os nós mineradores pré-geram pedaços de dados conhecidos como grafos, que são salvos no disco. O número de parcelas que o nó armazena é efetivamente sua velocidade de mineração. A cada bloco, o nó minerador correrá as parcelas salvas e obterá uma quantidade de tempo até poder extrair um bloco, se outro bloco ainda não tiver sido encontrado. Depois de ler as plotagens, o *hardware* permanece ocioso até o próximo bloco. Quanto mais grafos o nó obtiver em seu disco rígido, maiores serão suas chances de resolver o próximo bloco. O algoritmo de Prova de Capacidade mostra um potencial para a evolução das criptomoedas, pois requer menos energia que a PoW baseada em ASIC (*Application Specific Integrated Circuits*) [25].

2.2.5 Protocolos de Consenso Baseados em Votação

Todos os mecanismos de consenso descritos anteriormente eventualmente alcançam a consistência da visão global da cadeia de blocos existente na rede. A consistência eventual

[3]Disponível em https://bitcointalk.org/index.php?topic=731923.0.

é alcançada sem a necessidade de conhecer o número de participantes do consenso. Outro grupo de mecanismos de consenso é composto por protocolos baseados em votação. Para alcançar o consenso na rede, é necessário que determinada fração dos participantes do consenso entre em comum acordo quanto à adição do bloco na cadeia de blocos. Para tanto, um grupo de nós da rede participa do consenso, votando a favor ou contra qualquer proposta de modificação nos dados do sistema. Alguns exemplos desse grupo de protocolos são PBFT [10], Ripple [50], Paxos [35] e Raft [35].

Os protocolos baseados no **Modelo Prático de Tolerância a Falhas Bizantinas** (PBFT) consideram que os nós da rede podem exercer comportamentos arbitrariamente maliciosos ou falhas que fogem do protocolo pré-definido [5, 11, 1]. Apesar da participação de nós maliciosos, os protocolos baseados no PBFT garantem o consenso entre os nós da rede até o número limite de nós maliciosos, chamados de nós bizantinos, atingir f, em que $f \leq \frac{n+1}{3}$ e n representa o número total de nós da rede. O mecanismo de consenso PBFT pode ser resumido em quatro fases. Na primeira fase é determinado o nó líder da rodada de mineração, que geralmente segue um rodízio entre os nós da rede, considerando que todos os nós são iguais e participam do consenso. Na segunda fase, o nó líder gera um novo bloco e o encaminha para todos os nós da rede. Na terceira fase, o nó líder aguarda a resposta de no mínimo $f + 1$ nós com o mesmo resultado para o novo bloco. A quarta fase é de execução das transações contidas no bloco, visto que o consenso foi alcançando e os nós compartilham da mesma visão da cadeia. Como resultado final, todos os nós legítimos chegam a um acordo sobre a ordem das transações e as aceitam ou rejeitam. Além disso, algumas aplicações de PBFT oferecem opções nas quais a maioria absoluta de nós legítimos pode decidir se um líder está com defeito, ou sendo desonesto. A maioria pode votar para removê-lo da rotação de líder nas próximas rotações para a geração de blocos. Em comparação à PoW, os protocolos baseados em PBFT são vantajosos em termos de processamento. Por outro lado, estes protocolos exigem grande complexidade de mensagens, $O(n^2)$, o que gera um problema de escalabilidade para a rede. Consequentemente, protocolos baseados em PBFT são adequados para redes com poucos nós.

Schwartz *et al.* propõem o protocolo **Ripple** como mecanismo de consenso distribuído para cadeias de blocos federadas[4] [50]. O Ripple é tolerante a falhas bizantinas e é robusto contra ataques de conluio. Essa robustez advém da criação de subconjuntos de zonas confiáveis, nas quais não se espera uma conspiração entre os nós para atacar o sistema. Dessa forma, os nós consultam apenas o subconjunto de nós confiáveis para alcançar

[4]Cadeias de blocos federadas são cadeias privadas sob a liderança de um grupo ou consórcio de nós.

o consenso. Um dos problemas do Ripple é a escalabilidade quanto ao número de nós designados para alcançar o consenso [9].

Paxos [35] e **Raft** [45] são protocolos equivalentes, com o objetivo de gerenciar registros replicados de entradas de dados. Primeiramente é eleito um líder, que recebe todas as propostas de modificação de dados no sistema. O líder torna-se responsável por compartilhar todas as modificações com todos os outros nós, para que eles possam votar. Em seguida, o líder compartilha a decisão coletiva com todos os outros nós participantes do consenso. Ambos os protocolos são resilientes a f falhas, em que $f \leq \frac{n+1}{2}$ e n representa o número total de nós da rede. Esses protocolos foram desenvolvidos para serem usados em ambientes confiáveis, uma vez que não consideram o comportamento malicioso de nós que participam do consenso. Dessa forma, só devem ser utilizados em cadeias de blocos privadas. Variações dos protocolos Paxos e Raft consideram a redução do número de fases para alcançar o consenso com menor número de mensagens e a assinatura das mensagens trocadas permite extrapolar o uso desses protocolos em ambientes não confiáveis [40, 9].

2.3 Plataformas para Desenvolvimento de Cadeias de Blocos

A **Bitcoin**[5] é uma plataforma para desenvolvimento de aplicações para cadeias de blocos proposta por Satoshi Nakamoto [44]. Ela permite a criação de redes públicas não permissionadas nas quais um nó pode participar e exercer qualquer função na rede. Essa característica faz com que certos critérios de confiabilidade sejam exigidos de seus participantes, por isso esta plataforma utiliza a Prova de Trabalho como mecanismo de consenso. Antonopoulos define que as funções exercidas por cada nó podem ser divididas em: (i) cliente de referência, que possui todas as funções, (ii) nó completo que não exerce a função de mineração, (iii) nó leve que apenas interage com a rede para enviar e receber transações e (iv) nó minerador que é apenas responsável por executar a prova de trabalho [2].

A *Bitcoin* permite aos seus usuários criar regras que restringem o modo como os valores enviados através das transações são gastos. Apenas usuários capazes de satisfazer essas regras podem ter acesso aos valores a eles relacionados. Para expressar essas regras, a *Bitcoin* implementa uma linguagem denominada *Script*. Um usuário, ao criar uma transação, acrescenta um código executável que define as condições para gastar aquele valor. Ao utilizar esse valor para criar outra transação, deve-se acrescentar outro código

[5]Disponível em https://bitcoin.org/.

executável que, quando executado juntamente ao anterior, resulte em uma execução bem-sucedida. A *Script* é uma linguagem baseada em pilha e não é do tipo *Turing* completa, ou seja, ela não possui estruturas de laços. Essa restrição se faz necessária para impedir que ataques de negação de serviço sejam feitos contra um nó através da criação de laços infinitos. A regra mais utilizada para criar transações é a que atrela um dado valor à chave pública de outro usuário e, assim, garante que apenas ele possa gastá-lo. Em suas primeiras versões, a *Bitcoin* permitia somente a criação de soluções financeiras baseadas na troca de valores. No entanto, as versões mais recentes da plataforma permitem o envio de dados no lugar de apenas valores. Aplicações não financeiras foram propostas sobre a *Bitcoin*, como o *FairAccess* [46], cujo objetivo é prover controle de acesso a recursos, utilizando a estrutura das transações como meio para enviar as permissões. A linguagem *Script* é usada para expressar as condições de acesso aos recursos.

A **MultiChain**[6] é uma plataforma para desenvolvimento de aplicações utilizando cadeias de blocos, cujo projeto foi desenvolvido com base na implementação da *Bitcoin*. A plataforma é totalmente compatível com o protocolo e capaz de funcionar como um nó da rede *Bitcoin*. A *MultiChain* permite a criação de redes privadas permissionadas [26], o que exige a participação de um administrador na rede, que é responsável por permitir a entrada e gerenciar as permissões dadas aos nós. As permissões variam desde a capacidade de executar buscas na cadeia, até a permissão para um nó ser minerador ou tornar outro nó administrador. O papel de administrador é concedido ao nó responsável por criar o bloco *gênesis* da rede. Sempre que um administrador concede ou revoga uma permissão de um nó, esse evento fica registrado na cadeia através de uma transação especial. Dessa forma, todos os demais nós são capazes de verificar quais permissões estão vinculadas a cada chave pública da rede.

Usuários de uma rede *MultiChain* são capazes de criar e gerenciar seus próprios ativos, através de uma transação especial, chamada "Transação Gênesis", que contém os metadados necessários para registrar o canal na cadeia de blocos [26]. Para tanto, o usuário deve possuir permissão concedida por um administrador da rede. Após a criação do canal, o criador assume o papel de administrador, podendo decidir quem pode enviar, receber e criar novos ativos naquele canal. A criação de canais privados permite que apenas os usuários que tenham interesse em certo tipo de ativo obtenham acesso ao canal e às informações presentes na cadeia relativa ao ativo.

Ao contrário da plataforma *Bitcoin*, a *MultiChain* não possui suporte para criação

[6]Disponível em https://www.multichain.com/.

de regras através da linguagem *Script*. Logo, um usuário é apenas capaz de enviar e receber ativos pela rede. No entanto, quando utilizada para se conectar à rede *Bitcoin*, a plataforma fornece suporte à criação de regras através da *Script*. O mecanismo de consenso oferecido pela *MultiChain* permite que, através da configuração de parâmetros no momento da criação da rede, o consenso funcione como PoW ou como PoA - cada minerador alterna a criação de um novo bloco, sem a necessidade de haver competição. Isso permite que haja maior flexibilidade na criação de novas redes e elimina os altos custos computacionais da Prova de Trabalho. A manutenção da Prova de Trabalho permite a retrocompatibilidade com a rede *Bitcoin*.

A **Ethereum**[7] é uma plataforma criada e mantida pela *Ethereum Foundation* e representa o início da segunda geração das plataformas de cadeias de blocos [7]. Essa plataforma foi desenvolvida com o intuito de possibilitar a realização de novas transações, além das transações de troca de posse de ativos. A plataforma possibilita a criação de aplicações que, através da invocação de contratos inteligentes, podem exercer diferentes funções na rede. A utilização de contratos inteligentes estabeleceu uma nova geração de aplicações de cadeias de blocos, pois o desenvolvedor de aplicações na plataforma *Ethereum* é capaz de criar aplicações que interagem com a rede *Ethereum* de maneira transparente ao usuário final. A principal proposta da plataforma *Ethereum* é a criação de aplicações que utilizem a cadeia de blocos da própria rede *Ethereum*. Contudo, a plataforma possibilita a criação de redes públicas não permissionadas e, assim como na rede *Bitcoin*, a rede *Ethereum* possui sua própria moeda corrente, o *Ether*. A plataforma possibilita a realização de transações financeiras simples, sem necessidade de utilização de contratos complexos. O mecanismo de consenso utilizado pela *Ethereum* é a Prova de Trabalho, mas a versão implementada por esta plataforma impede a utilização de *hardware* especializado para a otimização do tempo de resolução do desafio computacional. Isso garante melhor distribuição das capacidades de mineração através da rede [7]. Segundo seus desenvolvedores, em uma versão futura, será acrescentado suporte à utilização da Prova de Participação (*Proof of Stake*, PoS) como uma alternativa à Prova de Trabalho.

A **Parity**[8] inicialmente foi proposta como um meio de interação com a rede *Ethereum*. Assim, a *Parity* possui todas as características presentes na plataforma *Ethereum*. Além disso, a *Parity* agrega uma ferramenta para desenvolvimento e depuração de aplicações baseadas em contratos inteligentes utilizando a linguagem nativa da *Ethereum*, a *Solidity*. A *Parity* possui suporte para a utilização de carteiras digitais e para o gerenciamento de

[7]Disponível em https://www.ethereum.org/.
[8]Disponível em https://www.parity.io/.

chaves de usuários. A *Parity* estende as funções padrão da plataforma *Ethereum*, permitindo a criação de redes privadas permissionadas, que possuem compatibilidade com a rede *Ethereum* e permitem que aplicações desenvolvidas para a *Ethereum* funcionem normalmente na rede privada *Parity*. Para a criação de redes privadas, a *Parity* fornece suporte ao mecanismo de consenso original da *Ethereum*, a Prova de Trabalho, porém também possui implementação do mecanismo PoA. Nessa implementação, os nós mineradores se revezam para criar blocos dentro de uma janela de tempo determinada no momento da criação da rede.

O **Hyperlegder**[9] é um projeto formado por diversas corporações, como a Linux Foundation, a IBM e a Intel, que visam a desenvolver soluções para promover a aplicação comercial e incentivar os estudos sobre a tecnologia de cadeia de blocos [27]. Um dos desdobramentos do projeto é a plataforma **Hyperledger Fabric**, proposta pela IBM [8]. A plataforma permite a criação de soluções empresariais baseadas no uso de cadeias de blocos privadas permissionadas. A arquitetura modular permite que serviços específicos de rede sejam distribuídos entre nós especializados, o que permite que sejam oferecidos altos graus de confiabilidade, resiliência, flexibilidade e escalabilidade [27].

Os nós da rede podem fornecer serviços de validação, consenso, controle de credenciais e armazenamento. Nós que fornecem serviços de validação são responsáveis por conectar clientes aos serviços de consenso, através da emissão de transações. Esses nós não possuem capacidade para executar as transações, apenas para verificá-las. O serviço de consenso é oferecido pelos nós responsáveis por executar o mecanismo de consenso, que nesta plataforma consiste em uma implementação do Modelo Prático de Tolerância a Falhas Bizantinas (*Practical Byzantine Fault Tolerance* - PBFT). Esses nós também são responsáveis por validar as transações. O serviço de consenso também é responsável por atualizar o estado da cadeia de blocos. O controle de credenciais é o serviço responsável por criar os certificados que identificam os usuários da rede. Através dessa identificação, é possível exercer o controle de permissão da rede. Ao realizar uma transação, um usuário deve obrigatoriamente se identificar através de seu certificado. O serviço de armazenamento é oferecido por todos os nós da rede, com exceção dos nós que realizam controle de credenciais. O serviço consiste em armazenar uma cópia da cadeia de blocos e permitir a realização de consultas. As informações da cadeia de blocos são armazenadas em um banco de dados não relacional e apenas as referências aos dados ficam na cadeia. Esse procedimento torna a rede escalável, pois reduz o espaço necessário para armazenar uma cópia local da cadeia e reduz o tempo para realizar buscas por informações. A Hypeledger

[9]Disponível em http://www.hyperledger.org.

Fabric possui suporte para a criação de contrato inteligente, denominado *ChainCode*. Esses contratos ficam armazenados na cadeia e, quando invocados, são executados pelos nós que implementam o serviço de consenso. A linguagem utilizada para o desenvolvimento dos contratos é a GO[10]. Para adicionar um contrato a uma cadeia, um usuário executa uma transação, ficando assim também registrado o momento da criação do contrato.

A plataforma **R3 Corda**[11] foi desenvolvida com o propósito de se diferenciar das demais plataformas de desenvolvimento de cadeias de blocos, já que propõe o isolamento dos dados de seus usuários. O isolamento ocorre através da criação de canais de comunicação entre usuários que estejam interessados em realizar transações entre si, formando pequenas cadeias de blocos acessíveis apenas entre eles. A *Corda* possibilita a criação de cadeias privadas permissionadas. Para que um usuário possa fazer parte de uma rede, é necessário que ele possua os certificados gerados pela entidade responsável por administrar aquela rede. Uma vez na rede, o usuário pode se associar a canais com participantes com os quais deseja realizar transações.

Contratos inteligentes são tratados de maneira diferente pela *Corda*, uma vez que esses contratos são disponibilizados na forma de aplicativos, que podem ser instalados em nós específicos da rede, sem a necessidade de estarem disponíveis em todos os nós. Esses aplicativos são responsáveis por realizar a interação dos usuários com a rede. Para o desenvolvimento das aplicações, a *Corda* oferece bibliotecas nas linguagens *Java* e *Kotlin*, porém qualquer linguagem compatível com a máquina virtual Java (JVM) pode ser utilizada. O desenvolvedor pode criar uma interface gráfica e disponibilizá-la através de um servidor *web* integrado à sua aplicação. Isso torna a utilização das aplicações mais amigáveis aos usuários. Na versão atual, versão 3.0, a *Corda* disponibilizada três opções para mecanismos de consenso, o *Raft*, o *bftSMaRt* e o consenso personalizado. O *Raft* é um protocolo simplificado de consenso por votação, mas não resiste a falhas bizantinas [45]. O *bftSMaRt* consiste em uma implementação do modelo prático de tolerância a falhas bizantinas [5]. A aplicação (*Cordapp*) pode fornecer implementação própria de um mecanismo de consenso personalizado. O armazenamento dos dados da rede é feito através da utilização de bases de dados externas à cadeia em cada nó. As bases de dados armazenam apenas as transações em que tiveram participação, restringindo o acesso aos dados apenas a usuários que utilizam os canais aos quais o nó pertence.

[10]Disponível em https://golang.org/.
[11]Disponível em http://www.corda.net/.

Capítulo 3

Comparação de Desempenho entre as Plataformas de Desenvolvimento*

Com o objetivo de aplicar da tecnologia de cadeia de blocos para o armazenamento de registros médicos eletrônicos (EMR), entende-se que a abordagem de rede privada permissionada possibilita maior controle sobre o acesso aos dados sensíveis dos pacientes e a utilização de mecanismos de consenso não competitivos. Por isso, é importante entender o funcionamento de plataformas de cadeia de blocos que se enquadram nessa taxonomia e identificar padrões de comportamento, visto que as documentações não são completas e muitas das plataformas ainda estão em fase de testes. Este capítulo apresenta a comparação de desempenho entre duas plataformas de desenvolvimento de aplicações de cadeia de blocos. As plataformas avaliadas são a *Multichain*, baseada na cadeia blocos original da *Bitcoin*, e a *Parity*, que é executada sobre a cadeia de blocos da *Ethereum*. Essas plataformas são apontadas como as principais soluções para o desenvolvimento de cadeias de blocos privadas permissionadas [17]. São comparados os tempos de validação de uma transação, de mineração de blocos e de busca por transações de blocos já inseridas na cadeia. A avaliação das plataformas é realizada através da inserção de transações geradas seguindo a probabilidade real de chegada de novas transações na rede *Bitcoin*. Por fim, propõe-se a modelagem da distribuição de probabilidades de cada parâmetro avaliado.

*Este capítulo é baseado no artigo "Uma Avaliação de Desempenho de Cadeias de Blocos Privadas Permissionadas através de Cargas de Trabalho Realísticas" [16]. Agradecimentos pela colaboração dos autores Gabriel R Carrara, Natalia C Fernandes, Ricardo C Carrano, Célio V. N. Albuquerque and Dianne S. V. Medeiros e Diogo M. F. Mattos.

3.1 Trabalhos Relacionados

Alguns trabalhos apresentam comparações qualitativas das tecnologias e propostas relativas às cadeias de blocos. Zyskind *et al.* verificam ameaças à privacidade dos dados em serviços *online*. Os autores afirmam que a necessidade de uma terceira entidade fragiliza a privacidade dos usuários, que não têm controle sobre quais dados estão sendo coletados e armazenados [62]. Dai *et al.* abordam o problema de escalabilidade, apresentando opções de descarte dos blocos mais antigos da cadeia e consideram a abordagem de operar sobre os resumos dos dados [14]. Pahl *et al.* comparam as características de diferentes cadeias de blocos, para propor um arcabouço de auxílio à decisão sobre qual tecnologia utilizar [47]. Wang *et al.* apresentam uma visão geral sobre as tecnologias de cadeias de blocos, enfatizando suas diferenças arquiteturais e comparando seus algoritmos de consenso [60]. Nessa mesma linha, Julien *et al.* realizam uma comparação entre as plataformas *Ethereum*, IBM *Open Blockchain* (OBC), Intel *Sawtooth Lake*, *BlockStream Sidechain Elements* e *Eris*, relacionada à usabilidade, à flexibilidade e ao desempenho [38].

Dinh *et al.* foram pioneiros ao desenvolverem uma plataforma analisadora de desempenho para estudar e comparar plataformas de cadeias de blocos privadas. Segundo os autores, o *Blockbench* visa a testar as plataformas escolhidas, implementado cargas de trabalho em forma de contratos inteligentes. O objetivo da plataforma é comparar e compreender a fundo as diferentes organizações de cadeias de blocos privadas [17]. Xu *et al.* realizaram uma análise de desempenho em cadeias de blocos baseadas no problema de consenso bizantino (*Practical Byzantine Fault Tolerance* - PBFT). A análise foca na questão da escalabilidade, mostrando que as plataformas *Hyperledger Fabric* v0.6 com consenso PBFT, *Ripple* com algoritmo de consenso XRP e *Hyperledger Fabric* v1.0 baseado em consenso BFT-SMaRt não escalam mais que algumas dezenas de dispositivos [31]. Uma abordagem diferente foi proposta por Zagar *et al.* que compararam as cadeias de bloco com base em seu consumo de energia. O foco está na verificação dos blocos, analisando os algoritmos de consenso passo a passo [4].

Aplicações de cadeia de blocos são propostas para armazenar dados distribuídos e executar ações distribuídas em diversos campos do conhecimento [43, 62, 29]. Outras propostas visam à criação de novas ferramentas de cadeias de blocos que não se baseiam em plataformas já estabelecidas [1, 49]. Ao considerar a comparação entre plataformas para a criação de cadeias de bloco, a proposta *BlockBench* se destaca ao propor um arcabouço de avaliação [17]. Contudo, essas propostas não focam na modelagem do comportamento das plataformas. Neste capítulo, implementam-se duas redes de testes, uma para cada

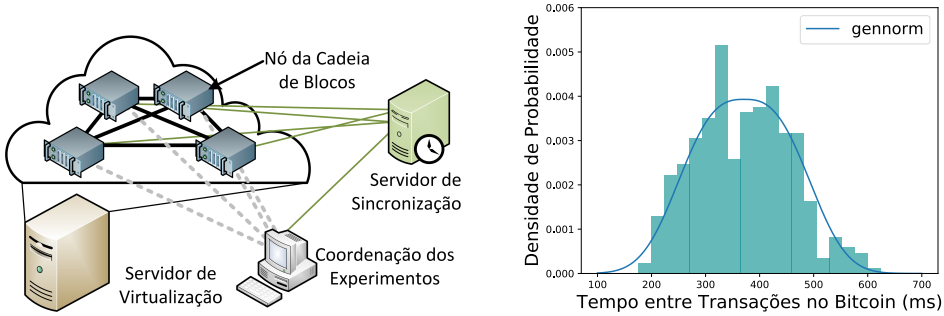
plataforma avaliada, e compara-se o desempenho das plataformas. Ademais, os resultados mostram que a plataforma *Multichain* possui o melhor desempenho ao se considerar o tempo de efetivação total das transações, ao custo de permitir a execução de transações simples, quando comparada à execução de códigos complexos de contratos inteligentes [16].

3.2 Avaliação Experimental de Plataformas de Cadeia de Blocos

A avaliação de desempenho das plataformas *Multichain* e *Parity* é realizada aplicando uma carga de trabalho realística sobre a rede implantada. As redes de cadeia de bloco são implantadas em um ambiente virtualizado, com 10 nós virtuais com isolamento de CPUs, criados sobre a plataforma de virtualização VMWare ESXi 5[1], em um servidor com dois processadores Xeon E5-2650, em que cada nó da rede da cadeia de blocos é configurado com 4 GB de RAM e 1 núcleo de processamento virtual, mostrados na Figura 3.1(a). O cenário de testes conta ainda com um computador usado para monitorar e coordenar os experimentos e um servidor de sincronização de tempo NTP (*Network Time Protocol*), para garantir que todos os nós das redes implantadas e o computador de coordenação estejam com a mesma referência de tempo. A carga de trabalho é obtida a partir da distribuição de probabilidade do tempo entre chegadas de transações da rede *Bitcoin* no período entre junho de 2017 e junho de 2018[2]. A Figura 3.1(b) mostra que o tempo de ocorrência entre transações segue uma distribuição normal generalizada, com $\mu = 0.371$, $\alpha = 0.143$ e $\beta = 2.786$. Em todos os experimentos as distribuições de probabilidade que melhor definem os dados são calculadas pelo método dos mínimos quadrados e o ajuste da distribuição aos dados é realizado pela biblioteca *Stats* do pacote *Scipy* da linguagem *Python*.

Os experimentos consistem na geração e envio de transações para as cadeias seguindo a carga de trabalho da *Bitcoin* durante o período de uma hora. Cada uma das máquinas virtuais executa um *script* em *Python 3* em que é sorteado um valor, segundo a distribuição normal generalizada, que representa o tempo em que cada máquina deve esperar para emitir uma nova transação. As dez máquinas em conjunto componhe o tempo entre transações na rede, similar ao tempo medido na rede *Bitcoin*. Neste cenário, as cargas de trabalho representam clientes da rede enviando transações uns para os outros através dos nós da rede. A carga é configurada para que cada nó aguarde um intervalo de

[1]Disponível em https://www.vmware.com/ .
[2]Disponível em https://blockchain.info/



(a) Cenário de execução das redes.

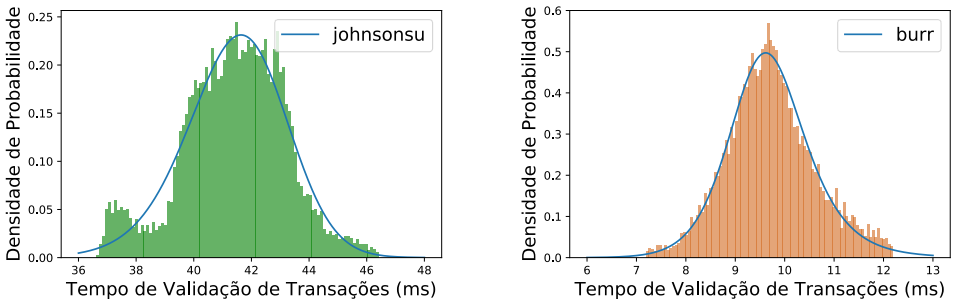
(b) Intervalo entre transações na *Bitcoin*.

Figura 3.1: Avaliação das redes de cadeia de blocos. a) Cenário de avaliação com cadeia executando em ambiente virtual. b) Distribuição do tempo estimado entre chegadas de transações na rede *Biticoin* estimado no período entre junho de 2017 e junho de 2018. Os valores seguem uma distribuição normal generalizada com $\mu = 0.371$, $\alpha = 0.143$ e $\beta = 2.786$.

tempo definido pela variável aleatória que segue a distribuição observada na Figura 3.1(b). As execuções das redes são realizadas usando configurações padrão recomendadas pelos desenvolvedores da *Parity*[3] e da *Multichain*[4], para a criação de redes privadas. Os parâmetros que determinam o funcionamento do mecanismo de consenso são determinados no momento da construção das redes de cadeia de blocos. Os parâmetros são definidos para que as redes apresentem o comportamento mais similar possível, a mérito de justiça na comparação. A execução de cada rede é iniciada com os nós conectados, porém sem qualquer transação sendo submetida. Em seguida, as cargas de trabalho são aplicadas e cada nó submete transações enviando uma unidade da moeda corrente de sua posse para outros nós vizinhos. Esse processo é executado durante uma hora. Durante essa etapa são armazenados os registros de tempo de envio das transações e o tempo decorrido para que uma submissão seja aceita. Além disso, também são armazenados os identificadores de cada transação submetida, para posteriormente realizar o experimento de tempo de busca das transações na cadeia de blocos.

O tempo de validação da transação representa o tempo desde que um usuário envia uma transação, já assinada, para um nó e este realiza o processo de validação, verificando se a transação está devidamente formada. Registrando-se o momento em que a transação é submetida e o tempo em que é recebida a confirmação de que a transação é válida, é possível calcular o tempo gasto para que uma transação seja validada. A Figura 3.2

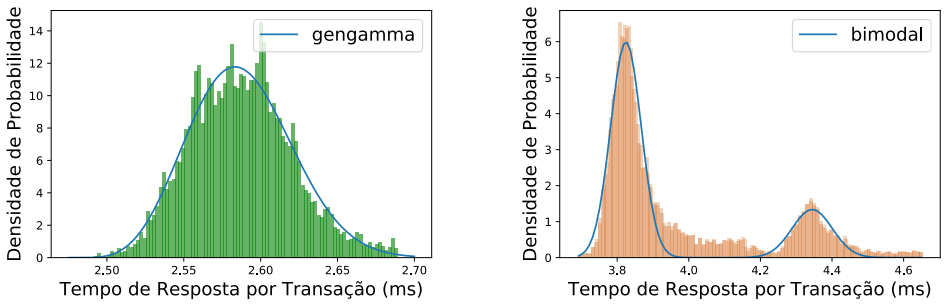
[3]Disponível em https://parity.io/
[4]Disponível em https://multichain.com/



(a) *Parity* (b) *Multichain*

Figura 3.2: Desempenho medido em tempo de validação de transações. A *Multichain* é até quatro vezes mais rápida para validar uma transação. a) O tempo de validação segue uma distribuição S_u de Johnson. b) O tempo de validação segue uma distribuição de Burr, log-logística generalizada.

apresenta os resultados para o tempo de validação de transações das duas plataformas. Ressalta-se que o comportamento do tempo de validação das transações na *Parity* segue uma distribuição S_u de Johnson, uma variação da distribuição normal, enquanto na *Multichain*, o tempo de validação segue uma distribuição de Burr, distribuição log-logística generalizada. Na Figura 3.2(a), observa-se que o tempo de validação na plataforma *Parity* variou entre 37 ms e 46 ms, com a maior concentração de validações acontecendo em um intervalo em torno de 42 ms. A Figura 3.2(b) apresenta os resultados observados para os tempos de validação de transações na plataforma *Multichain*, que variaram entre 7 ms e 12 ms, tendo maior concentração de acontecimentos entre 9 ms e 10 ms. Portanto, é possível afirmar que a *Multichain* é, em média, quatro vezes mais rápida para validar uma transação. Essa vantagem pode ser justificada pela simplicidade das transações na *Multichain*, que suporta somente transferência de ativos, enquanto a *Parity* suporta contratos inteligentes e tem uma mecanismo de validação mais complexo. O tempo de busca por uma transação representa o tempo que a requisição de busca por uma determinada transação leva para retornar uma resposta válida. A busca é feita utilizando como parâmetro o identificador da transação, que busca a transação desejada localmente na cadeia armazenada. O tempo de busca é obtido registrando o instante em que a requisição é enviada e o instante em que a resposta da requisição é recebida. O tempo de busca é a diferença entre o instante de submissão e o de resposta. A Figura 3.3(a) mostra o tempo de buscas por transações na *Parity*, que variam entre 2,5 ms e 2,7 ms, seguindo a função de distribuição de probabilidade Gamma generalizada. A Figura 3.3(b) apresenta os resultados da *Multichain* para as buscas por transação, observando que os resultados têm dois picos de concentração no histograma, o primeiro e maior em torno de 3,8 ms, e o segundo

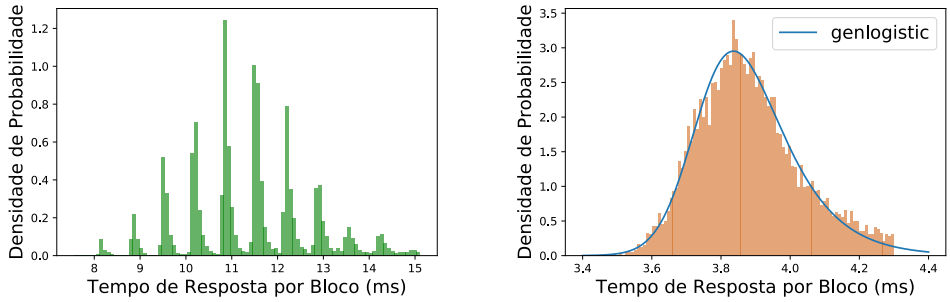


(a) *Parity* (b) *Multichain*

Figura 3.3: Desempenho medido em tempo de busca por transações. O tempo de resposta para a busca por transações é, em média, duas vezes menor na *Parity* que na *Multichain*. a) O tempo de busca de transações segue a distribuição Gamma generalizada. b) A*Multichain* apresenta uma distribuição de tempo para busca de transações bimodal, indicando a presença de diferentes níveis de armazenamento dos dados.

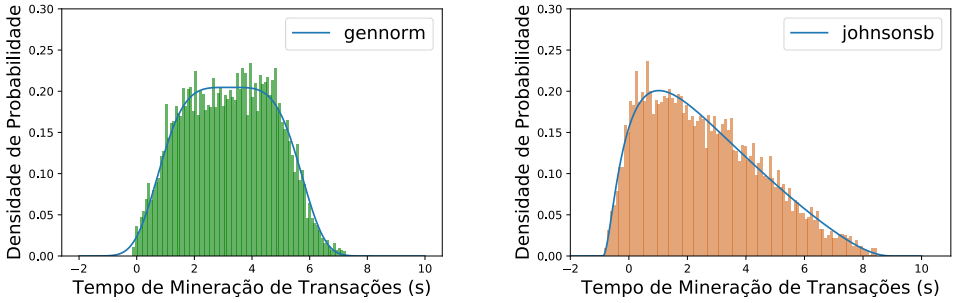
em 4,35 ms, com uma distribuição bimodal formada por duas gaussianas com as médias nos dois picos de concentração. O comportamento da distribuição bimodal apresentada pela plataforma *Multichain* pode ser justificado pela utilização de um *cache* de memória volátil, que armazena os blocos recém acessados na busca por transações, em detrimento de blocos não acessados que são armazenados em disco. Isto faz com que buscas por transações armazenadas no mesmo bloco de uma transação pesquisada recentemente não sejam feitas novamente em toda a cadeia, mas somente na memória volátil. Por isso, existe um pico em torno de 3,8 ms que representam as buscas que obtiveram respostas no *cache*, enquanto existe um pico em torno de 4,35 ms que representam as buscas que não foram encontradas no *cache* e foram feitas em disco.

De forma semelhante à busca por transação, o tempo de busca por um bloco é calculado utilizando o registro do instante da submissão da busca e o instante em que a resposta é recebida. Para realizar a busca, é utilizado o identificador do bloco que se deseja buscar. O tempo de busca de um bloco é a diferença entre os instantes de tempo de submissão da operação de busca e o retorno da resposta. A Figura 3.4(a) apresenta os resultados de buscas por blocos na cadeia *Parity*. A distribuição de probabilidades apresenta um comportamento periódico entre 8 ms e 15 ms, que não se adéqua a uma distribuição de probabilidades predefinida. Esse comportamento periódico pode ser justificado pelo fato de a *Parity* realizar um processo de sincronização dos nós a cada intervalo de 1 ms, o que gera períodos de atraso no retorno das buscas por blocos na cadeia. No entanto, os resultados apresentam média de respostas em torno de 11 ms, devido ao tempo de acesso



(a) *Parity* (b) *Multichain*

Figura 3.4: Desempenho medido em tempo de busca blocos. O tempo de resposta para a busca por bloco é no mínimo duas vezes mais rápido na *Multichain*. a) Comportamento periódico para busca por blocos indica o armazenamento em memória volátil de blocos recentemente acessados. b) Tempo de acesso a blocos segue uma distribuição logística generalizada.



(a) *Parity* (b) *Multichain*

Figura 3.5: Desempenho medido em tempo de mineração de transações. A latência entre a submissão de uma transação e a efetivação da transação na cadeia é estatisticamente a mesma em ambas as plataformas. a) O tempo de mineração de um bloco segue uma distribuição normal generalizada. b) O tempo de mineração segue a distribuição S_b de Johnson.

dos dados armazenados em disco. A Figura 3.4(b) apresenta os resultados das buscas por blocos na *Multichain*. O tempo de busca por blocos é modelado por uma distribuição logística generalizada, tendo sua maior concentração de resposta em torno de 3,9 ms. As buscas por blocos na cadeia *Multichain* têm a vantagem de ser indexadas na cadeia de duas maneiras. Convencionalmente, o bloco contém o *hash* do bloco anterior em seu cabeçalho. Contudo, a *Multichain* armazena também no cabeçalho do bloco o *hash* do bloco seguinte ao adicioná-lo na cadeia. Esta busca bidirecional faz com que a *Multichain* apresente respostas mais rápidas.

Para calcular o tempo de mineração de transações em cada uma das plataformas é

necessário conhecer o registro de tempo do bloco e o registro de tempo da transação. O registro de tempo do bloco pode ser obtido através da resposta da busca por um bloco e o registro de tempo da transação é armazenado durante a execução da carga de trabalho na etapa inicial da execução da rede de cadeia de blocos. Por fim, o tempo de mineração é obtido através da diferença entre o tempo de submissão de uma transação e o registro de tempo de o bloco ter sido inserido na cadeia. A Figura 3.5 mostra os resultados dos tempos de mineração de transações para as duas plataformas. A Figura 3.5(a) apresenta o histograma dos resultados da cadeia *Parity*, que segue a distribuição de probabilidades normal generalizada, com os tempos de mineração variando entre 0 s e 7 s, com a média em 3,5 s. A Figura 3.5(b) apresenta os resultados da cadeia *Multichain* para o tempo de mineração das transações. Os resultados variam entre 0 s e 8 s[5], seguindo uma distribuição S_b de Johnson, em que o pico de ocorrências é em 1 s. Vale ressaltar que, quanto aos resultados de tempo de mineração, as duas plataformas levam estatisticamente o mesmo tempo para minerar os blocos.

[5]Os valores abaixo de 0 s são erros de precisão devido à sincronização ao se utilizar o servidor NTP e, portanto, não são relevantes para a análise de desempenho.

Capítulo 4

Mecanismo de Consenso Baseado em Confiança para Cadeia de Blocos

Alcançar o consenso em um sistema distribuído é um desafio. Os mecanismos de consenso devem ser resilientes às falhas de nós, particionamento da rede, atrasos de mensagens que chegam fora de ordem e corrompidas. O ambiente colaborativo é suscetível a ataques, e lida com nós egoístas e maliciosos. Entre os ataques mais comuns em redes públicas está o ataque de conluio, ou "ataque de 51%", em que um indivíduo ou instituição excede 51% do poder de mineração, podendo enganar a rede para aceitar transações ilegais ou outras ações maliciosas [12]. Além disso, nós maliciosos podem fazer ataque de eclipse em que um nó em posição estratégica, ou um conjunto de nós em conluio, se organizam para que parte da rede não tenha acesso à cadeia atualizada, filtrando transações e blocos para que os nós sob ataque tenham somente acesso à visão dos atacantes. Se bem-sucedido, o atacante pode mediar a maioria ou toda a comunicação, fazendo como um eclipse que esconde parte da rede [51]. Com a habilidade de minerar a maioria dos blocos em ataques de eclipse e/ou conluio, os mineradores atacantes podem gerar bifurcações de visão sobre cadeia de blocos, gerar transações de gasto duplo ou executar ataques de negação de serviço (DoS) contra endereços ou transações específicas. Em um ataque de bifurcação, o gasto duplo acontece quando um atacante faz com que blocos já confirmados sejam invalidados ao fazer uma bifurcação em um nível abaixo deles, convertendo a visão global para a bifurcação. Por consequência, transações que antes eram consideradas imutáveis são invalidadas. Para o atacante, fazer gasto duplo da própria transação é rentável quando ao invalidar uma transação, recebe um pagamento irreversível ou um produto sem ter que pagar [12]. Por exemplo, se Bob realiza uma compra através de uma transação de transferência de ativos para o vendedor e, após o recebimento do produto ou serviço, o bloco que armazenava a transação for invalidado em uma bifurcação, o ativo usado na

compra volta a ser posse de Bob.

Por outro lado, uma das vantagens das aplicações em redes privadas é permitir a expulsão dos nós que têm comportamento prejudicial à rede. No entanto, após a análise das plataformas privadas permissionadas *Parity* e *Multichain*, observa-se que o mecanismo de consenso Prova de Autoridade não é completamente distribuído. A mineração fica sobre a responsabilidade de um subgrupo de nós de autoridade, mas não existe um monitoramento sobre o comportamento dos mineradores. É uma abordagem ingênua assumir que os nós que receberam a permissão de minerador não estão sujeitos a falha, a invasão ou a agir em benefício próprio. No entanto, nas duas plataformas avaliadas, não foi observado um mecanismo de revogação da permissão ou expulsão de nós de autoridade. Somente um nó administrador da rede pode alterar as permissões, ferindo a natureza descentralizada da rede.

Neste capítulo, é proposto um modelo de confiança para ser associado aos mecanismos de consenso em redes privadas permissionadas. A proposta é que, através de um monitoramento distribuído, os nós mineradores sejam avaliados a partir de suas ações, construindo uma reputação entre os nós da rede. Portanto, para que o nó minerador seja mantido, é preciso que a reputação construída seja superior ao limiar de confiança exigido pela rede. Caso a reputação do minerador esteja abaixo do limiar de confiança, um júri formado por nós da rede vota pela expulsão e, se forem acumulados votos suficientes, o nó malicioso é expulso da rede. Além disso, o modelo propõe um critério para a seleção dos nós mineradores de forma autônoma e que mantém a rede escalável. Propõe-se também a criação de uma cadeia de blocos de controle que seguem os mesmos critérios de confiança associados à cadeia de blocos da aplicação, para o armazenamento das transações do mecanismo de consenso. Desta maneira, os dados de controle não se misturam com os dados da cadeia de blocos da aplicação.

Em redes permissionadas, a distribuição das permissões de mineradores pode seguir métricas diferentes. Esta proposta segue o conceito de maturidade na rede [61]. O critério de maturidade pode ser empregado como critério de confiança, pois os nós necessitam se manter ativamente na rede para serem considerados candidatos ao posto de mineradores. Quanto mais tempo como participante da rede, segundo seu *timestamp* de ingresso, maiores suas chances de se tornar um nó minerador, adquirindo assim a função de gerador de blocos para a cadeia de blocos de aplicação e de controle. Além disso, é também sua responsabilidade autorizar o acesso e conceder permissão de minerador a novos nós, de acordo com a necessidade da rede. Por isso, o modelo proposto proporciona a auto-

organização da rede, para que o crescimento ou a expulsão de nós mineradores não afete o seu funcionamento.

O modelo de confiança segue o critério de monitoramento escalável proposto por Ferraz *et al.*, em que, ao invés de todos os nós avaliarem o comportamento de um determinado nó, um subconjunto de nós é selecionado de maneira pseudoaleatória para ser responsável pelo monitoramento [24]. Nesta proposta, o subconjunto é denominado júri de um nó minerador e é responsável por avaliar a reputação do minerador de acordo com as ações observadas. Cada juiz atribui uma nota inicial ao nó minerador e, de acordo com as ações desempenhadas, esta nota é atualizada. Se a reputação de um minerador estiver abaixo de um limiar de confiança, o júri deve votar e expulsar o nó não confiável.

Sendo assim, o modelo apresenta um mecanismo de controle de acesso distribuído, auto-organizável e escalável e uma análise da confiança para selecionar nós monitoradores, monitorar a reputação e expulsar nós maliciosos. Identificar e classificar a gravidade da ação maliciosa não está no escopo deste trabalho.

4.1 Trabalhos Relacionados

Fernandes *et al.* propõe um mecanismo de autenticação e monitoramento em redes *ad hoc*, chamado *AMORA* (Autenticação e MOnirotamento em Redes *Ad hoc*), que realiza controle de acesso e monitoramento dos nós da rede sem a necessidade de uma entidade centralizadora que autorize e manipule a rede. Esta característica aproxima as aplicações em rede *ad hoc* às aplicações em rede de cadeia de blocos. O *AMORA* apresenta uma autorização de ingresso de novos nós à rede a partir de nós já autorizados, formando uma cadeia de delegação, confiando na rede social entre os usuários da rede para permitir novos integrantes. Além disso, o mecanismo propõe o monitoramento das ações e a detecção de nós maliciosos ou não-cooperativos, para que não permaneçam na rede. No sistema de monitoramento, sempre que um nó da rede observa uma ação maliciosa, este contata os delegados do nó, enviando uma denúncia. Os nós delegados verificam se a denúncia é válida e penalizam a variável de reputação do nó. Se esta alcançar o limiar de expulsão, os nós delegados devem emitir um atestado de expulsão, em que pelo menos um número mínimo de nós deve concordar pela expulsão do nó malicioso, revogando o certificado de permanência na rede [23].

Ferraz *et al.* apresentam um mecanismo de controle de acesso e exclusão baseado em confiança para as redes *ad hoc*. Chamado de TEAM (*Trust-based Exclusion Access-control*

Mechanism), o mecanismo controla o acesso do nó à rede, monitora o comportamento do nó e exclui nós com comportamento inadequado. Através da interação entre os nós, o controle de acesso é obtido por uma combinação de mensagens trocadas entre nós testemunhas e juízes. Usando um modelo de confiança escalável, as testemunhas fazem interações locais para identificar a natureza dos réus, seus vizinhos de um salto. Assim, as testemunhas classificam os réus em níveis de confiança, chamados de reputação, e notificam o júri sobre o comportamento de cada réu. Para cada réu é selecionado aleatoriamente um conjunto de nós na rede para compor o júri. O modelo de confiança local produz informações mais precisas para serem enviadas para os júris e evita sobrecarga de mensagens. Quando o júri recebe a notificação sobre um comportamento malicioso, vota a exclusão desse réu. O mecanismo de votação é importante, porque requer um consenso da maioria do júri, validando a análise do comportamento local em um contexto global [24].

Velloso *et al.* propõe construir um modelo de confiança entre nós em rede *ad hoc* inspirado na confiança humana entre indivíduos, chamado HIT (*Human-Inspired Trust Model*). Cada nó deve atribuir um nível de confiança para outros nós, com base na recomendação de vizinhos confiáveis e suas próprias experiências. O objetivo é tornar os nós capazes de coletar informações, aprender com as experiências e tomar suas próprias decisões. O nível de confiança baseia-se nas experiências anteriores e na recomendação de outros nós da rede. Experiências anteriores permitem que os nós julguem ações executadas por outros nós, que podem levar a três tipos de veredito, uma ação afeta negativamente, positivamente ou não afeta outros nós. Os dois primeiros tipos geram uma atualização de nível de confiança, mas também podem alterar o comportamento do nó. A recomendação de outros nós pode ser levada em consideração, enquanto é calculado o nível de confiança. Para isso, é apresentado o conceito de maturidade, que é baseado na idade do nó em relação aos outros nós da rede. Este conceito permite que os nós deem mais importância às recomendações enviadas por vizinhos mais antigos do que as recomendações enviadas por novos vizinhos [53].

Virendra *et al.* apresentam uma arquitetura de segurança baseada em confiança para redes *ad-hoc* em dispositivos móveis (*Mobile Ad-hoc Networks* - MANETs). Semelhante aos trabalhos apresentados, o objetivo é criar uma métrica de confiança que permita que os nós tomem decisões importantes para a segurança e bom funcionamento da rede. A arquitetura define métricas para avaliar e estabelecer a confiança entre os nós, sendo uma combinação de fatores. A confiança é construída pela observação das ações tomadas pelo nó e pela recomendação, que é a confiança que os nós vizinhos construíram. Além de definir métricas, Virendra *et al.* apresentam um método de avaliação de confiança que

se divide em três fases: inicialização e monitoramento, busca e avaliação, atualização e recrutamento. Estas fases abordam como os nós devem agir, de acordo com a análise da confiança entre os nós para cada momento da arquitetura [54].

Zhu *et al.* apresentam o LHAP (*Lightweight Hop-by-Hop Authentication Protocol For Ad hoc Networks*), um protocolo de autenticação leve para redes *ad hoc*. O LHAP baseia-se em uma técnica que verifica a autenticidade de todos os pacotes transmitidos na rede a partir dos vizinhos de um salto, esta característica se assemelha à validação de uma transação em cadeia de blocos, visto que um nó só encaminha a transação à frente se esta estiver válida e assinada. Para redes móveis *ad hoc*, a assinatura digital a partir de chaves assimétricas pode exigir mais recursos do que os dispositivos se dispõem. Por isso, os autores oferecem um método de assinatura que utiliza um chaveiro unidirecional para autenticação de pacotes e para reduzir a sobrecarga e estabelecer confiança entre nós, sendo este chaveiro uma lista de nós confiáveis que podem assinar o pacote utilizando uma mesma chave [61].

Em redes de cadeia de blocos, Schwartz *et al.* propõe um mecanismo de consenso também baseado em confiança. No *Ripple*, cada nó da rede cria uma lista única de nós confiáveis, chamada de UNL (*Unique Node List*). O consenso sobre o próximo bloco a ser fechado é votado entre os nós da rede, mas o nó somente vota a favor de outros nós da UNL. Como cada nó pode ter UNLs diferentes, os autores mostram que o consenso será alcançado entre todos os nós, independentemente da UNL, sem a ocorrência de bifurcações da cadeia [50]. No entanto, o *Ripple* apresenta problema de escalabilidade. Outro mecanismo de consenso aplicado em cadeia de blocos é o Raft [45], derivado do protocolo Paxos [35]. Nos dois mecanismos o consenso acontece na eleição de um nó líder que tem a responsabilidade de compartilhar decisões para serem votadas coletivamente e, após alcançado a consenso pela votação, o líder deve atualizar a rede sobre o resultado do consenso. Os dois mecanismos apresentam vulnerabilidade, por não considerarem um comportamento malicioso dos nós líderes.

Diferente dos mecanismos apresentados, o mecanismo de consenso baseado em confiança proposto foi desenvolvido para aplicações em cadeias de blocos privadas permissionadas, garantindo um consenso distribuído. Na proposta, os nós mais antigos da rede têm preferência para serem mineradores, pelo critério de maturidade. Os mineradores são monitorados por um júri selecionado aleatoriamente, que avalia a reputação do nó de acordo com o limiar de confiança exigido pela rede. Se o minerador estiver com a reputação abaixo do limiar, é feita a votação pelo júri e a expulsão do nó malicioso. Além disso,

é proposto um mecanismo de controle de acesso que auto organiza a rede em relação ao número de mineradores necessário para manter a rede escalável.

4.2 Controle de Acesso à Rede de Cadeia de Blocos

Os mecanismos de controle de acesso à rede par a par podem variar de acordo com a aplicação da cadeia de blocos. Nesta proposta, o controle de acesso é feito a partir da autorização de pelo menos um nó minerador, que assina autorizando a transação de ingresso do usuário que deseja participar. O usuário solicita o ingresso em um grupo e espera que seja aceito pelo grupo [53].

A Figura 4.1 mostra o mecanismo de controle de acesso proposto. Em (i) o nó deve gerar um par de chaves assimétricas. Essas chaves são utilizadas para assinar as ações do nó na rede e a chave pública do nó é utilizada para endereçá-lo, visto que não existe uma autoridade certificadora que faça a manutenção o domínio dos nós participantes. Por isso, a chave pública do nó deve ser enviada à rede em forma de transação, transação de solicitação de ingresso ($\text{Tx}_{\text{ingresso}}$) como mostra a Figura 4.2(a). Em (ii) a transação precisa ser validada, por um dos nós mineradores que assina a transação, e encaminhada para o repositório de transações válidas. Após a mineração da transação em (iii), a transação faz parte de um bloco da cadeia de controle e todos os nós da rede podem adicioná-lo à lista de chaves de identificação, com o registro de tempo (*timestamp*) da transação de ingresso. Como esse é um nó novo para a rede, ele deve ingressar como um nó comum, com acesso ao conteúdo completo da cadeia e somente permissão de gerar transações. Por isso, nesta proposta, não se atribuem notas de confiança a suas ações.

4.3 Seleção de Nós Mineradores

Para manter a eficiência da rede e não sobrecarregar os nós mineradores, a quantidade de nós mineradores M é diretamente proporcional ao número de nós na rede N. Logo, ao considerar o crescimento da rede, o número de nós mineradores deve crescer proporcionalmente ao crescimento da rede e assim evitar problemas de escalabilidade.

Segundo o critério de maturidade proposto por Velloso *et al* [53], os nós mais antigos na rede, de acordo com o registro de tempo de entrada, têm preferência na seleção dos nós mineradores. Por isso, nesta proposta a informação do número total de nós participantes da rede N é armazenada localmente e atualizada a cada entrada ou expulsão de nós. A

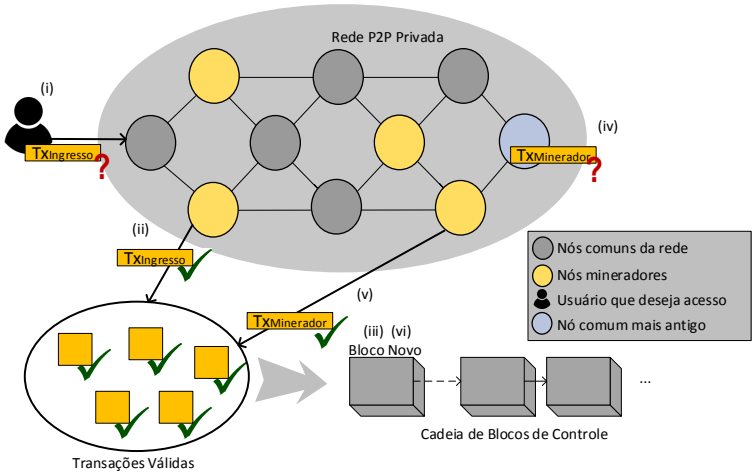
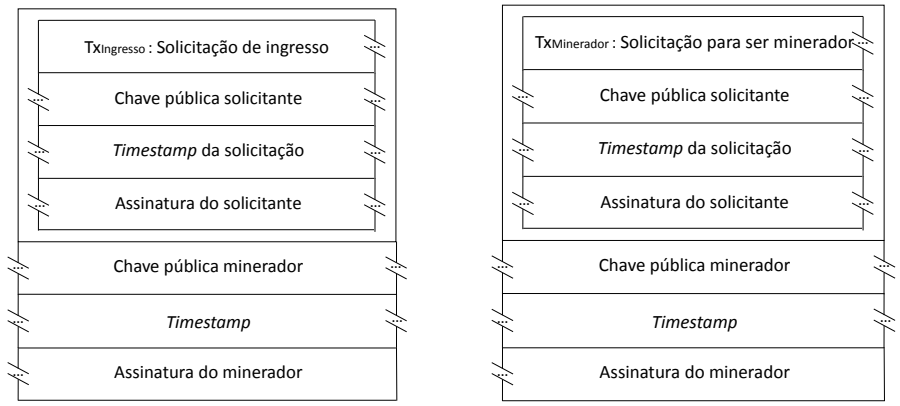


Figura 4.1: Mecanismo de controle de acesso à rede P2P privada, com a entrada de um novo nó minerador. Em (i) o usuário que deseja acesso deve emitir $\text{Tx}_{\text{ingresso}}$ e endereçar a rede. Em (ii) um nó minerador autoriza o acesso assinando $\text{Tx}_{\text{ingresso}}$ e enviando para o repositório de transações válidas. A transação é minerada em (iii) e toda a rede conhece o novo nó. Em (iv) o nó comum mais antigo emite $\text{Tx}_{\text{minerador}}$. Um nó minerador assina e valida a $\text{Tx}_{\text{minerador}}$ em (v). Em (vi) toda a rede conhece o novo nó minerador.

auto seleção dos mineradores acontece quando o conjunto de mineradores precisa de mais nós para manter os padrões de escalabilidade da rede, $M_E \leq M$, em que M_E é o número esperado de mineradores.

Calculando a diferença entre o valor esperado e o atual de nós mineradores, $V = M_E - M$, encontra-se o número de vagas para novos mineradores. O registro do momento em que o nó ingressou na rede possibilita a ordenação das transações de ingresso e permite que todos os nós tenham uma única visão sobre quais são os nós mais antigos. Assim, os αV nós comuns mais antigos devem se prontificar e emitir uma transação Solicitação para se tornar nó minerador ($\text{Tx}_{\text{minerador}}$), evitando assim que nós que não estejam ativos ou que não se prontifiquem a ser mineradores atrapalhem o crescimento escalar da rede. Figura 4.1 no processo (iv) mostra quando um nó se prontifica e emite uma transação tipo $\text{Tx}_{\text{minerador}}$. As transações devem ser validadas de acordo com o *timestamp* para garantir que só os αV nós mais antigos se candidatem, no processo (v) a transação é assinada por um nó minerador, que valida a transação, e encaminhada para o repositório de transações válidas. Em (vi) o minerador que gerar o bloco seguinte tem a tarefa de selecionar as transações $\text{Tx}_{\text{minerador}}$ que foram emitidas pelos nós mais antigos para efetivar quais são os V novos nós mineradores da rede.

A Figura 4.2(a) apresenta as informações que são armazenadas na transação $\text{Tx}_{\text{ingresso}}$.



(a) Transação de solicitação de ingresso

(b) Transação de solicitação para se tornar minerador

Figura 4.2: Transações do mecanismo de controle de acesso. (a) Transação é emitida pelo usuário que deseja entrar e necessita ser validada por um nó minerador. (b) Com a entrada de novos nós, outros nós devem se tornar mineradores e manter a escalabilidade da rede. O nó comum mais antigo emite $\text{Tx}_{\text{minerador}}$.

No primeiro momento, o usuário endereça à rede sua chave pública, *timestamp* da solicitação de ingresso e assinatura do solicitante. Após um dos mineradores da rede conceder o acesso ao novo nó, em um segundo momento, o minerador assina a adiciona a chave pública do nó ingressante, o *timestamp*, a concessão de acesso e assinatura de todo o conteúdo da $\text{Tx}_{\text{ingresso}}$. A Figura 4.2(b) mostra a transação $\text{Tx}_{\text{minerador}}$. Semelhante ao processo de ingresso, a $\text{Tx}_{\text{minerador}}$ é emitida pelo nó antigo, que envia a chave pública, *timestamp* da solicitação de minerador e assinatura do solicitante. Como é de comum conhecimento a necessidade de novos nós mineradores, um minerador deve validar a $\text{Tx}_{\text{minerador}}$ adicionando a chave pública do minerador, *timestamp* da concessão da permissão de minerador e assinatura de todo o conteúdo. Uma vez em que a transação faz parte de um bloco minerado da cadeia de blocos de controle, todos os nós concordam sobre o novo nó minerador e este passa a ser monitorado por nós juízes que irão atribuir notas a todas ações e comportamentos desempenhados na rede.

4.4 Seleção dos Juízes

No mecanismo proposto, todos os nós participantes da rede são monitorados por outros, chamados de nós juízes. Cada minerador deve receber um número J_E de juízes, responsáveis por monitorar o comportamento e julgar a confiança do nó a cada ação desempenhada. O número de juízes necessários para um bom funcionamento da rede de-

pende da resiliência esperada, considerando fatores como a quantidade de nós maliciosos, tempo esperado para a exclusão de nó malicioso e a acurácia dos nós de juízes em relação à confiança depositada sobre o nó réu.

Um nó não pode ser capaz de identificar previamente seus juízes, por isso é escolhido um algoritmo de seleção pseudoaleatório não reversível. Baseado na construção implícita de um Filtro de *Bloom*, aplicado sobre as chaves públicas do nó minerador, os nós que são determinados pelo filtro assumem a posição de juízes. Esse filtro permite estimar um valor esperado de juízes, alterando o tamanho do filtro e a quantidade de interações necessárias para atingir um número médio de juízes.

4.4.1 Filtro de *Bloom* Aplicado à Chave Pública do Minerador

O Filtro de *Bloom* [6] é uma estrutura de dados usada para representar a pertinência de elementos a um conjunto $S = \{s1, s2, ..., sn\}$ de n elementos. Ele é constituído, de maneira genérica, por um vetor de m bits e por k funções *hash* independentes $h_1, h_2, ..., h_k$ cujas saídas variam uniformemente no espaço discreto $\{0, 1, ..., m-1\}$.

Este vetor m é inicialmente composto por todos os bits iguais a zero. Sobre os elementos são aplicadas k funções *hash*. Para cada elemento $s_i \in S$, os bits do vetor correspondente às posições $h_1(s_i), h_2(s_i), ..., h_k(s_i)$ são preenchidos com 1, onde a mesma posição do bit pode ser preenchida mais de uma vez. Uma vez que o Filtro de *Bloom* é uma forma compacta de representar um conjunto de elementos, testes de pertinência podem ser realizados visando a determinar se um elemento x pertence ou não ao conjunto S. Para isso, é necessário verificar se os bits do vetor correspondente às posições $h_1(x), h_2(x), ..., h_k(x)$ estão preenchidos com 1. Se pelo menos um bit estiver zerado, então com certeza $x \notin S$. Por outro lado, se todos os bits estão preenchidos, assume-se que $x \in S$. Um elemento externo $x \notin S$ pode ser reconhecido como um autêntico elemento do conjunto, criando um falso positivo. Tal anomalia ocorre quando todos os bits $h_1(x), h_2(x), ..., h_k(x)$ são preenchidos por outros elementos de S inseridos no filtro [36].

Dado que são usadas funções *hash* perfeitamente independentes e aleatórias, a probabilidade p de um bit se manter zero depois da inserção de n elementos é

$$p = (1 - \frac{1}{m})^{kn} \approx e^{\frac{-kn}{m}}, \tag{4.1}$$

em que n é igual ao número de elementos contidos no filtro, k é o número de interações *hash* sobre o elemento e m o comprimento do filtro em bits.

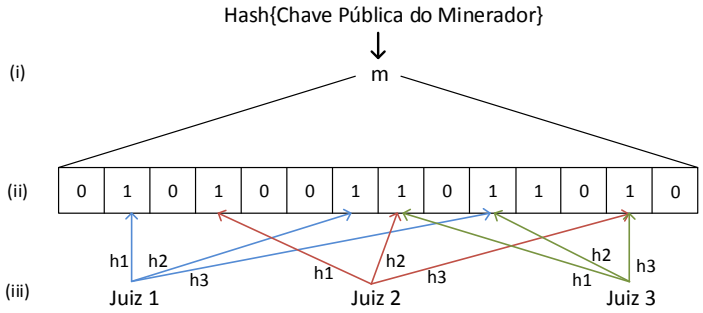


Figura 4.3: Esquema de representação do filtro de *Bloom* a partir do *hash* da chave pública do minerador. (i) Algoritmo *hash* sobre a chave pública do minerador. (ii) Os m primeiros bits resultantes do algoritmo *hash* preenchem o filtro. (iii) O nós da rede checam a pertinência no filtro de *Bloom* do novo minerador.

Na proposta, o *hash* da chave pública do minerador é interpretado como o Filtro de *Bloom*, em que o vetor de m bits é representado pela seleção dos m primeiros bits do *hash*. Sendo a função *hash* perfeitamente aleatória, a probabilidade de ocorrências de 0's e 1's dentro da seleção dos m primeiros bits é $p = 0,5$. Ao considerar a criação do filtro de maneira implícita, pela seleção aleatória da chave pública, é observado um valor de juízes esperados J_E para cada minerador. Com o intuito de se empenhar menos processamento em k diferentes interações de *hash*, determina-se o valor de k e o número de elementos $n = J_E$ que pertencem ao conjunto na equação 4.2, aplicando a probabilidade $p = 0,5$ na equação 4.1 e calcula-se o comprimento do filtro pela equação 4.3.

$$m = -\frac{n * k}{ln(p)} \tag{4.2}$$

$$m = -\frac{n * k}{ln(0,5)} \approx \frac{n * k}{0,693} \tag{4.3}$$

A Figura 4.3 mostra a criação do filtro realizada por todos os nós da rede, seguindo três etapas. A partir do *hash* da chave pública do minerador na etapa (i), o valor de m delimita o comprimento do filtro a partir dos m primeiros bits resultantes da função *hash* na etapa (ii) . Sendo assim, o filtro de pertinência dos nós juízes deve ser criado de maneira implícita, na qual não se aplica o critério de falso positivo. Os nós pertencentes ao conjunto de juízes desse minerador são desconhecidos, quando a chave pública do minerador for criada de maneira aleatória. Com a formação do filtro, é responsabilidade

de todos os nós da rede verificar a sua pertinência na etapa (iv), em que o nó executa k diferentes interações *hash* sobre a própria chave pública para verificar sua pertinência ao filtro. Os resultados das interações representam as posições em que o vetor do filtro devem ter bits em 1, se em pelo menos umas das posições o bit for 0, o nó não pertence ao conjunto de juízes do minerador desse filtro. Logo, se as respectivas posições do filtro estiverem com 1's, o nó pertence ao filtro e é juiz do minerador. As transações que apresentam filtros que não seguem o critério de aleatoriedade dos bits, com probabilidade em torno de 0,5 entre 0's e 1's, serão consideradas inválidas e descartadas da rede, i.e., quando o número de 0's é muito maior que o de 1's no vetor m.

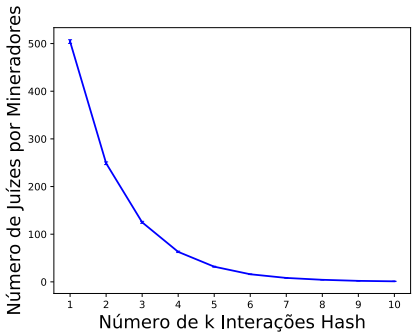
O valor esperado de juízes que o nó minerador recebe segundo o tamanho da rede, é calculado pelo valor esperado de elementos que estão contidos no filtro, de acordo com a probabilidade de os bits, nas k posições definidas pelas funções *hash*, serem 1. Pela Equação 4.1, p é a probabilidade de um bit se manter 0 após k interações e n inserções. Logo, $(1 - p)$ é a probabilidade de um bit se manter 1 após k interações e n inserções. Na verificação de se um elemento pertence ao filtro, as k posições definidas pelas funções *hash* devem estar preenchidas com 1's no vetor de bits do filtro. Então, a probabilidade de um elemento pertencer ao filtro é igual a $(1-p)^k$, já que o preenchimento de cada bit no vetor é um processo independente. Logo, para calcular o valor esperado de juízes que um nó tem na rede, $\hat{J}_E$, para N nós na rede, tem-se que:

$$|E(J_E)| = \hat{J}_E = \sum_{i=1}^{N}(1-p)^k = N(1-p)^k \tag{4.4}$$

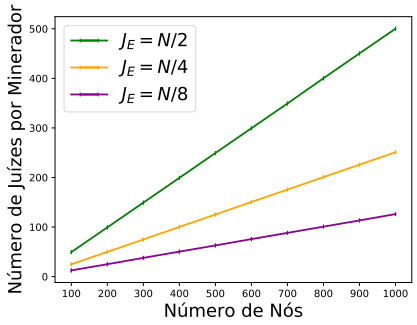
Como o filtro é formado implicitamente pela função *hash* de probabilidade $p = 0,5$, o valor medido de J_E decresceu exponencialmente, conforme há o aumento do número de funções *hash*, k, como mostra a Figura 4.4(b), em uma simulação em que $N = 1000$. A Figura 4.4(a) apresenta o resultado da simulação do teste do filtro de *Bloom*, para redes com 100 a 1000 nós, para a seleção dos juízes por nó minerador. Os valores reais de juízes são, em média, o valor de juízes esperados J_E com intervalo de confiança de 95%, com valores de, no máximo, $0,15$ em torno da média.

4.5 Monitoramento dos Nós Mineradores

Após a seleção dos juízes pelo Filtro de *Bloom*, se o nó pertencer ao conjunto de juízes J_E do nó minerador, esse adiciona a chave pública do minerador na tabela de confiança



(a) Número médio de juízes pertencentes ao filtro de acordo como k funções de *hash* em uma rede de 1000 nós.



(b) Número médio de juízes pertencentes ao filtro de acordo com o crescimento da rede.

Figura 4.4: Número médio esperado de juízes pertencentes ao filtro de *Bloom*.(a) O gráfico mostra o resultado esperado de juízes ao aumentar o número de k interações em uma rede de 1000 nós, em que $p = 0,5$. A curva representa os resultados simulados e calculados a partir do Equação 4.4, com intervalo de confinação não expressivo. (b) O gráfico mostra que o número médio de juízes é igual ao número esperado de juízes, de acordo com o crescimento de N nós, simulados com $k = 1$ para $J_E = N/2$, $k = 2$ para $J_E = N/4$ e $k = 3$ para $J_E = N/8$.

e associa uma Nota Inicial de reputação (NI). Em uma abordagem inocente, um novo nó minerador pode receber, inicialmente, uma nota alta de reputação ($NI > L$). Já em uma abordagem desconfiada, os nós não esperam que os novos sejam confiáveis, recebendo assim uma nota inicial de reputação no limiar de confiança ($NI = L$). O nó minerador deve ter a nota de reputação sempre superior ou igual ao limiar de confiança, $R \geq L$, para se manter participante da rede. Logo, os nós juízes devem monitorar os blocos gerados pelos seus réus e julgar se houve alguma ação maliciosa.

Os nós mineradores são responsáveis por gerar blocos tanto na cadeia de blocos de dados, quanto na cadeia de blocos de controle. Os seus juízes devem analisar o comportamento do minerador nas duas cadeias e associar uma nota a cada ação desempenhada [23, 24, 53]. Se o minerador gera um bloco corretamente, seus juízes devem incrementar a nota de reputação $R^i = R^{i-1} + Up$. Se for observada maliciosidade na geração do bloco, seus juízes devem decrementar a nota de reputação $R^i = R^{i-1} - Up$. R pode variar entre 0 e 10, e Up é um valor de atualização que pode variar de acordo com o impacto da ação realizada na cadeia.

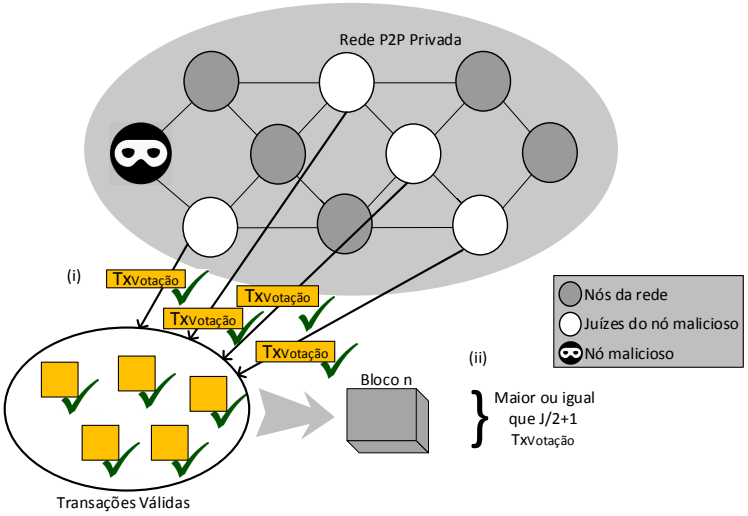


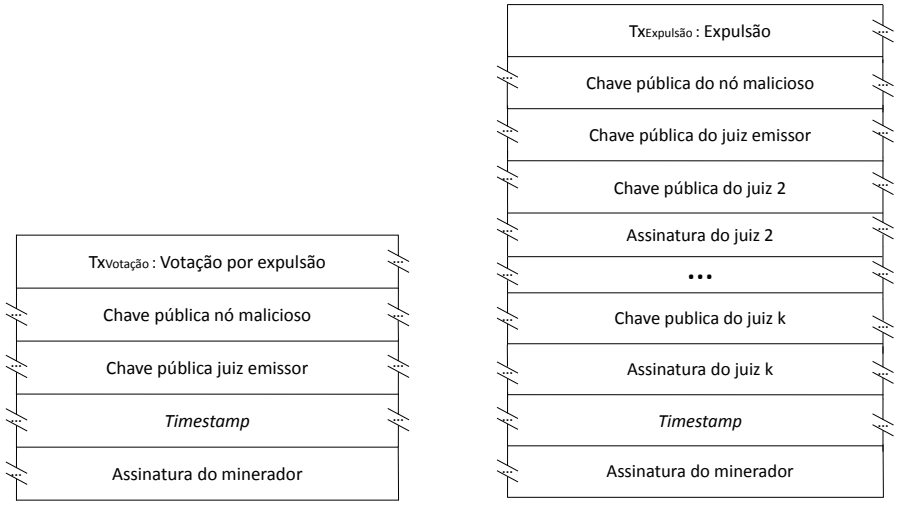
Figura 4.5: Esquema de votação pela expulsão do nó malicioso. Nós juízes emitem transações $\text{Tx}_{\text{votação}}$ pois o nó julgado tem sua reputação menor do que o limiar de confiança. A votação pela expulsão termina quando são emitidas $\frac{J_E}{2} + 1$ $\text{Tx}_{\text{votação}}$

4.6 Expulsão dos Nós Maliciosos

O mecanismo de expulsão de um nó malicioso age em duas etapas: votação e expulsão. A etapa de votação acontece pela natureza descentralizada da rede, cada nó juiz de um minerador terá a nota de reputação R calculada e armazenada localmente. Por isso, é necessário que pelo menos mais da metade do número esperado de juízes J_E concordem sobre a reputação do minerador estar abaixo do limiar de confiança. A etapa de expulsão ocorrerá caso o nó minerador receba $\frac{J_E}{2} + 1$ $\text{Tx}_{\text{votação}}$ na etapa de votação Neste caso a expulsão do nó malicioso é concretizada.

Ao atualizarem as notas de reputação do minerador durante a votação, se os juízes observarem que $R \leq L$, onde L é o valor de limiar de confiança, emitirão a transação de Votação por expulsão, como mostra a Figura 4.6(a). A transação contém a chave pública do possível nó minerador malicioso, a chave pública do juiz, o *timestamp* e a assinatura do juiz sobre todo o conteúdo da transação. Para que a votação converta-se em expulsão do nó malicioso, é preciso que sejam emitidas pelo menos $\frac{J_E}{2} + 1$ de transações de votação. Na Figura 4.5, em (i) os nós juízes do réu malicioso emitem a transação $\text{Tx}_{\text{votação}}$, em (ii) as transações são validadas, e após mineradas são contabilizados os votos pela expulsão do nó.

Na expulsão, após a mineração das transações é possível determinar qual foi a tran-



(a) Transação de votação por expulsão de nó malicioso

(b) Transação de expulsão de nó malicioso

Figura 4.6: Transações do mecanismo de exclusão de nós maliciosos. (a) Quando a nota de reputação é menor que o limiar de confiança, os juízes do minerador emitem a transação de votação. (b) Se a votação receber mais que $J_E/2 + 1$ transações, o juiz que emitiu a primeira $\text{Tx}_{\text{votação}}$ emite a $\text{Tx}_{\text{expulsão}}$.

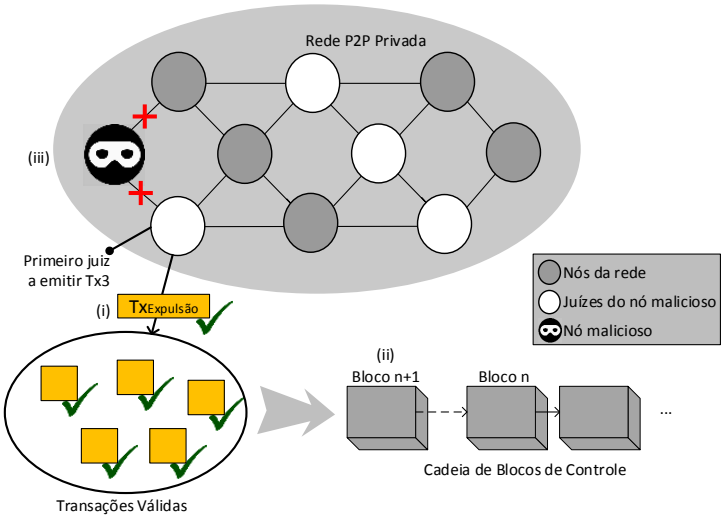


Figura 4.7: Expulsão do nó malicioso. Após a votação acumular $\frac{J_E}{2} + 1$ $\text{Tx}_{\text{votação}}$, o nó juiz que emitiu a primeira transação de votação emite a transação de expulsão $\text{Tx}_{\text{expulsão}}$. Após minerada, toda a rede exclui a chave do nó malicioso.

sação de votação por expulsão ($\text{Tx}_{\text{votação}}$) com o *timestamp* mais antigo. Após o tempo de votação, o nó que gerou a $\text{Tx}_{\text{votação}}$ mais antiga, emite a $\text{Tx}_{\text{expulsão}}$, caso este não gere

a transação, o nó que gerou a segunda mais antiga assume a geração da transação de expulsão. A Figura 4.6(b) mostra a transação $\text{Tx}_{\text{expulsão}}$ que contém a chave pública do nó malicioso, a chave pública do juiz emissor da transação , o *timestamp* e as $\frac{J_E}{2} + 1$ assinaturas coletadas na votação pela expulsão e a assinatura do juiz emissor sobre todo o conteúdo da transação . Na Figura 4.7, em (i) um dos juízes emite a transação $\text{Tx}_{\text{expulsão}}$, que deve ser validada. Após minerada em (ii), todos os nós da rede são informados da expulsão do nó malicioso e em (iii) os nós apagam a chave pública do nó expulso.

Todo o mecanismo de expulsão pode ser validado verificando-se a real pertinência dos nós juízes ao Filtro de *Bloom* do minerador. Transações de expulsão só são mineradas em um bloco após a validação das assinaturas dos juízes. Após a mineração, todos os nós atualizam a visão da rede em relação aos números de nós, mineradores ou não, e retiram o endereçamento do nó expulso, descartando sua chave pública.

Capítulo 5

Avaliação do Mecanismo de Consenso Baseado em Confiança*

A avaliação do mecanismo de consenso baseado em confiança proposto é realizada através de simulações. Para tanto, desenvolveu-se um simulador de eventos discretos para redes de cadeia de blocos privadas permissionadas[1]. O simulador foi escrito em Python 3 e todas as funções de criptografia são implantadas com biblioteca PyCrypto[2]. A rede de cadeia de blocos do simulador é semelhante à rede implementada com a plataforma *Multichain* no Capítulo 3. A plataforma *Multichain* permite que um usuário administrador forneça permissões para os novos nós na rede, como leitura, gravação e mineração. O mecanismo de consenso da *Multichain* é a Prova de Autoridade [26], nele os nós com autoridade de mineração se revezam na tarefa de minerar blocos. Assim, o *Multichain* é utilizado como base para desenvolvimento do simulador, por apresentar bons resultados na avaliação de desempenho entre as plataformas no Capítulo 3 e apresentar um mecanismo de consenso barato, mas que não avalia a confiança dos nós mineradores.

5.1 Modelo do Atacante

Esta proposta considera o modelo de atacante de Dolev *et al.*, que afirma que o atacante é capaz de ler, enviar e descartar transações endereçadas à cadeia de blocos, ou qualquer pacote da rede [19]. O atacante pode agir passivamente, conectando-se à rede e capturando todas as trocas de mensagens, ou ativamente, injetando, repetindo, filtrando

*Agradecimento pela colaboração de Lúcio H. A. Reis

[1]O simulador está disponível em https://github.com/marcelatuler/Private-Blockchain-Proof-of-Trust.git.

[2]Disponível em https://pypi.org/project/pycrypto/.

ou trocando informações [1].

Os ataques em cadeia de blocos são tentativas de impedir que uma transação ou bloco legítimo seja incorporado à cadeia. Para que um ataque seja bem-sucedido, o invasor deve controlar uma parte significativa da rede, a ponto de afetar o mecanismo de consenso. O ataque conhecido como Ataque de 51% exige que um usuário, ou um grupo que tenha interesse no ataque, tenha controle de, pelo menos, 51% do poder de mineração da rede [12]. Este ataque é mitigado pelo modelo de confiança proposto, pois, para se manter como nó minerador, é necessário ter controle também sobre mais da metade do número esperado de juízes de cada um dos nó. Dado que os juízes são escolhidos aleatoriamente, o ataque exigiria o controle de quase a totalidade dos nós da rede.

Ataques que tentam modificar ou corromper uma transação não são possíveis, porque cada transação é acompanhada por um *hash* correspondente assinado. Contudo, o ataque do Minerador Egoísta permite que transações sejam ignoradas por algum tempo [22]. O minerador malicioso, neste caso, seleciona quais transações serão mineradas por ele, ignorando propositalmente alguma transação. No entanto, no mecanismo PoA existe uma rotatividade entre os mineradores, o ataque deve durar por uma rodada. Além disso, uma vez identificada a mineração egoísta, com o modelo de confiança os nós podem votar pela expulsão do nó malicioso.

Além de ignorar transações pelo ataque egoísta, no Ataque de Eclipse, os ataques de rede representam a tentativa de isolar um único nó ou um grupo de nós da rede, impedindo assim que a rede execute transações ou leia e atualize o conteúdo da cadeia de blocos [51]. Essa categoria de ataque contempla ataques clássicos de rede que podem ser mitigados através do estabelecimento de caminhos redundantes entre os nós distribuídos. O modelo proposto assume que todos os participantes estão interconectados por uma rede de malha completa redundante. Contudo, a mitigação completa de ataques de rede está fora do escopo deste trabalho.

Em redes de criptomoedas, o Ataque de gasto duplo acontece quando o atacante realiza duas transações, praticamente consecutivas, com dois nós diferentes da rede. Por exemplo, duas transferências de pagamento com quantias de moedas que ultrapassam o valor que o atacante contém em carteira, mas que no momento das transferências foram validadas. Este tipo de ataque pode ser mitigado, se as vendas só forem efetivadas depois que uma quantidade mínima de blocos for minerada após o bloco que armazenou a transferência monetária. O encadeamento de *hash* e a cadeia distribuída tornam improvável invalidar esta transação. O mecanismos proposto, por não ser baseado em competição, evita a

criação de bifurcações na cadeia. Assim, ataques de gasto duplo não acontecem.

Em mecanismos de consenso sem competição, o minerador que se ausenta na geração de bloco, ou não participa ativamente das funções na rede, pode ser visto com um nó malicioso, pois prejudica o funcionamento de aplicações de baixa latência. Dessa forma, a abstenção de um nó minerador pode ser classificado como uma ação maliciosa e tratada pelo mecanismo proposto. Além disso, existem ataques que consistem em tentar obter informações de configuração ou personificação do alvo. Ataques de personificação não são possíveis, porque todas as transações enviadas na rede são assinadas pelo emissor. Os ataques que buscam obter informações de configuração são atenuados pela criptografia de informações confidenciais, em que o invasor precisa obter a chave privada do destinatário alvo. Este trabalho não aborda o caso em que um nó é comprometido por meio da invasão de terminal ou sequestro de chave. Além disso, o modelo proposto permite a auditoria de todas as transações passadas. Portanto, se um invasor tentar modificar a cadeia de blocos usando pares de chaves roubados, a tentativa será registrada [1]. Após a descoberta de um atacante, os pares de chaves roubados podem ser facilmente excluídos da rede, restabelecendo a segurança e evitando mais danos.

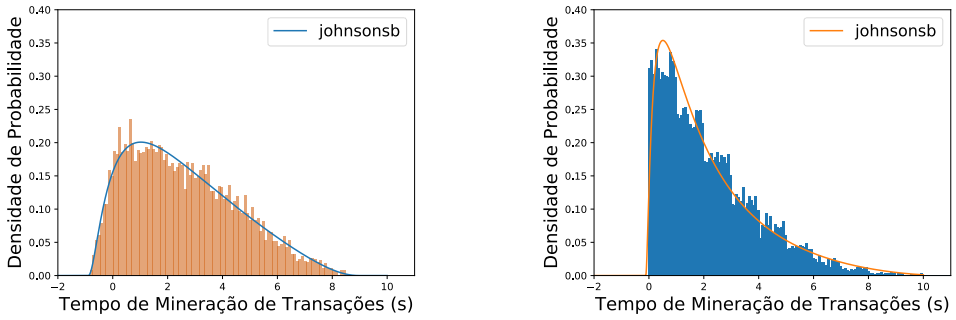
5.2 Validação do Simulador

O simulador baseia-se no comportamento observado na plataforma *Multichain* para as métricas de tempo de validação de uma transação e mineração de um bloco, simulando a implementação real da rede *Multichain*. A simulação discreta no tempo em cada passo de execução que representa 1 ms de execução real. Assim, é possível comparar o tempo de execução da *Multichain* com a estimativa de tempo discreta da simulação. A rede par a par simulada é composta por dez nós, para simular a rede implementada na plataforma *Multichain*, todos com permissão para gerar transações, participar do consenso e minerar blocos. A rede é formada com as mesmas características de configuração da rede *Multichain* implementada no Capítulo 3. Então, para validar o simulador desenvolvido, é aplicada a mesma distribuição do tempo entre chegada de transações observada na rede *Bitcoin* no período entre junho de 2017 e junho de 2018. Os dez nós da rede esperam um tempo dentro da Função de Distribuição Probabilidade (FDP) normal generalizada com os parâmetros $\mu = 0.371$, $\alpha = 0.143$ e $\beta = 2.786$, para emitir uma nova transação, que varia probabilisticamente entre 100 e 700 ms, com média de 371 ms.

A métrica do tempo de validação de transações observada no *Multichain*, que repre-

senta o intervalo de tempo desde a emissão até a validação da transação, seguiu a FDP log-logarítmica generalizada de Burr, com os parâmetros $\mu = 9.798$, $a = 19.208$, $b = 1.048$ e $c = -0.156$. O tempo de validação varia probabilisticamente entre 6 e 13 ms, com média em $9,798$ ms. Outra métrica utilizada para o configurar o simulador é o tempo de geração de blocos do *Multichain*, que representa o intervalo entre o *timestamp* dos blocos. Essa métrica seguiu a FDP gamma, com os parâmetros $\mu = 5.462$, $k = 2.000$ e $\theta = 0.825$. O tempo de geração de um bloco varia probabilisticamente entre 0 e 12 s, com média em $5,462$ s.

Os resultados mostram que o tempo para minerar uma transação na *Multichain* e no simulador seguem a mesma FDP, S_b de Johnson, como mostrado nas Figuras 5.1(a) e 5.1(b). Os parâmetros das duas distribuições diferem devido à imprecisão na medição do tempo na implementação do *Multichain*, devido à sincronização ao utilizar o servidor NTP, o que implica tempos negativos na Figura 5.1(a). A simulação discreta no tempo não apresenta essa imprecisão, mostrando maior concentração de ocorrências do histograma em trono da média da FDP de 1.474 s. Esse comportamento não pode ser observado, pois a FDP S_b de Johnson do *Multichain* foi suavizada para englobar os valores negativos consequentes da imprecisão, o que justifica a diferença dos parâmetros das FPDs e valida o simulador.



(a) Tempo de mineração de transação na *Multichain*

(b) Tempo de mineração de transações no simulador

Figura 5.1: Tempo de mineração de transações no *Multichain* e no simulador. a) Distribuição do tempo entre a validação e a mineração de transações na rede *Multichain* (a) e na rede simulada (b) seguem a mesma distribuição Sb de Jonhson, para redes de um mesmo tamanho.

5.3 Simulação do Controle de Acesso

A simulação do mecanismo proposto de controle de acesso utiliza o mesmo simulador validado. Inicialmente, com uma rede simulada de dez nós, que recebe transação de solicitação de ingresso ($\text{Tx}_{\text{ingresso}}$) segundo uma distribuição uniforme em um intervalo entre 1 ms e 1 h. A chegada de transações simula o interesse de novos usuários que desejam acesso à rede par a par. Para que o usuário emissor da transação $\text{Tx}_{\text{ingresso}}$ possa ter acesso à rede, é preciso que a transação seja assinada por pelo menos um nó minerador. Sendo assim, a assinatura do minerador representa a validação da $\text{Tx}_{\text{ingresso}}$.

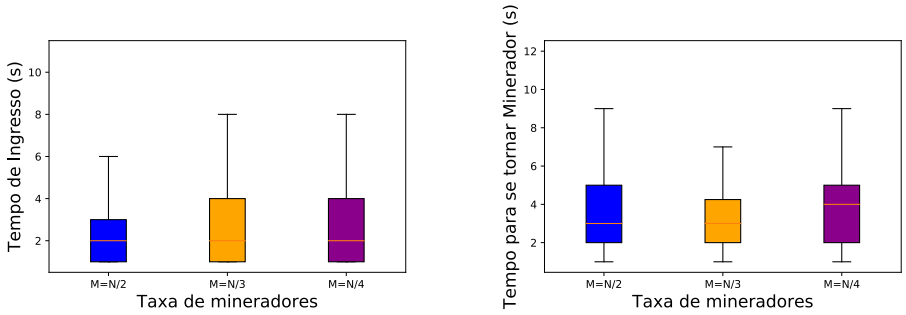
Após validadas, as $\text{Tx}_{\text{ingresso}}$ são mineradas em blocos e fazem parte efetiva da cadeia de controle. Assim, todos os nós da rede atualizam a visão sobre a quantidade de nós na rede N, o que pode desencadear a necessidade da atualização do número de mineradores (M), sendo M uma porção dos N nós da rede, que cresce linearmente de acordo com o crescimento da rede. Caso exista a oportunidade para um novo nó minerador, o nó que tiver o tempo mais longo na rede, de acordo com o *timestamp* de ingresso, emitirá a transação de solicitação para ser minerador ($\text{Tx}_{\text{minerador}}$), que é validada e minerada seguindo as mesmas métricas de tempo da rede simulada.

Com o objetivo de avaliar a interferência do número de nós mineradores (M) na eficiência da rede, são montados três cenários de rede, em que M varia entre $M = \frac{N}{2}$, $M = \frac{N}{3}$ e $M = \frac{N}{4}$. Os cenários de rede, inicialmente com $N = 10$, são simulados durante 1 horas e as métricas de avaliação são o tempo médio para o ingresso de um novo nó, o tempo médio para nó se tornar minerador e as cargas em número de transações e bytes geradas pelo mecanismo de controle de acesso.

A métrica de tempo de ingresso de um novo nó é medida pela diferença entre o *timestamp* do bloco que armazena $\text{Tx}_{\text{ingresso}}$ e o *timestamp* de emissão de $\text{Tx}_{\text{ingresso}}$. A Figura 5.2(a) mostra o tempo médio para que um usuário se torne um nó da rede, para os três cenários simulados. Com a análise dos resultados, o cenário de $M = \frac{N}{2}$ apresenta o menor tempo probabilístico para o acesso de um novo nó, que varia entre 1 e 6 s. Enquanto no cenário da rede com $M = \frac{N}{3}$ e $M = \frac{N}{4}$ o tempo de acesso varia entre 1 e 8 s. Por mais que a diferença entre os resultados de tempo de ingresso para rede com 100 nós seja sutil, entende-se que a seleção do valor de M tem maior impacto no controle de acesso com o crescimento da rede.

A métrica de tempo para um nó se tornar nó minerador é medida pela diferença entre o *timestamp* do bloco que armazena $\text{Tx}_{\text{minerador}}$ e o *timestamp* de emissão de $\text{Tx}_{\text{minerador}}$. A

Figura 5.2(b) mostra o tempo médio para um nó se tornar minerador, para os três cenários simulados. A partir da análise dos resultados, observa-se que a diferença de tempo é de menos de 1% entre os três cenários. Isso pode ser justificado, pois o número de $\text{Tx}_{\text{minerador}}$ é inferior ao número de $\text{Tx}_{\text{ingresso}}$ nos repositórios de transações válidas, convergindo os resultados para valores similares aos medidos para o tempo de ingresso.



(a) Tempo médio para ingresso de um novo nó.

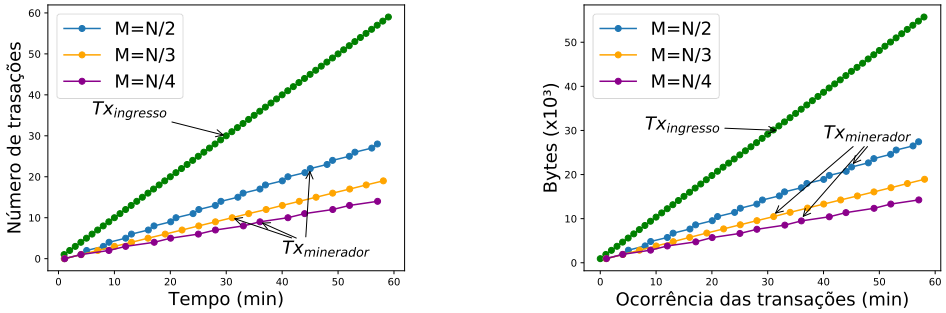
(b) Tempo médio para se tornar um nó minerador.

Figura 5.2: Tempo médio para ingresso e se tornar minerador para diferentes proporções de mineradores. Na rede com $M = \frac{N}{2}$, na rede com $M = \frac{N}{3}$ e na rede com $M = \frac{N}{4}$, (a) tempo desde a emissão até a mineração de $\text{Tx}_{\text{ingresso}}$ e (b) tempo desde a emissão até a mineração de $\text{Tx}_{\text{minerador}}$

A última métrica de avaliação dos cenários é a carga gerada pelo mecanismo de controle de acesso, em número de transações e em bytes, armazenados na cadeia de blocos de controle. O resultados estão representados na Figura 5.3. As cargas em número de transações são demonstradas na Figura 5.3(a) e em quantidade de bytes de controle gerados na Figura 5.3(b). Os resultados apresentam as cargas geradas $\text{Tx}_{\text{ingresso}}$, que acontecem igualmente nos três cenários, e as cargas geradas em cada um dos cenários para as transações $\text{Tx}_{\text{minerador}}$. O crescimento das transações de minerador é relacionado ao número desejado de mineradores da rede. Logo, o crescimento da carga gerada por $\text{Tx}_{\text{minerador}}$ é linear ao crescimento da rede, seguindo a proporção do número de mineradores esperados. Portanto, em uma hora de simulação, a cadeia de blocos de controle acumula $27,434\text{x}10^3$ bytes, na rede de $M = \frac{N}{2}$, enquanto na rede de $M = \frac{N}{3}$ acumula $18,932\text{x}10^3$ bytes e na rede de $M = \frac{N}{4}$ acumula $14,250\text{x}10^3$. Com isso, carga em bytes gerada no cenário de $M = \frac{N}{2}$ é duas vezes superior à carga em $M = \frac{N}{4}$ e aproximadamente $1,5$ vezes superior em $M = \frac{N}{3}$.

Considerando os resultados alcançados em tempo de ingresso, o cenário que tem metade da rede com permissão de minerador $M = \frac{N}{2}$ apresenta os menores tempos. Esta rapidez em relação aos outros cenários é justificada pela maior divisão nas tarefas de

mineração e validação das $\text{Tx}_{\text{ingresso}}$ e $\text{Tx}_{\text{minerador}}$ com mais nós mineradores. Contudo, para o acumulo de cargas com as transações utilizadas para o controle de acesso à rede, é observado que $M = \frac{N}{4}$ obtém o menor acumulo de bytes na cadeia. Por isso, visando a uma arquitetura de rede que mostre maior flexibilidade e resultados intermediários para as duas métricas, o cenário $M = \frac{N}{3}$, que demonstra ser uma solução de compromisso entre tempo e carga gerada, será utilizado para as próximas simulações.



(a) Crescimento linear do número de transações (b) Crescimento linear da carga em bytes

Figura 5.3: A análise do crescimento em número de transações $\text{Tx}_{\text{ingresso}}$ e $\text{Tx}_{\text{minerador}}$ e da carga gerada em bytes para os três diferentes cenários.

5.4 Monitoramento e Expulsão de Mineradores Maliciosos

Nesta simulação, a rede inicia com $N = 100$ nós participantes, sendo o número de mineradores $M = \frac{N}{3}$ do total de nós na rede, resultando em um total de $M = 33$ nós mineradores. Dentro do grupo de nós mineradores, são selecionados $M_{maliciosos} = \frac{M}{3}$ para realizar ações maliciosas, totalizando $M_{maliciosos} = 11$. Os nós maliciosos têm a oportunidade de agir a cada rodada de mineração e na simulação é atribuída uma probabilidade $0,8$ de executar uma ação maliciosa a cada rodada.

Parâmetros como limiar de confiança (L), nota inicial de reputação (NI) e número de juízes esperado (J_E), devem ser escolhidos, para que a rede possa detectar nós maliciosos e expulsá-los com maior eficiência. Por isso, são simulados 16 cenários de rede, alternando os parâmetros L, NI e J_E a fim de identificar qual configuração fornece melhores resultados. Todos os cenários são simulados 30 vezes, com duração de uma hora, em tempo discreto.

O número de juízes varia em casos em que todos os nós da rede são juízes de um minerador ($J_E = N$), metade dos nós da rede são juízes de um minerador ($J_E = \frac{N}{2}$),

Cenários				
L	6	7	8	6
NI	8	9	10	6

Tabela 5.1: Combinação de valores de limiar de confiança e notas iniciais de reputação utilizada nos cenários de rede simulados.

um quarto dos nós da rede são juízes de um minerador ($J_E = \frac{N}{4}$) e um oitavo dos nós da rede são juízes de um minerador ($J_E = \frac{N}{8}$). Para cada uma das configurações de juízes, são configuradas L e NI, como mostra a tabela 5.1.

A avaliação dos parâmetros é feita a partir das métricas de tempo de comportamento malicioso, tempo de votação para a expulsão de um nó malicioso e carga gerada em bytes.

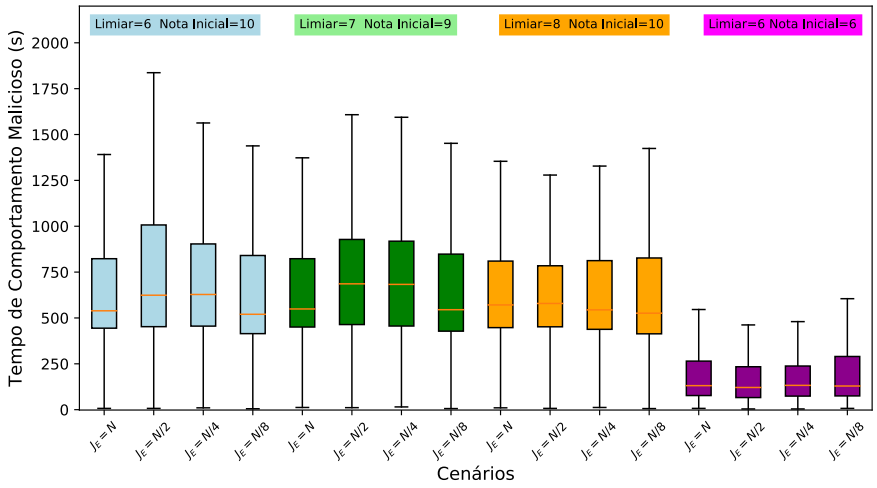


Figura 5.4: Tempo de comportamento malicioso na rede, desde a primeira ação até a expulsão da rede.

A Figura 5.4 apresenta a média e o intervalo de confiança de tempo em que um nó malicioso agiu na rede, desde sua primeira ação maliciosa até a sua expulsão. Em relação as notas de limiar e nota inicial, o cenário em que o nó minerador recebeu a nota inicial de confiança igual a nota de limiar é o que tem o menor tempo de ações maliciosas. Neste cenário, a permanência de um nó malicioso é três vezes menor em relação aos outros cenários, enquanto os outros cenários não apresentam diferença significativa em tempo. No entanto, ao alterar os cenários em número de J_E, os resultados não apresentam diferença no tempo de percepção de nós maliciosos e expulsão desses nós.

Na análise do tempo de votação para expulsão de um nó malicioso, a Figura 5.5

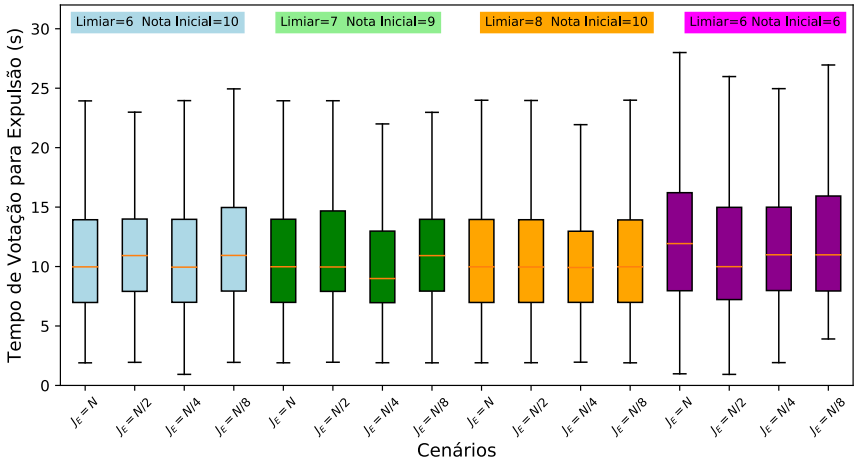


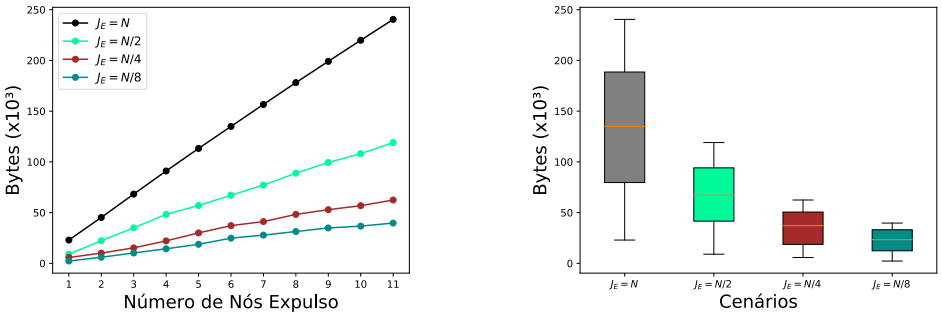
Figura 5.5: Tempo de votação pela expulsão do nó malicioso, desde a emissão da primeira $\text{Tx}_{\text{votação}}$ até a emissão da transação $\text{Tx}_{\text{expulsão}}$.

apresenta o resultado em média e intervalo de confiança para os 16 cenários de rede. Este tempo é calculado desde o momento em que o primeiro juiz emite a $\text{Tx}_{\text{votação}}$, solicitando a expulsão do nó até a emissão da $\text{Tx}_{\text{expulsão}}$. Os resultados apresentados mostram que o tempo médio de votação em todos os cenários está em torno de 10 s. Vale ressaltar que a votação converte para a expulsão do nó quando são mineradas $\frac{J}{2} + 1$ transações $\text{Tx}_{\text{votação}}$ e o tempo de geração de blocos no simulador varia entre 1 e 10 s. Por isso, os resultados mostram que as transações $\text{Tx}_{\text{votação}}$ são geralmente mineradas no mesmo bloco da cadeia e resultam na convergência das médias de tempo em todos os cenários simulados. O intervalo de confiança mostra que algumas votações para a expulsão resultaram no total de transações necessárias sendo mineradas em blocos diferentes.

Em relação à carga em bytes acumulada com as transações de votação e expulsão, a Figura 5.6(a) mostra o crescimento linear da carga, de acordo com o número de J_E, em uma simulação em que é possível expulsar os 11 nós mineradores maliciosos dentro de uma hora de simulação. Os resultados mostram que com $J_E = N$ a carga gerada é duas vezes maior do que com $J_E = \frac{N}{8}$, cinco vezes maior que com $J_E = \frac{N}{4}$ e dez vezes maior que com $J_E = \frac{N}{2}$, de acordo com a proporção de transações e assinaturas, que aumenta com o número de juízes.

A Figura 5.6(b) mostra os resultados, em média, para a carga em bytes para trinta simulações de cada cenário proposto. Quanto o maior o intervalo de confiança e o valor médio de bytes inferior ao valor medido das simulações com sucesso de expulsão, menos

expulsões acontecem.



(a) Crescimento linear da carga em bytes por expulsões

(b) Carga média em bytes por expulsões

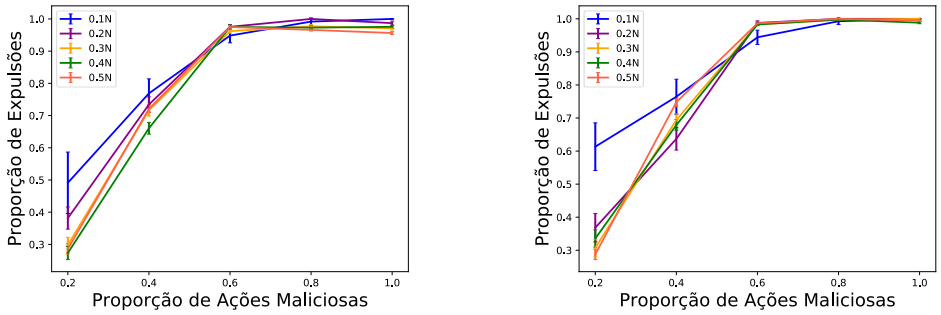
Figura 5.6: A análise do crescimento linear da carga em bytes. (a) Apresenta a carga em bytes gerada para um caso de sucesso de expulsão de onze nós maliciosos da rede em uma hora. (b) A carga em média acumulada durante uma hora em trinta simulações em cada cenário.

Simulação alterando a proporção de nós maliciosos e ações maliciosas

A partir dos resultados apresentados nas simulações anteriores, observa-se que, no cenário que adotou a nota inicial de reputação igual ao limiar de confiança ($NI = L$), o tempo em que os nós maliciosos realizam ações na rede é, em média, quatro vezes menor do que nas outras abordagens. Sendo assim, essa abordagem é adotada, por ser a mais segura para o mecanismo. Em relação ao número de juízes por nó minerador, no cenário em que são esperados $J_E = N$ a carga é muito alta e os de resultados semelhantes a cenários com menos juízes, evidenciando que não é necessário utilizar um consenso com todos os nós da rede sobre a expulsão de nós maliciosos. Contudo, os resultados alcançados para $J_E = \frac{N}{8}$ não mostram o real tempo de expulsão de todos os nós maliciosos, pois, em algumas simulações, os nós maliciosos não recebem o número de juízes suficiente para que a votação pela expulsão aconteça. Por isso, são utilizados os cenários $J_E = \frac{N}{2}$ e $J_E = \frac{N}{4}$ para simular o tempo de ação maliciosa e a proporção de nós maliciosos expulsos da rede, quando alterada a proporção de nós maliciosos na rede e a proporção de ações maliciosas que cada nó realiza.

A proporção de nós maliciosos é variada entre 10%, 20%, 30%, 40% e 50% da rede, sendo distribuídos aleatoriamente entre os $N = 100$ nós da rede. Sendo que o número de nós mineradores é $M = \frac{N}{3}$. Logo, a proporção de nós mineradores maliciosos pode ser diferente em cada simulação. Com isso, é observada a probabilidade de expulsão dos nós, a partir do número total de nós mineradores maliciosos. A Figura 5.4 mostra a proporção

de nós expulsos em relação a proporção de ações maliciosas na rede. Isso quer dizer que, nessa simulação, cada nó malicioso, em sua vez de minerar um bloco, opta por uma ação com probabilidade $0,2$, $0,4$, $0,6$, $0,8$ e 1 de ser maliciosa. Para cada uma das proporções de nós maliciosos da rede, são testadas as probabilidades de ações maliciosas. A Figura 5.7(a) mostra os resultados da média de expulsão para redes com $J_E = \frac{N}{4}$ e Figura 5.7(b) para redes com $J_E = \frac{N}{2}$ em simulações com uma hora de duração. Apesar de as duas redes apresentarem quase 100% de expulsão, quando os nós agem com uma probabilidade de ações maliciosas superior a $0,6$, a rede com $J_E = \frac{N}{2}$ apresenta maior proporção de expulsão mesmo com probabilidades de ações maliciosas inferiores a $0,6$.

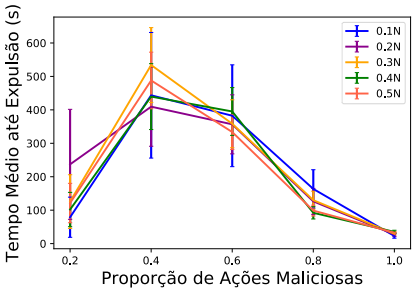


(a) Proporção de mineradores maliciosos expulsos para rede com $J_E = \frac{N}{4}$

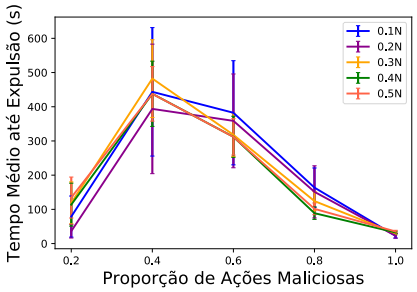
(b) Proporção de mineradores maliciosos expulsos para rede com $J_E = \frac{N}{2}$

Figura 5.7: Proporção de nós mineradores maliciosos expulsos da rede. A partir da probabilidade de ações maliciosas superiores a $0,6$, aproximadamente todos os mineradores maliciosos são expulsos durante uma hora de simulação. (a) Apresenta a proporção de expulsão para redes com $J_E = 25$. (b) Apresenta a proporção de expulsão para redes com $J_E = 50$.

O tempo de expulsão dos nós maliciosos na rede é o intervalo desde que um nó malicioso faz a primeira ação maliciosa até sua expulsão. A Figura 5.4 mostra os tempos em média para cada proporção de nós maliciosos da rede e sua probabilidade para ações maliciosas. A Figura 5.8(a) apresenta o tempo em média para redes com $J_E = \frac{N}{4}$ e a Figura 5.8(b) para redes com $J_E = \frac{N}{2}$. Os resultados não apresentam grande diferença entre as duas redes. Percebe-se que, com probabilidade de $0,2$ de ações maliciosos, os resultados para tempo de comportamento maliciosos é baixo, pois não é possível excluir todos os nós maliciosos nesse cenário em uma hora de simulação, sendo que, com probabilidade de $0,2$, o nó pode não ter executado nenhuma ação maliciosa dentro do tempo da simulação. Esses resultados representam somente os nós que agem maliciosamente e são excluídos durante as simulações. Logo, é possível haver nós que agem maliciosamente por mais tempo, porém não são computados nesta análise. Em torno de 70% dos nós



(a) Tempo médio de expulsão dos nós maliciosos para rede de $J_E = \frac{N}{4}$



(b) Tempo médio de expulsão dos nós maliciosos para rede de $J_E = \frac{N}{2}$

Figura 5.8: Tempo desde a primeira ação maliciosa até a expulsão do nó da rede, alterada a probabilidade de agir maliciosamente. Os resultados são semelhantes para os dois cenários. Observando os resultados, quando a probabilidade de ação maliciosa é de $0,2$, o tempo de expulsão dos nós maliciosos é menor, pois não é possível expulsar todos os nós maliciosos em uma hora de simulação.

expulsos da rede que agem com uma probabilidade de ações maliciosas de $0,4$ são os nós que agem por mais tempo nos resultados computados.

Então, como os dois cenários apresentam resultados semelhantes, mesmo alterando as proporções de nós maliciosos e as probabilidades de ações maliciosas, a melhor abordagem é de $J_E = \frac{N}{4}$, pois acumula menos cargas de transações de expulsão.

Capítulo 6

Aplicação Distribuída de Registros Médicos Eletrônicos *

6.1 Introdução

A adoção de Registros Médicos Eletrônicos (EMRs) desempenha um papel crítico no aprimoramento da inteligência, qualidade, experiência do usuário e custos relacionados ao sistema de saúde [57]. O EMR armazena as informações privadas do paciente em relação ao diagnóstico e tratamentos. Essas informações privadas são altamente confidenciais, mas são frequentemente compartilhadas entre pares não confiáveis, como provedores de serviços de saúde, farmácias, familiares de pacientes e outros médicos [21]. Portanto, o gerenciamento de EMR impõe o desafio de preservar a privacidade do paciente, mas garantindo a disponibilidade de dados para os pares autorizados.

Um paciente deve poder controlar a entrega de suas informações apenas para os pares em que confia. Além disso, os dados de EMR devem ser acessíveis a vários pares de diferentes instituições. A falta de interoperabilidade entre os sistemas de clínicas e hospitais dificulta o compartilhamento de informações entre pares no sistema de saúde [3]. Os registros de saúde são mantidos fragmentados principalmente em bancos de dados locais, o que impede que um paciente tenha um registro médico eletrônico consolidado [43]. Da mesma forma, um paciente perde o controle de onde seus dados de saúde estão e de quem os está acessando.

A comunicação entre os participantes do setor de saúde, por meio de interfaces de

*Este capítulo é baseado no artigo submetido para IEEE International Conference on Communications: "Towards a Blockchain-based Secure Eletronic Medical Record for Healthcare Applications". Agradecimentos pela colaboração dos autores Lúcio H. A. Reis, Ricardo C. Carrano, Flávio L. Seixas, Débora C. M. Saade, Célio V. Albuquerque, Natalia C. Fernandes, Silvia D. Olabarriaga, Dianne S. V. Medeiros e Diogo M. F. Mattos

programação de aplicativos (APIs), denominada integração B2B (*business to business*), é essencial para permitir a unificação do EMR. É obrigatório padronizar interfaces, para facilitar a comunicação entre empresas. A padronização da integração B2B se concentra em três pilares: o formato de dados, o processo de negócios e o protocolo de comunicação [42]. A tecnologia de cadeia de blocos satisfaz a esses pilares, pois define um único formato de dados em todos os nós da rede par a par e, portanto, todos os participantes seguem o padrão. Consequentemente, o uso da cadeia de blocos reduz os custos de integração.

Os sistemas de saúde baseados em cadeias de blocos são uma estrutura de dados altamente distribuída, comumente proposta para armazenar, compartilhar e consultar EMR [59]. A cadeia de blocos permite que o EMR seja verificado e registrado por meio de um consenso de pares na rede, assegurando a integridade dos dados, a responsabilidade e o não-repúdio [13]. No entanto, o desafio das aplicações em cadeia de blocos é limitar o acesso dos pares às informações armazenadas e especificar permissões refinadas sobre a privacidade dos usuários.

Neste capítulo é proposta uma abordagem de prontuário eletrônico baseada em cadeias de blocos privadas permissionadas, que criptografam e armazenam as informações do paciente como uma transação da cadeia. A ideia principal é que o paciente mantenha o controle sobre cada transação para acessar suas informações, uma vez que todos os dados são armazenados na cadeia usando um algoritmo de criptografia simétrica. Um paciente dá acesso ao seu EMR apenas a partes confiáveis. Para este fim, o paciente executa, de forma segura, uma transação para transferir uma chave de sessão para decifrar o seu EMR, para o endereço na rede do nó confiável.

Por isso, aplica-se uma abordagem híbrida, na qual os certificados emitidos por uma Autoridade de Certificação (CA) identificam cada participante. A abordagem híbrida atende aos requisitos de segurança para aplicativos médicos e também evita o pseudo-anonimato fornecido pelas tecnologias de cadeia de blocos públicas [44]. Além disso, a proposta depende de uma rede par a par composta por nós participantes da cadeia. Cada nó é colocado em uma instituição de saúde e é responsável por todas as transações originadas dessa instituição. Essa abordagem adota um mecanismo de consenso baseado em confiança, para manter baixa a sobrecarga de processamento. Como a rede de cadeia de blocos é composta por várias instituições de saúde, a proposta trata do desafio de consolidar registros médicos em um banco de dados distribuído globalmente, preservando a privacidade do paciente.

Trabalhos anteriores desenvolvem soluções de cadeia de blocos para armazenar e com-

partilhar EMR [3, 59, 57, 21]. No entanto, algumas propostas fornecem apenas um controle de privacidade de baixa granularidade [57] ou propõem o uso de cadeias de blocos públicas com mecanismos de consenso que exigem muito processamento [3]. Outros trabalhos propõem o uso de cadeias de blocos permissionadas para armazenamento de EMR, mas falham em fornecer controle de acesso de baixa granularidade às informações do paciente [21]. Além disso, há também algumas propostas para considerar o uso da computação em nuvem no armazenamento e proteção de EMRs [15]. Ao contrário de trabalhos anteriores, esta proposta se concentra em fornecer controle de acesso de baixa granularidade às informações do paciente, através do compartilhamento seletivo de uma chave de sessão entre o paciente e um parceiro confiável pretendido.

É simulada a abordagem da cadeia proposta, para verificar sua escalabilidade. Os resultados mostram que essa se adapta bem, já que a sobrecarga de transação é diretamente proporcional ao número de novas transações na rede. Os resultados evidenciam que, com esta abordagem, o tempo médio de validação da transação permanece constante, quando o número de mineradores aumenta devido ao crescimento do tamanho da rede. Portanto, a abordagem é escalável e garante privacidade para as informações do paciente, ao mesmo tempo em que disponibiliza as informações entre pares autorizados.

6.2 Trabalhos Relacionados

Abordagens recentes para o gerenciamento de Registros Médicos Eletrônicos (EMR) consistem em simplesmente armazenar registros médicos em uma cadeia de blocos [39] e na criação de uma rede social abrangente, permitindo aos usuários compartilhar dados coletados por sensores médicos e armazenados na cadeia [58]. Uma preocupação primária sobre o atual EMR é que as instituições de saúde, por exemplo hospitais, controlem inteiramente o EMR em vez dos pacientes, os proprietários reais [37]. Além disso, os sistemas de gerenciamento de identidades controlam os dados de saúde e aumentam a confiança e a privacidade dos sistemas EMR [18]. No entanto, a centralização do gerenciamento de identidades introduz um ponto único de falha e um gargalo em todo o sistema [21]. Guo *et al.* [29] propõem um sistema de registro médico eletrônico distribuído usando a tecnologia de cadeia de blocos, e um esquema de assinatura baseado em atributos com múltiplas autoridades. Em sua proposta, um paciente endossa uma transação de acordo com o atributo dado, sem divulgar nenhuma outra informação além da transação endossada. Embora o esquema forneça a um paciente a capacidade de gerenciar e compartilhar seus registros médicos com segurança, ele introduz uma sobrecarga, pois a transação deve

ser assinada por várias autoridades. Além disso, não aborda a confidencialidade dos dados armazenados na cadeia. Da mesma forma, Dang *et al.* [15] propõem o uso de criptografia baseada em atributos, para controlar a privacidade dos dados de saúde em um ambiente híbrido de neblina (*mist*) e nuvem.

Em relação à segurança e privacidade dos registros médicos, Yue *et al.* propõem um aplicativo para celular que interage com um aplicativo *gateway* que controla o acesso do paciente ao EMR armazenado na cadeia de blocos [57]. O mecanismo proposto, no entanto, fornece apenas um controle de acesso de baixa granularidade, pois controla o acesso aos blocos e não às transações. Dubovitskaya *et al.* [21] analisam vários cenários para a aplicação de cadeias de blocos permissionadas, e propõem uma estrutura simples para armazenar e consultar dados EMR. No entanto, os autores enfocam as permissões de cadeias de blocos e não discutem um controle de acesso refinado sobre os dados armazenados na cadeia. Em contraste, o MedRec é um protótipo prático de EMR em cadeia de blocos [3]. O MedRec é baseado em uma cadeia de blocos pública e conta com o protocolo de consenso Prova de Trabalho (PoW), que é computacionalmente custoso. Assim, o MedRec aborda o desafio de existirem registros médicos fragmentados entre várias instituições de saúde, ao custo de altos gastos com processamento computacional. Zhang *et al.* discutem a granularidade do controle de acesso aos dados armazenados em uma cadeia de blocos [59]. Os autores afirmam que a permissão em nível de bloco não oferece controle de permissão apropriado aos dados eletrônicos de saúde. Em contraponto, propõem acesso em camadas ao conteúdo criptografado dentro da cadeia. Assim, sua proposta introduz um atraso na recuperação de dados, uma vez que a camada de controle de acesso valida cada acesso.

Nesta proposta, desenvolve-se uma abordagem de prontuário eletrônico cujo controle de acesso é centrado no paciente. A abordagem depende da Infraestrutura de Chave Pública (PKI) e da tecnologia de cadeia de blocos. A ideia chave é herdar a confiança na autenticidade fornecida pela PKI e a integridade e a auditabilidade fornecidas pela cadeia de blocos. Ao contrário de trabalhos anteriores, propõe-se um EMR distribuído seguro que requer uma infraestrutura computacionalmente simples e fornece controle de acesso refinado, com pouca sobrecarga.

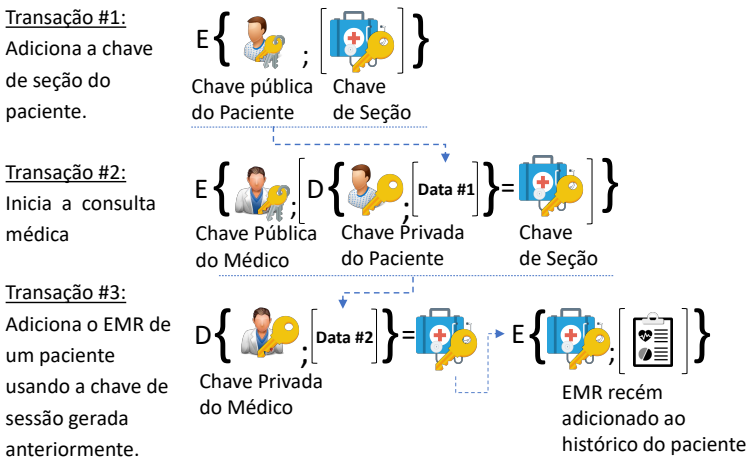


Figura 6.1: Três tipos principais de transações criptografadas da cadeia de blocos proposta. A primeira transação é adicionar a chave de sessão do paciente na cadeia. A segunda transação é o compartilhamento da chave de sessão entre o paciente e um médico durante uma consulta médica. A terceira transação é realizada pelo médico, ao adicionar um novo EMR ao paciente.

6.3 Sistema Híbrido de PKI e Cadeia de Blocos

Esse trabalho propõe uma abordagem de prontuário eletrônico seguro como um sistema híbrido, no qual chaves assimétricas são suportadas por uma infraestrutura de chave pública (PKI), enquanto chaves de sessão secreta são usadas para armazenar registros médicos em uma cadeia de blocos distribuída. A proposta se concentra em fornecer privacidade aos dados do paciente, ao mesmo tempo em que fornece responsabilidade e não repúdio às atividades dos médicos. Portanto, é possível identificar e a autenticar pacientes e médicos usando certificados digitais fornecidos por uma Autoridade de Certificação (CA) oficial.

Vale ressaltar que alguns países, como o Brasil, possuem regulamentações nas quais os certificados digitais são obrigatórios para identificar usuários em sistemas eletrônicos de saúde, obedecendo a uma PKI. Embora confiar na PKI pareça ser uma abordagem contrária ao uso de uma cadeia de blocos, a proposta híbrida une as vantagens de ambas. O PKI, no entanto, ainda depende de uma CA, que é a âncora de confiança do processo de autenticação da nossa proposta.

Consideramos duas funções de usuário principais. O primeiro papel é do paciente, que tem seus dados armazenados na cadeia. As duas principais ações da função de paciente

são armazenar uma chave de sessão e enviar a chave da sessão a um médico. Essas duas ações simples são suficientes para assegurar que o controle de acesso dos dados pessoais de um paciente seja de fato centrado no paciente.

O segundo papel é atribuído ao médico que presta assistência médica a um paciente. O médico pode enviar e receber uma chave de sessão de um paciente, mas é proibido de enviar uma chave de sessão para outro médico. Essa proibição acontece, porque a troca de chaves entre dois médicos é contrária à ideia fundamental de manter a autorização de acesso aos registros de saúde de um paciente centrado no paciente.

Todos os atores que escrevem e leem a cadeia devem ter um par de chaves públicas e privadas, em que as chaves públicas devem ser apresentadas como um certificado X.509 assinado por uma CA confiável, associada à PKI. Cada transação inserida na cadeia de blocos é identificada pelas chaves públicas do paciente e do médico apresentadas em seus certificados. Como os certificados são emitidos por uma CA, qualquer nó que tenha uma cópia da cadeia valida as transações, verificando se os atores estão realmente assinando transações com suas chaves privadas. Além disso, a validação das assinaturas apresentadas é computada localmente. O processo de validação de transação também verifica se a transação sob validação está de acordo com a função de ação do certificado apresentado, que está assinando a transação.

O núcleo de segurança da proposta é a cadeia de blocos privada. Cadeias de blocos privadas requerem um protocolo de consenso menos custoso computacionalmente do que o requerido por uma cadeia pública. Assim, a proposta adota o mecanismo de consenso baseado em confiança.

Nesta abordagem, a rede é composta por nós específicos, colocados em centros de saúde, responsáveis por garantir o armazenamento das transações na estrutura de dados distribuídos da cadeia de blocos. O protocolo de consenso é executado entre esses nós, porque eles são os únicos que podem gerar bloco da cadeia. A segurança do ambiente é obtida com a implantação de uma rede privada virtual, criptografada com SSL/TLS e com controle de acesso baseado em certificação, para conectar nós que possuem privilégios de gravação.

Para garantir o consenso, os nós devem provar que são certificados com um certificado válido, no qual a função do nó é definida para a cadeia, certificado de médico e de paciente. Contanto que o nó esteja certificado com o certificado válido, ele pode fechar e assinar um novo bloco de transações. Para fechar um bloco de transações, um dos nós certificados calcula o *hash* do bloco inteiro, incluindo o cabeçalho do bloco, e assina o *hash* do bloco

com a chave privada do nó. Quando o bloco é fechado e assinado, o nó proponente inicia um protocolo de consenso para difundir o novo bloco na rede. Na abordagem proposta, baseada em *blockchain* para EMR, o protocolo de consenso é simplificado, para usar um protocolo de confirmação de três fases, que é simples e garante a consistência e a disponibilidade das transações [40]. O *commit* de três fases é amplamente usado em bancos de dados distribuídos e é adequado a esta abordagem, onde todos os nós participantes são confiáveis. Destacamos que é desnecessário distribuir as transações para todos os nós da rede antes de fechar o novo bloco, porque os nós não competem para resolver um desafio tipo PoW. Assim, os nós que não estão envolvidos nas transações precisam apenas receber a transação após o fechamento do bloco.

Nessa proposta, são três os tipos de transações principais, mostrados na Figura 6.1, que determinam um conjunto de transações primitivas que são a base da transferência de dados mais complexa através da cadeia de blocos. A primeira transação é a criação de uma chave de sessão. Todo acesso às informações do usuário é controlado pelo usuário. Assim, é uma prerrogativa do usuário compartilhar a chave para acessar suas informações apenas com outros usuários pretendidos. No contexto dos prontuários médicos, o paciente é quem controla o acesso aos seus registros de saúde. Portanto, todos os seus registros médicos são criptografados com uma chave de sessão. As chaves de sessão são armazenadas na cadeia de blocos, endereçadas com a chave pública do paciente e criptografadas com a chave pública do paciente. Dessa forma, somente o paciente pode descriptografar sua própria chave de sessão.

A ação de armazenar a chave de sessão criptografada na cadeia é essencial para fornecer o controle de acesso centrado no paciente. Para permitir que um médico acesse seus registros de saúde, o paciente realiza um segundo tipo de transação em que compartilha a chave de sessão com o médico, do seguinte modo. O paciente recupera sua chave de sessão da cadeia, endereçando-a com sua chave pública. A seguir, a chave da sessão é descriptografada com a chave privada do paciente. Como o paciente já tem sua chave de sessão, ele publica a chave de sessão na cadeia criptografada com a chave pública do médico. Este procedimento permite que o médico acesse os registros médicos do paciente, abordando-os apenas pela chave pública do paciente e descriptografando-os com a chave da sessão.

No final de uma consulta médica, o médico pode precisar publicar um novo prontuário médico para o paciente. Assim, o terceiro tipo de transação é efetivada quando o médico escreve um novo prontuário. O médico escreve um prontuário médico para um paciente

como um registro local. Depois disso, o médico criptografa o registro médico com a chave de sessão do paciente e assina o registro criptografado com a chave privada do médico. A assinatura do registro garante o não repúdio do médico ao registro. O registro criptografado e assinado é enviado como uma transação para os nós especiais que podem gerar blocos da cadeia. Como todos os registros médicos são criptografados, é garantida a confidencialidade dos dados. Além disso, como não há divulgação de uma identidade do paciente na cadeia, também é garantida sua privacidade, pois ninguém pode recuperar qualquer informação sobre qualquer paciente. A integridade dos dados e a auditabilidade de todo o banco de dados é assegurada pelas características intrínsecas da tecnologia de cadeia de blocos dos valores *hash* de encadeamento.

No entanto, alguns registros médicos podem ser grandes demais para serem armazenados na cadeia de blocos, como exames baseados em imagens e gravações de cirurgias. Esses dados devem ser armazenados, para manter o sistema de acordo com os regulamentos. Nesses casos, os grandes registros devem ser resumidos em um registro simplificado, como segue. Ao armazenar um registro grande, o registro é criptografado com a chave de sessão do paciente e sobre o resultado da criptografia é feito um resumo criptográfico (*hash*). O registro criptografado é armazenado em um dos nós especiais da cadeia como um objeto externo. O *Universal Resource Identifier* (URI) do objeto armazenado é concatenado com o valor de *hash* calculado e, em seguida, eles são armazenados em um bloco, como uma transação de um envio de registro médico comum. Embora o armazenamento de todo o objeto na cadeia forneça a disponibilidade, a integridade e a capacidade de auditoria de todos os dados, a proposta de armazenar apenas o *hash* dos dados criptografados reduz os requisitos de memória dos nós da rede, garantindo a integridade dos dados e a capacidade de auditoria ao custo da disponibilidade dos dados.

A rede prevista depende de servidores locais - nós comuns - e servidores habilitados para participar do consenso - nós com permissão de geração de blocos. Os servidores locais podem ser colocados em áreas remotas, com infraestrutura limitada. Assim, esses servidores não exigem recursos significativos de memória, de processamento e nem de armazenamento. Eles só executam criptografia e descriptografia de dados e também enviam a transação para os servidores com permissão de geração de blocos. Esses são os nós com permissão de mineradores. Cada servidor com permissão de geração de blocos possui seu próprio certificado assinado pela CA, no qual é declarada a função do servidor como minerador. Todos os servidores mineradores executam o protocolo de consenso e têm a versão mais atualizada da cadeia.

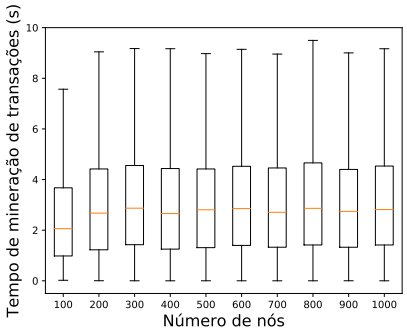
6.4 Resultados da Simulação

O EMR centralizado representa fragilidade de falha em um único ponto. Além disso, as partes devem confiar na administração do hospital, considerando que tem autoridade para alterar os dados e comprometer sua integridade e autenticidade. EMR controlado pelo hospital, em vez de o paciente dificulta a busca de registros médicos em diferentes hospitais.

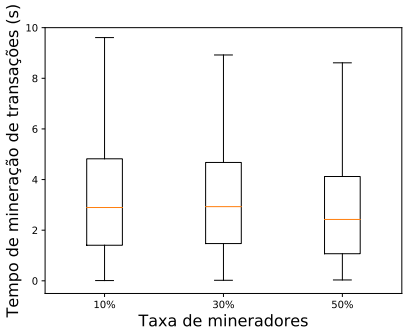
No gerenciamento de EMR em cadeia de blocos, cada parte de uma rede tem acesso a todo o banco de dados e ao seu histórico completo. Uma vez publicado e validado na cadeia, o paciente possui o controle de gerenciar e compartilhar seu próprio EMR. A identificação e autenticação confiáveis do paciente e o protocolo de controle de acesso mantêm a sensibilidade às questões legais, culturais e éticas associadas aos EMRs.

Após validar o simulador do Capítulo 5, é feita uma avaliação da abordagem proposta, através de simulações específicas para aplicações em saúde. O cenário considera que cada nó envia uma nova transação entre oito e quinze minutos, que é o tempo médio esperado para uma consulta médica no Brasil.

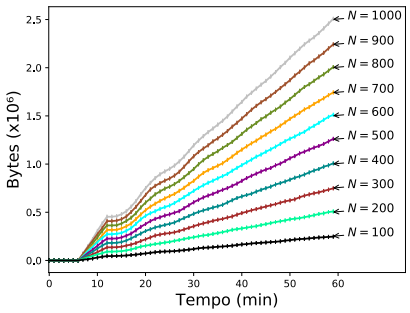
A ideia principal da simulação é avaliar o tempo médio para realmente escrever um EMR na cadeia de blocos e a sobrecarga para escrever os EMRs. O tempo para escrever um EMR na cadeia consiste na diferença entre o tempo em que o EMR é criado e o tempo em que um bloco, que contém a transação EMR, é minerado. Assim, o primeiro experimento avalia o tempo de mineração de transação para diferentes tamanhos da rede. Foram simulados cenários de 100 a 1000 nós, mostrados no *boxplot* da Figura 6.2(a). Os resultados revelam que o tempo médio para escrever uma transação na cadeia permanece o mesmo, entre 2 e 3 s, para todos os cenários avaliados. Também é avaliada a influência do número de nós de mineradores no desempenho da rede. Embora aumentasse o número de nós mineradores responsáveis por inserir blocos na cadeia, na Figura 6.2(b) pode ser visto que o tempo de mineração de uma transação permanece o mesmo, porque cada nó minerador executa geração de blocos em cada rodada. Portanto, aumentar o número de nós de mineradores reduz a carga sobre cada nó, mas não afeta o tempo de inserção de um EMR na cadeia de blocos. Também é avaliada a carga gerada em uma hora de simulação. A Figura 6.2(c) mostra que o tamanho da cadeia aumenta linearmente à medida que a simulação avança. Evidenciando que a cadeia começa a crescer após 8 min porque os parâmetros de simulação consideram que o intervalo entre transações é maior ou igual 8 min. Além disso, à medida que o número de nós na rede aumenta, o tamanho da



(a) Tempo médio de mineração de uma transação.



(b) Tempo médio de mineração de uma transação com base na taxa de mineradores.



(c) Carga gerada na cadeia de blocos em bytes.

Figura 6.2: Avaliação do tempo para executar uma transação e carga gerada. (a) À medida que a rede aumenta, o tempo para executar uma transação permanece o mesmo, mostrando que a proposta é escalável. (b) O número de mineradores na rede não afeta o tempo de execução de uma transação. (c) Quanto maior o número de nós gerando transações na rede, maior é a carga armazenada. A cadeia cresce linearmente de acordo com o número de transações.

cadeia aumenta, porque mais nós geram transações. Portanto, os resultados mostrados na Figura 6.2 endossam a ideia de que a abordagem proposta para um EMR baseado em cadeia de blocos é dimensionada para o número de nós na rede, e a disponibilidade de dados é assegurada após os 10 seg do limite superior de tempo para processar uma transação cadeia de blocos.

Capítulo 7

Conclusão

A tecnologia de cadeia de blocos apresenta diversas vantagens de segurança para aplicações distribuídas. Apesar de o caso de uso de maior sucesso ser voltado para uma rede pública não permissionada, diversos estudos surgem com o interesse de utilizar a tecnologia em aplicações em rede privadas permissionadas. O controle de acesso à rede privada e a distribuição de permissões para a participação do consenso trouxeram a oportunidade de usar mecanismos de consenso não competitivos e, ao mesmo tempo, escaláveis.

Com o avanço da tecnologia e dos mecanismos de consenso, surge a oportunidade da aplicação das cadeias de blocos no armazenamento de históricos médico e hospitalar. O desafio dessa aplicação é atribuir segurança e privacidade às informações sensíveis do paciente. Atualmente, o controle dos dados é centralizado nos hospitais e clínicas, apresentando problemas como a falta de interoperabilidade, defasagem nas atualizações dos prontuários e indisponibilidade das informações para o paciente. Como solução, as cadeias de blocos apresentam uma abordagem distribuída para o armazenamento dos prontuários médicos eletrônicos (EMR) que necessitam do consenso em uma rede privada permissionada.

Alcançar o consenso em aplicações distribuídas é um desafio. Muitas soluções apresentadas para mecanismos de consenso em redes privadas permissionadas se mostram mais eficientes, porém, não são verdadeiramente distribuídas. Além disso, assumem um comportamento benigno dos nós, não oferecendo um modelo de confiança para monitorar e expulsar nós maliciosos da rede. Contudo, como as aplicações em cadeia de blocos dispensam a terceira entidade confiável, é do interesse comum dos nós o bom funcionamento da rede. Isso oferece a oportunidade de utilizar a cooperação dos nós para monitorar o mecanismo de consenso aplicado. Essa dissertação propôs um mecanismo de consenso baseado em confiança para redes privadas permissionadas, que apresenta controle de acesso,

monitoramento da ação dos nós e expulsão de nós maliciosos.

Na primeira parte desta dissertação, foi analisado o desempenho de duas plataformas de desenvolvimento de cadeias de blocos privadas permissionadas, *Parity* e *Multichain*, que usam o mecanismo de consenso PoA. O objetivo foi avaliar a possibilidade da aplicação de uma das plataformas para o armazenamento EMRs, que exige mecanismos de buscas por informações eficientes e atualizações da cadeia com baixa latência.

Através de cargas de trabalho realísticas, foi possível verificar o funcionamento das plataformas e analisar o tempo necessário para a validação de transações, para buscas por transações e por blocos da cadeia e o tempo efetivo de uma transação emitida ser adicionada a um bloco minerado na cadeia. Os resultados demostraram que a plataforma *Multichain* apresenta melhor resultado para a métrica de tempo de validação de transações, ao custo de suportar somente transações de transferência de dados, não suportando a execução de contratos inteligentes. A *Multichain* também apresentou melhor desempenho em tempo de busca por blocos, o que pode ser justificado pelo encadeamento bi-direcional de *hash* em uma camada superior à cadeia de blocos. Já a *Parity* apresentou melhor desempenho no tempo de busca por transações, evidenciando que estrutura de encadeamento das transações dentro de bloco foi mais eficiente. Além disso, o *Parity* apresentou vantagem no resultado de tempo de mineração de transações em que o tempo variou com ocorrências desde 0 a 7 s , enquanto no *Multichain* apresentou ocorrências em 9 s. Os resultados temporais alcançados nas duas plataformas são suficientes para atender a aplicação em EMR. Contudo, a implementação do mecanismo PoA concentrou o poder de mineração em um grupo de nós de autoridade, sem um modelo de confiança responsável por monitorar o comportamento desses nós. Assim, o consenso deixa de ser puramente distribuído e se torna um risco à segurança de toda a rede, no caso de ataques de conluio entre mineradores.

A concepção de um modelo de confiança que pudesse ser aplicado ao mecanismo de consenso PoA oferece retomar a característica distribuída ao mecanismo de consenso. Por isso, foi proposto o mecanismo de consenso baseado em confiança que realiza um controle de acesso capaz de auto organizar a rede, seguindo a proporção do número de nós mineradores desejados na rede. O nós mineradores são monitorados por um júri formado de forma pseudoaleatória, através de um filtro de *Bloom* que permite estimar o o número de juízes que cada minerador receberá. O júri monitora todas as ações realizadas pelo réu minerador, atualizando uma nota de reputação R. Caso a reputação do nó esteja abaixo do limiar de confinação L, é realizada uma votação e determinada a expulsão de um

nó julgado malicioso. Para o funcionamento desse mecanismo, foi necessário determinar a proporção de nós mineradores M, a proporção de nós juízes esperando J_E, a nota inicial de reputação do minerador NI e o limiar de confiança da rede L.

Para tanto, foi desenvolvido um simulador de tempo discreto, baseado nos dados reais medidos na implementação da plataforma *Multichain*. No simulador foram desenvolvidos os processos necessários para que o modelo de confiança fosse implementado entre os nós da rede. Assim, foi possível simular o controle de acesso de novos nós, o que acarretava no aumento do número de nós mineradores segundo o critério de proporcionalidade empregado. Também foi simulado o monitoramento de nós maliciosos, com diferente número de juízes esperados J_E, notas iniciais NI de reputação e limiar de confiança L. Por fim, foram simulados diferentes cenários de rede com aumento da proporção de nós maliciosos e da probabilidade de ações maliciosas que cada um exercia na rede, a fim de analisar o quanto resiliente a ações maliciosas é o mecanismo.

Na simulação de controle de acesso, foram testados os cenários de $M = \frac{N}{2}$, $M = \frac{N}{3}$ e $M = \frac{N}{4}$ durante uma hora, com entradas de novos nós acontecendo dentro de um intervalo de 1 min. Os resultados dessas simulações mostram que a rede automaticamente adaptou-se ao crescimento, dando permissões para novos nós agirem como mineradores. O tempo de acesso à rede foi menor no cenário em que $M = \frac{N}{2}$, enquanto a carga em bytes gerada durante os novos acessos foi duas vezes superior ao cenário $M = \frac{N}{4}$, pela necessidade de emissão do dobro de transações $\mathrm{Tx}_{\mathrm{minerador}}$ para manter a proporcionalidade de mineradores na rede. Por isso, nas simulações seguintes, optou-se pelo cenário de $M = \frac{N}{3}$, por apresentar desempenho ponderado nas duas métricas.

Na simulação de monitoramento, a abordagem de atribuir a nota inicial de reputação igual ao limiar de confiança, $NI = L$, proporcionou o menor tempo de ação maliciosa dos nós na rede, em relação às outras abordagens. Portanto, como esperado, é arriscado atribuir crédito de reputação a um novo nó sem avaliar seu comportamento prévio. O tempo de votação por expulsão de nós maliciosos foi em média similar em todos os cenários, por se tratar de uma simulação de rede malha completa. Em relação ao sucesso de expulsão em uma hora de simulação, o cenário com $J_E = \frac{N}{8}$ foi inferior aos demais, pois o número de nós que verificavam a pertinência não foi suficiente para que o filtro de *Bloom* convergisse para o valor esperado de juízes em uma rede com somente 100 nós. De acordo com a análise da carga gerada, os cenários com $J_E = \frac{N}{2}$ e $J_E = \frac{N}{4}$ tiveram resultados similares nesta simulação e o mecanismo se mostrou eficiente na expulsão dos nós maliciosos. Assim, utilizou-se os $J_E = \frac{N}{4}$ e $J_E = \frac{N}{2}$ para testar a percepção do

júri sobre os nós maliciosos. Na simulação, foram alteradas as proporções do número de nós maliciosos e a probabilidade de ações maliciosas. Em uma hora foi possível expulsar quase 100% dos nós maliciosos que agiram com probabilidade de, no mínimo, $0,6$ ações maliciosas. Em relação ao tempo de comportamento malicioso dos nós, o pior caso de probabilidade de ações maliciosas medido foi de $0,4$, em que os nós maliciosos ficaram até 400 s desde a primeira ação maliciosa até a expulsão. Os resultados mostraram que o mecanismo foi efetivo na expulsão de nós maliciosos, com resultados similares para os $J_E = \frac{N}{4}$ e $J_E = \frac{N}{2}$. No entanto, a carga gerada no cenário com $J_E = \frac{N}{4}$ para a votação é inferior, logo esta se torna a abordagem mais eficiente.

Conclui-se que o mecanismo de consenso proposto é eficiente para a auto-organização da rede de maneira distribuída. Além disso, o mecanismo oferece um monitoramento constante dos nós mineradores, a partir de um júri formado de maneira pseudo aleatória, improvável de ser pré-selecionado pelo nó minerador monitorado. Visto que se for observada manipulação na geração das chaves, o nó que está manipulando a geração da chave já é eliminado na fase de solicitação de acesso. O mecanismo também se mostrou eficiente para a expulsão dos nós maliciosos que agiram com mais de 50A partir do mecanismo de consenso proposto, foi desenvolvida uma abordagem híbrida que combina as vantagens da cadeia de blocos e de uma infra-estrutura de chave pública, para desenvolver um sistema para registros eletrônicos de saúde, capaz de fornecer segurança e privacidade, com o controle de acesso centrado no paciente. Embora o uso da infraestrutura de chave pública seja obrigatório em alguns países, a proposta cumpre os requisitos exigidos pela lei brasileira para prontuários eletrônicos e é verdadeiramente distribuído, com alto nível de privacidade para os registros de saúde do paciente. Esta abordagem também mantém o sigilo entre paciente e médico devido à validação de todas as transações que são adicionadas à cadeia, uma vez que a divulgação de uma chave do paciente entre os médicos é proibida. Além disso, os resultados obtidos por meio de simulação mostram que a abordagem é escalável, pois o tempo para minerar transações permanece constante, quando o número de nós na rede aumenta, e o tamanho da cadeia aumenta linearmente com o crescimento da rede.

Como trabalho futuro, serão avaliados métodos para mensurar as ações maliciosas. Assim, uma única ação que tenha potencial de comprometer a rede pode levar à votação pela expulsão do nó malicioso, permitindo aumentar a efetividade do mecanismo de confiança, mesmo em cenários em que os nós raramente se comportem de forma maliciosa. Além disso, será avaliada a interferência de ações maliciosas entre todos os nós da rede, podendo ter resultados diferentes na votação, caso os juízes também ajam de maneira

não confiável. Com isso, serão simuladas ações maliciosas junto à abordagem híbrida para aplicação em registros médico eletrônicos e será construído um caso de uso real do mecanismo proposto.

Referências

[1] Alvarenga, I. D.; Rebello, G. A. F.; Duarte, O. C. M. B. Securing configuration management and migration of virtual network functions using blockchain. In *Proc. of IEEE/IFIP Network Operations and Management Symposium (NOMS)* (2018), pp. 1–9.

[2] Antonopoulos, A. M. *Mastering Bitcoin: unlocking digital cryptocurrencies*, 1 ed. O'Reilly Media, Inc., Dec. 2014.

[3] Azaria, A.; Ekblaw, A.; Vieira, T.; Lippman, A. MedRec: Using blockchain for medical data access and permission management. In *OBD '16* (Aug. 2016), pp. 25–30.

[4] Bach, L. M.; Mihaljevic, B.; Zagar, M. Comparative analysis of blockchain consensus algorithms. In *Proc. of International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)* (May 2018), pp. 1545–1550.

[5] Bessani, A.; Sousa, J.; Alchieri, E. E. P. State machine replication for the masses with BFT-SMART. In *Proc. of IEEE/IFIP International Conference on Dependable Systems and Networks* (2014), pp. 355–362.

[6] Bloom, B. H. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM 13*, 7 (1970), 422–426.

[7] Buterin, V., et al. Ethereum white paper, 2014. Tech. rep., 2013. Disponível em: `https://github.com/ethereum/wiki/wiki/White-Paper`.

[8] Cachin, C. Architecture of the hyperledger blockchain fabric. In *Workshop on Distributed Cryptocurrencies and Consensus Ledgers* (2016).

[9] Cachin, C.; Vukolic, M. Blockchain consensus protocols in the wild. In *International Symposium on Distributed Computing (DISC)* (Oct. 2017), pp. 1–16.

[10] Castro, M.; Liskov, B. Practical Byzantine fault tolerance. In *Symposium on Operating Systems Design and Implementation (OSDI)* (1999), pp. 173–186.

[11] Castro, M.; Liskov, B. Practical byzantine fault tolerance and proactive recovery. *ACM Transactions on Computer Systems (TOCS) 20*, 4 (2002), 398–461.

[12] Chicarino, V. R. L.; Jesus, E. F.; Albuquerque, C. V. N.; Rocha, A. A. A. Uso de blockchain para privacidade e segurança em Internet das coisas. In *Minicursos do Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSeg)*. Sociedade Brasileira de Computação (SBC), 2017, pp. 100–150.

[13] Christidis, K.; Devetsikiotis, M. Blockchains and smart contracts for the internet of things. *IEEE Access 4* (2016), 2292–2303.

[14] Dai, M.; Zhang, S.; Wang, H.; Jin, S. A low storage room requirement framework for distributed ledger in blockchain. *IEEE Access* (2018).

[15] Dang, L.; Dong, M.; Ota, K.; Wu, J.; Li, J.; Li, G. Resource-efficient secure data sharing for information centric e-health system using fog computing. In *ICC '18* (May 2018), pp. 1–6.

[16] de Oliveira, M. T.; Carrara, G. R.; Fernandes, N. C.; Carrano, R. C.; Albuquerque, C. V. N.; Medeiros, D. S. V.; Mattos, D. M. F. Uma avaliação de desempenho de cadeias de blocos privadas permissionadas através de cargas de trabalho realísticas. In *XVIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSeg'2018)* (Natal/RN, Brazil, Oct. 2018).

[17] Dinh, T. T. A.; Wang, J.; Chen, G.; Liu, R.; Ooi, B. C.; Tan, K.-L. Blockbench: A framework for analyzing private blockchains. In *Proceedings of the 2017 ACM International Conference on Management of Data* (2017), ACM, pp. 1085–1100.

[18] Dolera Tormo, G.; Gomez Marmol, F.; Girao, J.; Martinez Perez, G. Identity management–in privacy we trust: Bridging the trust gap in ehealth environments. *IEEE Security and Privacy 11*, 6 (Nov. 2013), 34–41.

[19] Dolev, D.; Yao, A. On the security of public key protocols. *IEEE Transactions on information theory 29*, 2 (1983), 198–208.

[20] Douceur, J. R. The sybil attack. In *International workshop on peer-to-peer systems* (2002), Springer, pp. 251–260.

[21] Dubovitskaya, A.; Xu, Z.; Ryu, S.; Schumacher, M.; Wang, F. Secure and trustable electronic medical records sharing using blockchain. In *AMIA '17* (2017), vol. 2017, pp. 650–659.

[22] Eyal, I.; Sirer, E. G. Majority is not enough: Bitcoin mining is vulnerable. *Communications of the ACM 61*, 7 (2018), 95–102.

[23] Fernandes, N. C.; Duarte, O. C. M. B. Controle de acesso auto-organizavel e robusto baseado em nós delegados para redes ad hoc.

[24] Ferraz, L. H. G.; Velloso, P. B.; Duarte, O. C. M. An accurate and precise malicious node exclusion mechanism for ad hoc networks. *Ad hoc networks 19* (2014), 142–155.

[25] Gauld, S.; von Ancoina, F.; Stadler, R. The burst dymaxion: An arbitrary scalable, energy efficient and anonymous transaction network based on colored tangles. Tech. rep., 2017.

[26] Greenspan, G. Multichain private blockchain, 2015. White Paper. Available on: http://www.multichain.com/download/MultiChain-White-Paper.pdf.

[27] Greve, F.; Sampaio, L.; Abijaude, J.; Coutinho, A.; Ítalo Valcy; Queiroz, S. Blockchain e a revolução do consenso sob demanda. In *Minicursos do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)*. Sociedade Brasileira de Computação (SBC), 2018, pp. 1–52.

[28] Greve, F. G. P. Protocolos fundamentais para o desenvolvimento de aplicaç oes robustas. In *Minicursos SBRC 2005: Brazilian Symposium on Computer Networks* (2005), pp. 330–398.

[29] Guo, R.; Shi, H.; Zhao, Q.; Zheng, D. Secure attribute-based signature scheme with multiple authorities for blockchain in electronic health records systems. *IEEE Access 6* (2018), 11676–11686.

[30] Gupta, S.; Sadoghi, M. *Blockchain Transaction Processing*. Springer International Publishing, 2018, pp. 1–11.

[31] Han, R.; Gramoli, V.; Xu, X. Evaluating blockchains for iot. In *Proc. of IFIP International Conference on New Technologies, Mobility and Security (NTMS)* (2018), pp. 1–5.

[32] Jesus, E. F.; Chicarino, V. R. L.; de Albuquerque, C. V. N.; Rocha, A. A. A. A survey of how to use blockchain to secure Internet of Things and the stalker attack. *Security and Communication Networks 2018* (2018), 1–28.

[33] Kiayias, A.; Russell, A.; David, B.; Oliynykov, R. Ouroboros: A provably secure proof-of-stake blockchain protocol. In *Advances in Cryptology – CRYPTO 2017* (2017), J. Katz and H. Shacham, Eds., pp. 357–388.

[34] King, S.; Nadal, S. PPCoin: peer-to-peer crypto-currency with proof-of-stake. Tech. rep., 2012.

[35] Lamport, L. Paxos made simple. *ACM SIGACT News 32*, 4 (Dec. 2001), 18–25.

[36] Laufer, R. P. *Rastreamento de Pacotes IP contra Ataques de Negação de Serviço*. Tese de Doutorado, Tese de mestrado, COPPE/UFRJ, 2005.

[37] Lesk, M. Electronic medical records: Confidentiality, care, and epidemiology. *IEEE Security Privacy 11*, 6 (Nov 2013), 19–24.

[38] Macdonald, M.; Liu-Thorrold, L.; Julien, R. The blockchain: A comparison of platforms and their uses beyond bitcoin. Tech. rep., 2017.

[39] Magyar, G. Blockchain: Solving the privacy and research availability tradeoff for ehr data: A new disruptive technology in health data management. In *NC '17* (Nov. 2017).

[40] Mattos, D. M. F.; Duarte, O. C. M. B.; Pujolle, G. A lightweight protocol for consistent policy update on software-defined networking with multiple controllers. *Journal of Network and Computer Applications* (2018). A ser publicado.

[41] Mattos, D. M. F.; Medeiros, D. S. V.; Fernandes, N. C.; de Oliveira, M. T.; Carrara, G. R.; Soares, A. A. Z.; Magalhães, L. C. S.; Passos, D.; Carrano, R. C.; Moraes, I. M.; Albuquerque, C. V. N.; Muchaluat-Saade, D. C. Blockchain para segurança em redes elétricas inteligentes: Aplicações, tendências e desafios. In *Minicursos do Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSeg)*. Sociedade Brasileira de Computação (SBC), 2018.

[42] Merz, M. Potential of the blockchain technology in energy trading. In *Blockchain Technology: An Introduction for Business and IT Managers*, D. Burgwinkel, Ed. DE GRUYTER, Alemanha, 2016, ch. 2, pp. 51–97.

[43] Mettler, M. Blockchain technology in healthcare: The revolution starts here. In *Healthcom '16* (Sept 2016), pp. 1–3.

[44] Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system.

[45] Ongaro, D.; Ousterhout, J. In search of an understandable consensus algorithm. In *Proceedings of USENIX Conference on USENIX Annual Technical Conference* (2014), USENIX ATC'14, pp. 305–320.

[46] Ouaddah, A.; Abou Elkalam, A.; Ait Ouahman, A. Fairaccess: a new blockchain-based access control framework for the internet of things. *Security and Communication Networks 9*, 18 (2016), 5943–5964.

[47] Pahl, C.; Ioini, N. E.; Helmer, S. A decision framework for blockchain platforms for iot and edge computing. In *Proc. of International Conference on Internet of Things, Big Data and Security - Volume 1: IoTBDS* (2018), pp. 105–113.

[48] Pilkington, M. *Blockchain technology: principles and applications*. Edward Elgar Publishing, 2016, pp. 225–251.

[49] Rebello, G. A. F.; Alvarenga, I. D.; Sanz, I. J.; Duarte, O. C. M. B. Sinfonia: Gerenciamento seguro de funções virtualizadas de rede através de corrente de blocos. In *Anais do WBlockchain - SBRC* (2018), vol. 1.

[50] Schwartz, D.; Youngs, N.; Britto, A. The Ripple protocol consensus algorithm. Tech. rep., Ripple Labs Inc., 2014.

[51] Singh, A., et al. Eclipse attacks on overlay networks: Threats and defenses. In *In IEEE INFOCOM* (2006), Citeseer.

[52] Tschorsch, F.; Scheuermann, B. Bitcoin and beyond: A technical survey on decentralized digital currencies. *IEEE Communications Surveys Tutorials 18*, 3 (third-quarter 2016), 2084–2123.

[53] Velloso, P. B.; Laufer, R. P.; Duarte, O. C.; Pujolle, G. Hit: A human-inspired trust model. In *Mobile and Wireless Communication Networks*. Springer, 2006, pp. 35–46.

[54] Virendra, M.; Jadliwala, M.; Chandrasekaran, M.; Upadhyaya, S. Quantifying trust in mobile ad-hoc networks. In *Integration of Knowledge Intensive Multi-Agent Systems, 2005. International Conference on* (2005), IEEE, pp. 65–70.

[55] Wood, G. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper 151* (2014), 1–32.

[56] Xu, X.; Weber, I.; Staples, M.; Zhu, L.; Bosch, J.; Bass, L.; Pautasso, C.; Rimba, P. A taxonomy of blockchain-based systems for architecture design. In *International Conference on Software Architecture* (April 2017), ICSA'17, pp. 243–252.

[57] Yue, X.; Wang, H.; Jin, D.; Li, M.; Jiang, W. Healthcare data gateways: Found healthcare intelligence on blockchain with novel privacy risk control. *Journal of Medical Systems 40*, 10 (Aug. 2016).

[58] Zhang, J.; Xue, N.; Huang, X. A secure system for pervasive social network-based healthcare. *IEEE Access 4* (2016), 9239–9250.

[59] Zhang, X.; Poslad, S. Blockchain support for flexible queries with granular access control to electronic medical records (EMR). In *ICC '18* (May 2018), pp. 1–6.

[60] Zheng, Z.; Xie, S.; Dai, H.; Chen, X.; Wang, H. An overview of blockchain technology: Architecture, consensus, and future trends. In *Proc. of International Congress on Big Data* (2017), pp. 557–564.

[61] Zhu, S.; Xu, S.; Setia, S.; Jajodia, S. Lhap: a lightweight hop-by-hop authentication protocol for ad-hoc networks. In *Distributed Computing Systems Workshops, 2003. Proceedings. 23rd International Conference on* (2003), IEEE, pp. 749–755.

[62] Zyskind, G.; Nathan, O., et al. Decentralizing privacy: Using blockchain to protect personal data. In *Proc. of Security and Privacy Workshops (SPW)* (2015), pp. 180–184.