CORRENTES DE BLOCOS EM REDES VIRTUALIZADAS: PROTOCOLOS DE CONSENSO E FATIAMENTO SEGURO DA REDE

Gabriel Antonio Fontes Rebello

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Elétrica.

Orientador: Otto Carlos Muniz Bandeira Duarte

Rio de Janeiro
Outubro de 2019

CORRENTES DE BLOCOS EM REDES VIRTUALIZADAS: PROTOCOLOS DE CONSENSO E FATIAMENTO SEGURO DA REDE

Gabriel Antonio Fontes Rebello

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Examinada por:

Prof. Otto Carlos Muniz Bandeira Duarte, Dr.Ing.

Prof. André Luiz Moura dos Santos, Ph.D.

Prof. Célio Vinicius Neves de Albuquerque, Ph.D.

Prof. Marcelo Gonçalves Rubinstein, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
OUTUBRO DE 2019

Rebello, Gabriel Antonio Fontes

Correntes de Blocos em Redes Virtualizadas: Protocolos de Consenso e Fatiamento Seguro da Rede/Gabriel Antonio Fontes Rebello. – Rio de Janeiro: UFRJ/COPPE, 2019.

XIV, 87 p.: il.; 29, 7cm.

Orientador: Otto Carlos Muniz Bandeira Duarte

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2019.

Referências Bibliográficas: p. 70 – 82.

1. blockchain. 2. nfv. 3. consenso. 4. hyperledger. I. Duarte, Otto Carlos Muniz Bandeira. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título

Aos meus pais e amigos.

Agradecimentos

Agradeço aos meus pais, Gabriel e Kátia, aos meus irmãos e à minha família por sempre incentivarem e se sacrificarem pela minha formação acadêmica. Sem eles, nada disto seria possível.

Aos meus amigos mais próximos, que sempre estiveram ao meu lado e me apoiaram em todos os momentos.

Ao meu orientador, professor Otto Carlos, e a todos os meus companheiros do Grupo de Teleinformática e Automação (GTA), que me proporcionaram o maior crescimento pessoal e profissional que já tive através de seus conselhos. Estes são os grandes responsáveis pela minha formação profissional e pela qualidade desta dissertação de mestrado. Aos amigos que construí na UFRJ, por sempre estarem presentes na caminhada na graduação e agora no mestrado.

A todos os meus professores, por sempre darem seu máximo e contribuírem para que eu pudesse me tornar um engenheiro e agora um mestre em ciências. Ao CNPq, CAPES e FAPESP por viabilizarem este trabalho e fomentarem continuamente a pesquisa de alto nível no Brasil.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

CORRENTES DE BLOCOS EM REDES VIRTUALIZADAS: PROTOCOLOS DE CONSENSO E FATIAMENTO SEGURO DA REDE

Gabriel Antonio Fontes Rebello

Outubro/2019

Orientador: Otto Carlos Muniz Bandeira Duarte

Programa: Engenharia Elétrica

A corrente de blocos (*blockchain*) é uma tecnologia disruptiva que deve revolucionar o nosso modo de viver, trabalhar e negociar. Assim como a Internet permite hoje a transferência de arquivos, esta nova tecnologia construirá uma Internet de Valores, na qual é possível transferir recursos únicos, como dinheiro, ações, propriedade intelectual, votos, etc. sem o intermédio de agentes reguladores. No entanto, um dos principais desafios de sistemas baseados em correntes de blocos é selecionar o protocolo de consenso mais adaptado para determinar a forma como se tomam as decisões coletivas em cada caso de uso. A primeira parte deste trabalho discute os conceitos e os modelos de consenso para diferentes tipos de correntes de blocos. Diversos protocolos de consenso são apresentados especificando suas características, suas vantagens, suas desvantagens e finalidades. A seguir, o trabalho propõe o uso de correntes de blocos para prover segurança à Internet baseada no fatiamento da rede (*network slicing*), que objetiva oferecer serviços fim-a-fim de rede ágeis e sob demanda para cada tipo de aplicação. As principais contribuições desta segunda parte são: i) a proposta e desenvolvimento de uma arquitetura baseada em correntes de blocos para prover auditabilidade às operações de orquestração de fatias de rede; e ii) uma proposta de categorização das correntes de blocos para atender os diversos cenários presentes nas redes do futuro. Um protótipo da arquitetura proposta foi desenvolvido e implementado utilizando a plataforma Hyperledger Fabric. No protótipo, cada fatia de rede opera sobre um canal isolado e cada corrente de blocos é implementada por um contrato inteligente. Os resultados mostram que é possível prover segurança à criação de fatias de rede, mas que a obtenção do consenso e o número de transações requeridas pelas fatias de rede são um grande desafio.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

BLOCKCHAIN IN VIRTUALIZED NETWORKS: CONSENSUS PROTOCOLS AND SECURE NETWORK SLICING

Gabriel Antonio Fontes Rebello

October/2019

Advisor: Otto Carlos Muniz Bandeira Duarte

Department: Electrical Engineering

Blockchains are a disruptive technology that will revolutionize our way of living, working and negotiating. Like the Internet today allows transfering files, this new technology will build an Internet of Value, in which it is possible to transfer unique resources such as money, stock, intellectual property, votes, etc. without the need for intermediate entities. One of the main challenges of blockchain-based systems, however, is to choose the consensus protocol to determine the way in which collective decisions are made in each use case. The first part of this work discusses the consensus concepts and models for different types of blockchains. We present several consensus protocols and clarify their features, advantages, disadvantages and main goals. Then, in a second part, we propose to use blockchains to secure the network-slicing-based Internet. Network slicing aims to offer agile and on-demand end-to-end services for each type of application. The main contributions of this second part are: i) the proposal and development of a blockchain-based architecture to provide auditability to network slicing orchestration operations; and ii) a proposal for the categorization of blockchains that address the diverse scenario of applications in future networks. A prototype of the proposed architecture was built using the Hyperledger Fabric platform. In the prototype, each network slice operates in an isolated channel and each blockchain is implemented through a smart contract. The results show that it is possible to provide security to network slice managements, but the consensus latency and transaction throughput might still pose a challenge.

Sumário

Lista de Figuras

Lista de Tabelas

SIGLAS

API - *Application Programming Interface*

CAPES - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior

CNPq - Conselho Nacional de Desenvolvimento Científico e Tecnológico

COPPE - Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa de Engenharia

DPoS - *Delegated Proof of Stake*

FAPESP - *Fundação de Amparo à Pesquisa do Estado de São Paulo*

FCAPS - *Fault, Configuration, Administration, Performance and Security*

GTA - Grupo de Teleinformática e Automação

IaaS - *Infrastructure-as-a-Service*

IoT - *Internet of Things*

IDS - *Intrusion Detection System*

IP - *Internet Protocol*

MAC - *Media Access Control*

MANO - *Management and Orchestration*

NaaS - *Network-as-a-Service*

NFV - *Network Function Virtualization*

NFVI - *Network Function Virtualization Infrastructure*

NSH - *Network Service Header*

OPNFV - *Open Platform for Network Function Virtualization*

OSD - *Object-based Storage Devices*

PaaS - *Platform-as-a-Service*

PoA - *Proof of Authority*

PoB - *Proof of Burn*

PoC - *Proof of Capacity*

PoET - *Proof of Elapsed Time*

PoS - *Proof of Stake*

PoW - *Proof of Work*

SINFONIA - *Secure vIrtual Network Function Orchestrator for Non-repudiation, Integrity, and Auditability*

SDN - *Software Defined Networks*

SFC - *Service Function Chaining*

SFP - *Service Function Path*

SFF - *Service Function Forwarder*

TCP - *Transmission Control Protocol*

TTM - *Time to Market*

UDP - *User Datagram Protocol*

UFRJ - Universidade Federal do Rio de Janeiro

VM - *Virtual Machine*

VNF - *Virtualized Network Function*

Capítulo 1

Introdução

A tecnologia de corrente de blocos é originalmente proposta como uma estrutura de dados distribuída e pública para processar transações de moedas digitais entre usuários sem a necessidade de uma autoridade centralizada [1]. Hoje, as tecnologias de corrente de blocos de primeira geração, baseadas no Bitcoin [1], e de segunda geração, baseadas nos contratos inteligentes do Ethereum [2] já revolucionaram o mundo atual ao criar uma camada de confiança para transferências seguras de ativos e execução segura de contratos. Esta tecnologia funciona como um livro-razão distribuído (*Distributed Ledger Technology* – DLT) envolvendo múltiplos participantes independentes e conectados através de comunicações par-a-par (*peer-to-peer* – P2P). A corrente de blocos em si provê como principais características i) a desintermediação, através do uso de protocolos de consenso que eliminam a necessidade de uma autoridade centralizada; ii) a imutabilidade dos dados registrados, através de técnicas criptográficas de garantia de integridade e replicação dos dados; e iii) o não-repúdio das transações registradas na corrente de blocos, através do uso de assinaturas e criptografia de chaves assimétricas. As diversas aplicações de correntes de blocos em criptomoedas e sistemas financeiros já atingiram um capital de mercado global de mais de 800 bilhões de dólares em 2018 [3], que corresponde a metade do PIB brasileiro em 2018 [4].

No entanto, a corrente de blocos pode ser utilizada na Internet do Futuro para prover confiança distribuída mesmo quando não há troca de moedas ou valores. A Internet do Futuro interconectará bilhões de dispositivos através de sistemas construídos a partir de milhares de serviços distribuídos. Aglomerados com um enorme número de máquinas físicas e virtuais manipularão grandes quantidades de dados gerados a partir de inúmeros sensores e atuadores da Internet das Coisas (*Internet of Things* - IoT), gerando um volume de informações jamais visto. A Internet será constituída de fatias de rede (*network slices*) adaptadas a cada tipo de aplicação, que serão criadas, alteradas e destruídas de maneira ágil e escalável para oferecer alta disponibilidade com baixo custo de operação. Esta Internet de nova geração,

baseada na tecnologia de Virtualização de Funções de Rede (*Network Function Virtualization* - NFV) e Redes Definidas por Software (*Software-Defined Networking* - SDN), será a essência das redes móveis de quinta geração (5G) e das cidades inteligentes (*smart cities*).

No novo paradigma 5G, um dos desafios fundamentais de pesquisa é garantir o provisionamento seguro de serviços em um ambiente de nuvem onde múltiplos inquilinos e usuários sem confiança mútua compartilham recursos computacionais. Uma forma de garantir a confiança de forma distribuída e permitir a análise forense para identificação dos responsáveis pelos problemas é através do uso da tecnologia de corrente de blocos (*blockchain*). A corrente de blocos, conhecida por suas aplicações em criptomoedas como o Bitcoin [1] e Ethereum [2], é uma tecnologia disruptiva que deve revolucionar não somente a tecnologia da informação como também a economia e a sociedade, permitindo uma maior descentralização do mundo atual. Pode-se prever um futuro no qual a computação e a Internet associadas à tecnologia de livros-razão distribuídos permitirão uma computação planetária distribuída que executa aplicações e contratos inteligentes através de um consenso entre computadores sem nenhuma entidade intermediária. Esta confiança distribuída sem intermediários provida pela corrente de blocos permitirá múltiplas ofertas de funções virtualizadas de rede e seleção criteriosa entre diversos provedores de infraestrutura, produzindo uma concorrência sem precedentes na área de telecomunicações. Em especial, as correntes de blocos atenderão a diversas aplicações, desde redes veiculares tolerantes a atraso até serviços críticos como saúde eletrônica (*e-Health*) [5, 6], cidades inteligentes (*smart cities*) [7–9] e redes elétricas inteligentes (*smart grids*) [10]. Por estes motivos, as tecnologias baseadas em corrente de blocos são consideradas tanto pela indústria como pela academia como a tecnologia mais disruptiva dos últimos tempos [11–13].

A inovação fundamental das correntes de blocos é prover, de forma descentralizada e sem intermediários, confiança entre um grupo de participantes. Para isso, o sistema deve ser capaz de obter consenso[1] entre os participantes para incorporar novos blocos à corrente. A proposta do Bitcoin, a primeira criptomoeda proposta por Satoshi Nakamoto [1] em 2008, resolve o problema do consenso através da prova de trabalho (*Proof-of-Work* – PoW), um protocolo baseado em competição entre os participantes para definir o líder da rodada de consenso. Todos os participantes tentam encontrar a solução para um desafio computacional baseado em funções resumo (*hash*) e o primeiro participante que resolver o desafio adquire o direito de adicionar o próximo bloco à corrente. No entanto, o crescimento do Bitcoin eviden-

[1]Consenso é um tipo de acordo produzido por consentimento entre todos os membros de um grupo. O consenso se estabelece quando dois ou mais membros chegam a um ponto comum de decisão durante uma negociação.

cia as limitações, gargalos de desempenho e problemas de sustentabilidade da prova de trabalho, como: i) consumo de energia excessivo; ii) baixa vazão de transações; e iii) tendência de centralização em participantes com maior poder computacional. A latência de consenso, de aproximadamente uma hora, e a vazão de transações, de até 7 transações por segundo, consistem em um desafio para atender às necessidades dos sistemas atuais [14, 15]. A baixa vazão do Bitcoin fica ainda mais evidente quando se compara o número de transações por segundo processadas pelas grandes companhias de cartões de crédito, que chegam a 2.000 transações por segundo em média e até 56.000 transações em pico, com tempos de resposta da ordem de segundos [16]. O processo de mineração da prova de trabalho no Bitcoin gera um consumo insustentável de energia sem retorno proporcional, atingindo mais de 70 TWh [17] por ano. Para se ter uma ideia, esse valor é mais de quatro vezes maior que os cerca de 15TWh de energia gerada pelas usinas nucleares de Angra [18]. O gasto energético para processar uma única transação no Bitcoin é suficiente para abastecer uma residência média brasileira durante três meses [17, 19].

Em resposta às limitações de desempenho do consenso baseado em prova de trabalho, diversos protocolos de consenso foram propostos como alternativas ao protocolo do Bitcoin. Dentre eles, destacam-se: i) protocolos alternativos baseados em prova, como a prova de posse (*Proof-of-Stake* – PoS) [20, 21]; ii) protocolos baseados em quórum, como os protocolos clássicos tolerantes a falhas de parada (e.g. RAFT [22] e Paxos [23]) e protocolos tolerantes a falhas bizantinas (e.g. PBFT [24] e BFT-Smart [25]); iii) protocolos híbridos, como o Casper FFG [26] e Tendermint [27]; e protocolos baseados em grafos acíclicos direcionados (*Directed Acyclic Graph* – DAG), como o IOTA [28]. Cada protocolo e cada categoria possui características específicas de desempenho e escalabilidade que devem ser consideradas para cada caso de uso.

A primeira parte desta dissertação discute as vantagens, desvantagens e características de cada protocolo de consenso, e busca clarificar a decisão de qual protocolo utilizar em cada caso. Esta primeira parte se baseia em trabalhos anteriores tais como artigos publicados em eventos nacionais e internacionais e, principalmente, os minicursos apresentados no Simpósio Brasileiro de Redes de Computadores "Segurança na Internet do Futuro: Provendo Confiança Distribuída através de Correntes de Blocos na Virtualização de Funções de Rede" [29] e "Blockchain e a Revolução do Consenso sob Demanda" [30]. A segunda parte do trabalho realizado foca na aplicação de correntes de blocos em redes virtualizadas e recebeu o prêmio de melhor artigo do II Workshop em Blockchain do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (WBlockchain SBRC 2019) com o artigo "Provendo uma Infraestrutura de Software Fatiada, Isolada e Segura de Funções Virtuais através da Tecnologia de Corrente de Blocos". A dissertação completa representa um

conjunto de publicações principais, listadas a seguir em ordem temporal:

- Rebello, G. A. F., Camilo, G. F., Silva, L. G. C., Souza, L. A. C., Guimarães, L. C. B., Alchieri, E. A. P., Greve, F. G. P. and Duarte, O. C. M. B. - "Correntes de Blocos: Algoritmos de Consenso e Implementação na Plataforma Hyperledger Fabric", in 38º Jornada de Atualização em Informática (JAI) do XXXIX Congresso da Sociedade Brasileira de Computação (CSBC 2019), Belém, Brazil, July 2019;

- Rebello, G. A. F., Alvarenga, I. D., Sanz, I. J. and Duarte, O. C. M. B. - "BSec-NFVO: A Blockchain-based Security for Network Function Virtualization Orchestration", in IEEE International Conference on Communications (ICC 2019), Shanghai, China, May 2019;

- Rebello, G. A. F., Camilo, G. F., Silva, L. G. C., Guimarães, L. C. B., Souza, L. A. C., Alvarenga, I. D. and Duarte, O. C. M. B. - "Providing a Sliced, Secure, and Isolated Software Infrastructure of Virtual Functions Through Blockchain Technology", in IEEE International Conference on High Performance Switching and Routing (HPSR 2019), Xi'An, China, May 2019;

- Rebello, G. A. F., Camilo, G. F., Silva, L. G. C., Guimarães, L. C. B., Souza, L. A. C., Alvarenga, I. D. and Duarte, O. C. M. B. - "Provendo uma Infraestrutura de Software Fatiada, Isolada e Segura de Funções Virtuais através da Tecnologia de Corrente de Blocos", in II Workshop em Blockchain: Teoria, Tecnologias e Aplicações (WBlockchain SBRC 2019), Gramado, Brazil, May 2019. Best paper award;

- Rebello, G. A. F., Camilo, G. F., Silva, L. G. C., Souza, L. A. C., Guimarães, L. C. B., and Duarte, O. C. M. B. - "Segurança na Internet do Futuro: Provendo Confiança Distribuída através de Correntes de Blocos na Virtualização de Funções de Rede", in Minicursos do XXXVII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2019), Gramado, Brazil, May 2019;

- Rebello, G. A. F., Alvarenga, I. D., Sanz, I. J., e Duarte, O. C. M. B. "SINFONIA: Gerenciamento Seguro de Funções Virtualizadas de Rede através de Corrente de Blocos", em I Workshop em Blockchain: Teoria, Tecnologias e Aplicações (WBlockchain - SBRC'2018). Campos do Jordão, Brasil. Best paper award.

O restante desta dissertação é organizado em cinco capítulos. O Capítulo 2 apresenta os fundamentos da tecnologia de corrente de blocos e o problema do

consenso em corrente de blocos. O Capítulo 3 discute os diversos protocolos de consenso em corrente de blocos, apresentando os protocolos baseados em prova e tolerantes a falhas de paradas e bizantinas. O Capítulo 4 apresenta uma proposta de aplicação de corrente de blocos para virtualização de redes. O Capítulo 5 discute os trabalhos relacionados. Por fim, o Capítulo 6 conclui a dissertação e discute as perspectivas futuras.

Capítulo 2

Correntes de Blocos e Consenso

Neste capítulo, são apresentados os tópicos fundamentais para melhor compreensão do trabalho realizado. Os tópicos abordados aqui são o problema do gasto duplo, a corrente de blocos, propriedades e categorias de corrente de blocos, modelos de rede, tipos de consenso e classes de protocolos de consenso.

2.1 O Problema do Gasto Duplo

Durante as últimas décadas, engenheiros, desenvolvedores, criptólogos e profissionais de tecnologia da informação procuraram resolver o problema do gasto duplo (*double spending problem*) para permitir a criação de uma moeda digital descentralizada [31]. O gasto duplo ocorre quando um mesmo ativo é utilizado mais de uma vez em diferentes transações de um sistema, e é especialmente importante em trocas de ativos digitais, que podem ser facilmente replicados. Ainda que comumente postulado para o caso de moedas digitais, o problema estende-se a qualquer tipo de recurso digital único, como endereços IP, domínios, registros de identidade, propriedade intelectual, recursos computacionais, serviços, etc.

A Figura 2.1 ilustra um breve enunciado do problema do gasto duplo. No exemplo, suponha que o usuário A deseja transferir moedas para o usuário B. Se o usuário A e o usuário B utilizam dinheiro em espécie, o usuário A deve ceder suas moedas ao usuário B e não possuirá mais as moedas após a transação. Porém, se uma moeda digital for utilizada, é necessária uma maneira de garantir que o usuário A gastou as moedas, pois moedas digitais são representações em bits que podem ser facilmente duplicadas. Neste caso, o usuário A pode enviar um arquivo digital com o valor desejado ao usuário B enquanto mantém uma cópia local do arquivo. Se o usuário A ainda mantiver uma cópia, ele pode enviá-la a uma terceira pessoa, o usuário C, realizando um gasto duplo.

Tradicionalmente, a prevenção do gasto duplo é realizada através da centralização em uma terceira entidade com poderes de autoridade, que utiliza regras

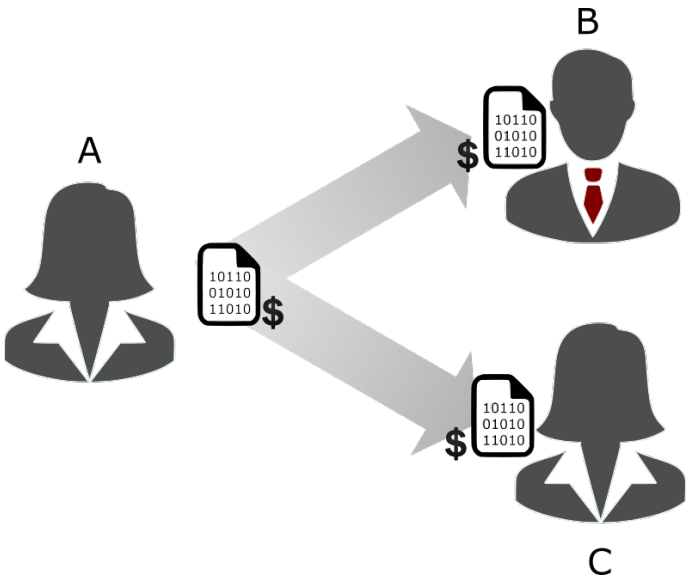


Figura 2.1: Exemplo do problema do gasto duplo no qual a mesma representação digital de uma moeda é replicada e gasta duas vezes.

de negócio para autorizar as transações e verificar se as moedas envolvidas foram gastas [32]. Essa abordagem é normalmente utilizada em bancos, companhias de crédito, plataformas de vendas *online* e, no caso de ativos na Internet, através de organizações como a *Internet Assigned Numbers Authority* (IANA)[1]. Porém, a centralização implica que todos os participantes do sistema possuam total confiança na autoridade central, que pode cobrar taxas, limitar o volume de transações e controlar a rede de forma arbitrária [1, 32]. Além disso, uma falha na autoridade central pode interromper todas as transações do sistema por tempo indeterminado. O modelo centralizado, portanto, cria um ponto único de falha na rede, impactando tanto a disponibilidade quanto a confiança no sistema.

2.2 Criptomoedas e Corrente de Blocos

O conceito de moeda digital descentralizada, doravante referida como criptomoeda, é o principal propulsor de aplicações envolvendo trocas de outros ativos. Durante décadas, procurou-se um mecanismo para viabilizar uma criptomoeda totalmente funcional. Em 1983, David Chaumian introduziu o primeiro protocolo anônimo para criação de criptomoedas utilizando o conceito de assinatura cega (*blind signature*) [33]. A assinatura cega provê privacidade às transferências de moedas, mas depende fortemente de entidades centralizadas para realizar as assinaturas. Em 1998, Wei Dai propôs *b-money*, uma criptomoeda que introduz a ideia de criar valor monetário através da resolução de desafios computacionais e com consenso

[1]Entidade que distribui e garante a unicidade do endereço IP, disponível em https://www.iana.org/

descentralizado [34]. No entanto, a proposta possuía poucos detalhes sobre a implementação do consenso, dificultando sua adoção. Em 2002, Adam Back propôs formalmente *Hashcash*, um algoritmo de prova de trabalho (*Proof of Work* - PoW) baseado em funções resumo (*hash*) criptográficas para prevenção de *spam* de e-mails e ataques de negação de serviço [35]. Em 2005, Hal Finney desenvolveu o conceito de provas de trabalho reutilizáveis (*Reusable Proof of Work* - RPoW) em um sistema que reúne os fundamentos do *b-money* e os desafios computacionalmente custosos do *Hashcash* [36]. No entanto, a proposta também dependia de entidades confiáveis para ser implementada. Em 2008, Satoshi Nakamoto[2] propôs e implementou pela primeira vez o Bitcoin, uma criptomoeda totalmente descentralizada, combinando a prova de trabalho com uma nova e revolucionária estrutura de dados organizada em blocos de transações: a corrente de blocos (*blockchain*) [1].

2.2.1 Corrente de Blocos

A corrente de blocos, conhecida por suas aplicações em criptomoedas tais como como o Bitcoin [1] e Ethereum [2], é uma tecnologia disruptiva que deve revolucionar não somente a tecnologia da informação como também a economia e a sociedade, permitindo, com a eliminação das entidades intermediárias, uma maior descentralização do mundo atual. O principal diferencial de uma corrente de blocos é ser uma estrutura de dados replicada que garante confiabilidade através de um protocolo de consenso sem necessidade de uma autoridade central. Pela primeira vez, duas partes que não confiam uma na outra ou mesmo que não se conhecem podem trocar ativos sem um intermediário. A corrente de blocos substitui o modelo centralizado por um modelo colaborativo no qual a maior parte da rede deve concordar sobre todas as decisões. A utilização da corrente de blocos elimina a necessidade de autoridades centralizadas e permite tomar decisões coletivas, potencialmente eliminando governos, bancos, agências de crédito e fornecendo poder à população. A corrente de blocos é aplicável em todo ambiente distribuído no qual os participantes não possuem confiança mútua e são incapazes de entrar em acordo sobre uma autoridade centralizada que governe todos os procedimentos sensíveis da rede [37]. Suas propriedades, discutidas com maior profundidade na Seção 2.2.3, garantem a auditabilidade, a imutabilidade e o não repúdio de todas as transações realizadas por participantes da rede, e são chave para a criação de um ambiente seguro e escalável.

A corrente de blocos é uma tecnologia de livro-razão distribuído (*Distributed Ledger Technology* - DLT) que utiliza máquinas independentes, denominadas nós mineradores[3], para gravar, compartilhar e sincronizar transações em um sistema

[2]Satoshi Nakamoto é um pseudônimo utilizado pelo criador ou criadores da moeda virtual Bitcoin. Sua identidade real é desconhecida.

[3]Em modelos de consenso sem recompensa por vencer um desafio, os nós mineradores são

de controle descentralizado sobre uma rede par a par (*peer-to-peer* - P2P). Cada participante do consenso possui uma cópia local da corrente de blocos na qual pode verificar qualquer transação emitida na rede desde sua criação. A adição de blocos é realizada de maneira descentralizada através de um protocolo de consenso comum entre os participantes. Assim, um atacante que deseja realizar um gasto duplo precisa controlar uma parcela grande o suficiente da rede para propor, utilizando o protocolo de consenso, um bloco que oculte uma de suas transações [1, 38]. Na prova de trabalho proposta no Bitcoin [1], a parcela deve ser de mais de 50% do poder computacional da rede. O protocolo de consenso garante que as cópias da corrente de blocos são idênticas em qualquer participante do consenso. Todas as transações são assinadas por seus emissores através de criptografia de chaves assimétricas e, na maior parte das implementações, utiliza-se a chave pública como identificador na rede. A assinatura garante a propriedade de não repúdio de transações.

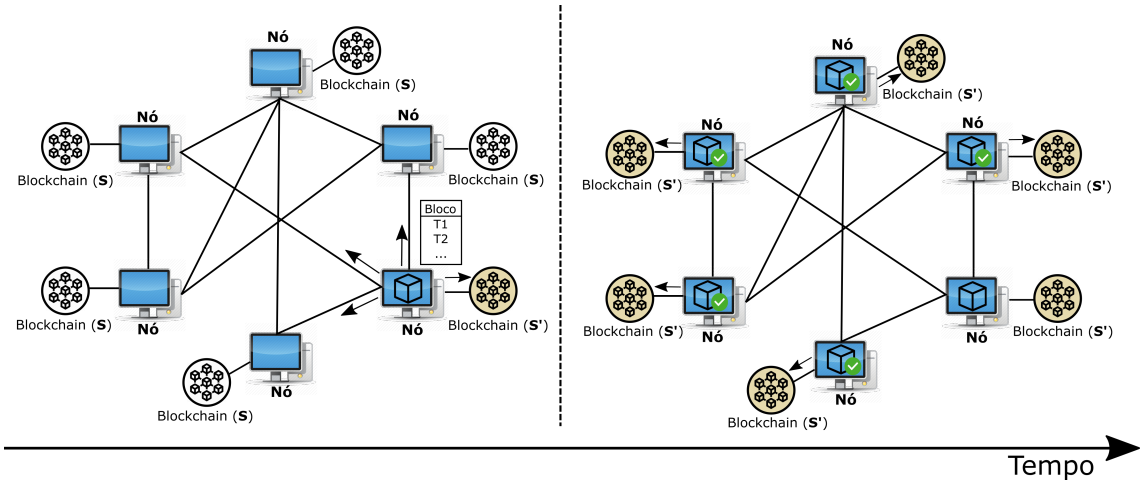


Figura 2.2: Atualização de uma corrente de blocos (*blockchain*) através de um protocolo de consenso genérico. Um líder do consenso propõe, em difusão, um novo bloco que muda o estado global de S para S' e os demais participantes verificam e adicionam independentemente o bloco proposto à corrente de blocos, replicando o novo estado global S' de maneira consistente.

A Figura 2.2 ilustra o funcionamento de uma corrente de blocos como um livro-razão distribuído através de um protocolo de consenso genérico. Um participante do consenso que possui o direito de alterar o estado atual S da corrente de blocos inicia uma rodada de consenso reunindo as transações recebidas de usuários em um novo bloco a ser proposto. A seguir, o bloco proposto é difundido na rede e validado localmente pelos demais participantes de acordo com políticas pré-definidas. Dependendo do protocolo adotado, o bloco proposto pode ser adicionado imediatamente, de forma assíncrona, à corrente de blocos do participante que propôs o bloco, ou

chamados de participantes do consenso, de validadores ou simplesmente de nós. Esta dissertação utiliza o termo "participantes do consenso"como o termo genérico que se aplica a todos os consensos.

de forma síncrona, através de troca de mensagens, em todos os participantes ao fim da rodada. A figura mostra o caso de um protocolo no qual o estado S é alterado para S' no participante que propôs o bloco antes da difusão. Ao receber o bloco proposto, cada participante do consenso valida o bloco independentemente e, caso aprove o bloco, o adiciona à corrente de blocos e atinge ao estado S'. O algoritmo de validação de um bloco e de cada transação deve ser acordado previamente por todos os participantes. Todo protocolo de consenso possui um mecanismo de atualização para obter o estado mais recente e corrigir possíveis discrepâncias de estado resultantes de blocos não aprovados ou perda de conectividade, garantindo a consistência da corrente de blocos em toda a rede. Os protocolos de consenso e suas premissas são melhor discutidos na Seção 3.

A Figura 2.3 mostra o fluxo de uma transação em um sistema de troca de ativos digitais através de uma transação simplificada de Bitcoin (BTC) [1]. Para enviar ou receber ativos, um usuário deve utilizar um par de chaves assimétricas para assinar transações. Por exemplo, para transferir 0,5 BTC ao usuário B, o usuário A cria uma transação contendo: i) uma entrada com o seu endereço e o total de BTC que possui; ii) uma saída com o endereço de o usuário B e o valor que deseja transferir; e iii) uma saída com seu próprio endereço e o valor de troco. A seguir, o usuário A utiliza um algoritmo criptográfico para assinar a transação, criando uma transação assinada, e a envia com sua chave pública em difusão para a rede *peer-to-peer* (P2P) na qual estão os participantes do consenso. A partir deste momento, qualquer participante do consenso pode aplicar o algoritmo de validação da rede para verificar a autenticidade da transação e se há conflito com outra transação já registrada na corrente de blocos. Transações inválidas são descartadas imediatamente. Caso a validação ocorra com sucesso, o participante do consenso adiciona a transação a um novo bloco a ser proposto na próxima rodada do protocolo de consenso. A maior parte dos sistemas de troca de ativos provê programas que gerenciam chaves, armazenam endereços e emitem transações assinadas de forma transparente ao usuário. Estes programas são amplamente conhecidos como "carteiras (*wallets*)".

2.2.2 Estruturas de Dados

A estrutura de dados da corrente de blocos proposta por Nakamoto como parte da criptomoeda Bitcoin [1] é uma lista encadeada de estruturas de dados menores, conhecidas como blocos. Cada bloco está conectado a seu antecessor através de uma função resumo (*hash*) criptográfica, criando uma corrente de blocos conforme mostra a Figura 2.4. A corrente de blocos funciona como um livro-razão (*ledger*), ou registro permanente (*log*), de blocos ordenados no tempo. Cada bloco na corrente possui um cabeçalho contendo o valor resultante de uma função resumo (*hash*) criptográfica

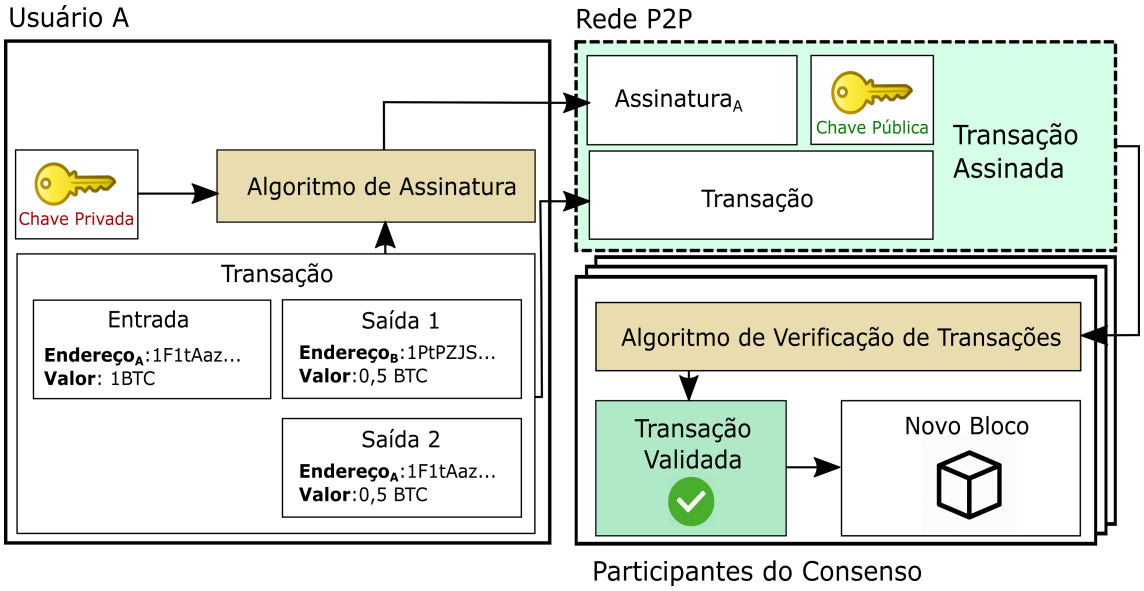


Figura 2.3: Fluxo de uma transação em uma corrente de blocos. O usuário A difunde na rede par a par uma transação assinada que pode ser verificada e adicionada a um bloco por qualquer participante do consenso.

do bloco antecessor, e uma seção de conteúdo que armazena dados relevantes a uma aplicação. A única exceção a esta regra é o bloco inicial, o gênesis, que por definição não possui bloco anterior. A utilização de uma sequência de funções resumo criptográficas, cujas saídas diferem drasticamente após a alteração de qualquer bit na entrada, implica que, se um atacante altera um bloco, todos os blocos seguintes devem ser reconstruídos. Assim, a adição de blocos torna cada vez mais custoso alterar a corrente e, como a corrente de blocos é replicada em cada participante do consenso, um atacante deve invadir todos os participantes do consenso e recriar cada corrente de blocos para modificar uma transação passada. A replicação e o encadeamento através de funções *hash* garante, portanto, a imutabilidade de todos os blocos da corrente de blocos.

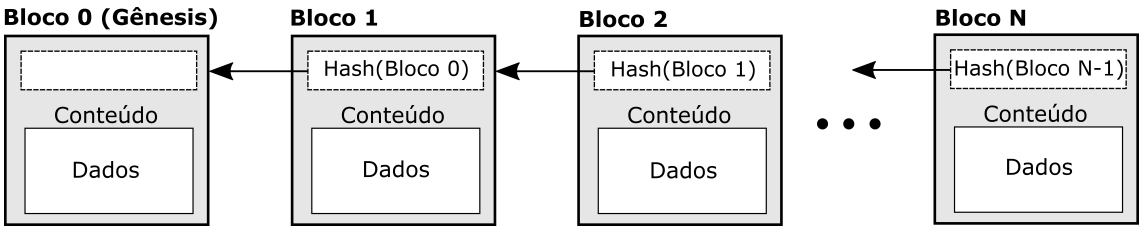


Figura 2.4: Estrutura de uma corrente de blocos, que associa cada bloco ao bloco anterior e garante a integridade das transações de um bloco através de uma função resumo (*hash*) criptográfica.

Na literatura, propostas de aplicações utilizam o conteúdo do bloco para diferentes propósitos. Nakamoto propõe originalmente registrar transações bancárias para criar uma criptomoeda [1]. Wood *et al.*, Frantz *et al.* e Androulaki *et al.* propõem

utilizar correntes de blocos para armazenar e executar contratos inteligentes através de uma máquina virtual distribuída [2, 39, 40]. Wilkinson *et al.* propõem um sistema baseado em correntes de blocos para prover armazenamento descentralizado em nuvem com privacidade [41]. Ali *et al.* propõem o uso de correntes de blocos para registrar nomes de domínio [42]. Azaria *et al.* propõem registrar informações de fichas médicas [43]. Lee *et al.* propõem registrar votos para criar um sistema de votação descentralizado [44]. Christidis e Hun *et al.* propõem rastrear dispositivos de Internet das Coisas (*Internet of Things* - IoT) através do armazenamento de informações na corrente de blocos [45, 46]. Bozic *et al.* e Alvarenga *et al.* propõem registrar informações de máquinas virtuais e funções virtualizadas de rede na corrente de blocos [47, 48].

O principal conteúdo de uma corrente de blocos, no entanto, são as transações. Uma transação representa uma ação atômica a ser armazenada na corrente de blocos que transfere um ativo entre duas entidades. A transação possui quatro componentes principais: i) um remetente, que possui um ativo que deseja transferir e que emite a transação; ii) um destinatário que receberá o ativo que se deseja transferir; iii) o ativo que será transferido; e iv) uma assinatura digital para comprovar a autenticidade e evitar o não repúdio do remetente da transação. O uso conjunto de transações assinadas e da propriedade de imutabilidade da corrente de blocos cria um sistema que funciona como um livro-razão distribuído. As transações no sistema são ordenadas dentro de cada bloco e podem ser verificadas por qualquer nó participante da rede, desde o bloco inicial da corrente de blocos. Pela propriedade de chaves criptográficas assimétricas, a assinatura do *hash* da transação usando a chave privada do remetente provê autenticidade, não repúdio e integridade à transação, fornecendo maior segurança à rede. Ainda, é possível obter pseudo-anonimidade utilizando chaves públicas ou endereços únicos, que não revelam as identidades pessoais, para identificar origens e destinatários. A privacidade de transações, desejada em casos onde apenas o emissor e destinatário devem poder consultar a transação [40], pode ser garantida encriptando a transação com a chave pública do destinatário.

2.2.3 Propriedades de Corrente de Blocos

A corrente de blocos possui propriedades que proveem benefícios às aplicações e sistemas baseados nesta tecnologia. As principais propriedades da corrente de blocos são [37, 49, 50]:

- **Descentralização** - Os sistemas baseados em correntes de blocos executam de maneira distribuída, servindo-se de um consenso entre os participantes da rede para definir o próximo estado e garantir consistência do estado global. Não há uma entidade centralizadora;

- **Desintermediação** - A corrente de blocos elimina a necessidade de um intermediário confiável para a troca de ativos. Os ativos podem ser trocados diretamente pela rede entre origem e destinatário, e a confiança é estabelecida através de consenso e criptografia;

- **Imutabilidade**. Os dados armazenados em uma corrente de blocos são imutáveis. Não é possível modificar ou recriar qualquer dado incluído na corrente de blocos de forma retroativa. Toda atualização na corrente de blocos é realizada de forma incremental;

- **Irrefutabilidade** - Os dados são armazenados na corrente de blocos em forma de transações assinadas, que não podem ser alteradas devido à propriedade de imutabilidade da corrente de blocos. Portanto, o emissor de uma transação jamais pode negar sua existência;

- **Transparência** - Todos os dados armazenados na correntes de bloco são acessíveis por todos os participantes da rede. Em correntes de blocos públicas, como o Bitcoin [1] e o Ethereum [2], as transações são abertas para qualquer usuário com acesso à Internet;

- **Auditabilidade** - Como consequência da transparência, todo participante pode verificar, auditar e rastrear os dados inseridos na corrente de blocos para encontrar possíveis erros ou comportamentos maliciosos. No caso de correntes de blocos federadas, o processo de auditagem pode responsabilizar um malfeitor na rede;

- **Disponibilidade** - As correntes de blocos são estruturas replicadas em cada participante da rede e, portanto, a disponibilidade do sistema é garantida mesmo sob falhas, devido à redundância de informações;

- **Anonimidade** - Em geral, os usuários e participantes de uma corrente de blocos são identificados por chaves públicas ou identificadores únicos que preservam suas identidades. Ainda, é possível utilizar uma chave pública em cada transação, evitando a rastreabilidade do usuário e conferindo um grau a mais de anonimidade.

Além das propriedades citadas, a corrente de blocos também pode prover privacidade, ao custo da transparência e auditabilidade. Em algumas implementações de corrente de blocos, os dados inseridos na corrente são criptografados com uma chave pública, evitando que todos os participantes possam ver o conteúdo da transação [40, 51]. Nesses casos, a transparência e a auditabilidade limitam-se a um ou mais participantes que detêm a chave privada correspondente e podem decriptar a transação criptografada.

2.2.4 Categorias de Corrente de Blocos

A utilização de correntes de blocos constrói uma máquina de estados replicada que garante a consistência do sistema e um estado global compartilhado entre todos os participantes da rede. No entanto, a corrente de blocos deve ser configurada para atender às demandas de cada aplicação através de decisões de projeto. Através da taxonomia proposta na literatura [45, 49, 50] e das principais aplicações de correntes de blocos [1, 2, 40], é possível categorizar as correntes de blocos em três tipos: correntes de blocos públicas, correntes de blocos federadas e correntes de blocos privadas. Os tipos de correntes de blocos diferenciam-se nos seguintes aspectos:

- **Permissão de escrita** - Correntes de blocos públicas, ou não-permissionadas, não impõem restrições de permissão a participantes da rede que desejam tornar-se participantes do consenso e adicionar blocos através do protocolo de consenso [1, 2]. Em correntes de blocos federadas e privadas, também conhecidas como permissionadas, um novo participante do consenso deve obter permissão de escrita dos demais participantes do consenso através, respectivamente, de consenso ou de uma decisão centralizada;

- **Permissão de leitura** - Transações em uma corrente de blocos pública são publicamente acessíveis. Em correntes de blocos federadas, o acesso pode ser público ou restrito aos membros das federações, e varia para cada aplicação [40, 51]. Em correntes privadas, a leitura só é permitida às entidades autorizadas;

- **Modelo de consenso** - Em correntes públicas, o protocolo de consenso considera alterações arbitrárias no conjunto de participantes, resultantes da liberdade de escrita, e implica no uso de protocolos baseados em prova com alto custo computacional [1, 2, 42]. Em correntes federadas, as restrições de permissão de escrita permitem adotar protocolos de consenso mais eficientes que se baseiam em trocas de mensagens entre participantes do consenso [40, 48]. Em correntes privadas, o protocolo de consenso é opcional e as decisões na rede podem ser tomadas de forma centralizada;

- **Incentivo** - Correntes de blocos públicas, devido principalmente ao alto custo dos protocolos de consenso, adotam mecanismos de incentivo para recompensar e estimular os participantes a gastar recursos computacionais para propor novos blocos. Em correntes de blocos federadas e privadas, o incentivo é opcional pois os participantes normalmente têm interesse direto na manutenção da corrente de blocos e adotam protocolos menos custosos;

- **Eficiência** - Os protocolos baseados em prova e as grandes quantidades de participantes tornam as correntes de blocos públicas pouco eficientes. Cor-

Tabela 2.1: Comparação entre diferentes categorias de correntes de blocos.

	Pública	**Federada**	**Privada**
Permissão de Escrita	Não-permissionada	Permissionada	Permissionada
Permissão de Leitura	Pública	Pública ou Privada	Privada
Modelo de Consenso	Baseado em prova	Baseado em troca de mensagens	Opcional
Incentivo	Obrigatório	Opcional	Opcional
Eficiência	Baixa	Média	Alta
Confiabilidade	Alta	Média	Baixa

rentes de blocos federadas possuem eficiência maior que as correntes públicas, mas a tomada colaborativa de decisões implica em menor eficiência que as correntes de blocos privadas [52];

- **Confiabilidade** - A grande descentralização das correntes de blocos públicas dificulta que um atacante domine uma parcela significativa do consenso e confere maior confiabilidade à rede. No entanto, em correntes federadas e principalmente em correntes privadas, a menor quantidade de participantes do consenso pode trazer riscos à segurança da rede e torná-las mais suscetíveis a ataques de conluio e ataques *Sybil* [53].

Além das características citadas a anonimidade na rede é uma decisão de projeto fundamental que define a forma de identificar indivíduos na rede. A maior parte das aplicações de correntes de blocos utiliza uma chave pública ou um *hash* assinado como endereço [1, 40, 42, 43]. Este tipo de identificação provê autenticidade às transações e pseudo-anonimidade aos usuários ao dissociar o identificador de uma identidade real. A maior parte das carteiras automatiza a criação de um novo endereço para cada transação realizada, com objetivo de dificultar o comprometimento da anonimidade. Para casos em que deseja-se revelar a identidade de um usuário, é possível associar identificadores a pessoas através de certificados públicos ou dicionários registrados na própria corrente de blocos. Esta identificação é especialmente importante em aplicações que proveem auditabilidade e rastreabilidade de transações [44, 46–48].

2.3 Consenso em Correntes de Blocos

A corrente de blocos é um livro-razão distribuído que reúne uma lista incremental de registros controlada por múltiplas entidades que não possuem garantia de confiança mútua. Os participantes do sistema baseado em corrente de blocos comunicam-se através de uma rede par a par para construir o livro-razão distribuído de forma colaborativa. Portanto, um sistema baseado em corrente de blocos é essencialmente um sistema distribuído que está sujeito a falhas de conectividade, falhas de falta de energia e comportamentos maliciosos. É necessário adotar um protocolo

tolerante a falhas para garantir que todos os participantes atinjam o mesmo estado global de forma consistente. Assim, o protocolo de consenso é uma parte fundamental da corrente de blocos para construir um sistema robusto que provê segurança, confiabilidade, disponibilidade, auditabilidade e integridade aos usuários [45, 49, 52].

2.3.1 Consenso em Sistemas Distribuídos

Em termos genéricos, consenso é o processo pelo qual um conjunto de participantes do consenso[4] independentes em um sistema distribuído atinge uma decisão comum a todo o sistema. Para isto, deve necessariamente haver troca de informação entre os participantes através de troca de mensagens ou memória compartilhada [54]. Os participantes do consenso de sistemas baseados em correntes de blocos normalmente estão fisicamente distantes um dos outros, o que dificulta o uso de memória compartilhada e, portanto, normalmente implementam algum mecanismo de troca de mensagens. As mensagens de uma rodada de consenso utilizam duas primitivas genéricas[5] [55]:

- *propose(P,x)*: usada para propor um novo valor x ao conjunto de participantes do consenso P. Em geral, o líder do consenso é responsável por emitir esta primitiva;
- *decide(y)*: usada para decidir sobre um valor y a partir da entrada recebida. Localmente, cada participante do consenso p recebe a entrada $\hat{x}$ e decide pelo valor $\hat{y}$ que, caso haja consenso, será igual ao valor final y.

Formalmente, o consenso através das duas primitivas citadas ocorre se e somente se as seguintes propriedades fundamentais forem satisfeitas sob a presença de falhas [23, 56, 57]:

- **Terminação (*termination*)** - Propriedade que determina que todo participante correto de consenso [6] decide por um valor;
- **Acordo (*agreement*)** - Propriedade que determina que todo participante correto de consenso decide pelo mesmo valor $\hat{y}$;
- **Validade (*validity*)** - Propriedade que determina que se todo participante de consenso recebe o valor x como proposta, então $y = x$;

[4]Este capítulo usa os termos nós (*node*), par (*peer*), computador, componente ou processo (*process*) como sinônimos de participante do consenso.

[5]Utiliza-se a notação $\hat{x}$ e $\hat{y}$ para denotar os valores locais ao participante do consenso p, em contraste aos valores x e y vistos por um agente externo ao sistema.

[6]Um participante correto de consenso é um participante que não está em estado de falha.

- **Integridade (*integrity*)** - Propriedade que determina que toda decisão $\hat{y}$ de um participante correto de consenso e o valor final y do consenso devem ter sido propostos por um participante correto do consenso.

A propriedade de terminação garante que todos os participantes corretos decidem por um valor, provendo vivacidade (*liveness*) ao consenso. A propriedade de acordo garante que este valor é idêntico em todos os participantes, e as propriedades de validade e integridade podem ser interpretadas como a corretude da decisão, garantindo que o valor decidido é o valor proposto pelos participantes corretos do consenso. Juntas, as propriedades de acordo, validade e integridade proveem a corretude (*safety*) do consenso. Os protocolos de consenso devem ter como objetivo garantir estas quatro propriedades a qualquer momento no sistema, mesmo sob a presença de falhas.

Schneider *et al.* definem dois elementos necessários para obter consenso entre processos distribuídos sob a presença de falhas [58]: i) uma máquina de estados replicada e determinística que armazena um estado global da rede; e ii) um protocolo de consenso que realiza transações de estado de forma descentralizada. Através destes dois elementos, é possível implementar a replicação de máquina de estados (*State Machine Replication* - SMR), uma técnica de tolerância a falhas que garante a consistência de um sistema distribuído.

Os principais desafios de projeto de corrente de blocos com alta disponibilidade são a tolerância a falhas e a coordenação entre os participantes. O teorema CAP (*Consistency*, *Availability*, *Partition*) prova que todo sistema distribuído apresenta um compromisso entre três propriedades [59]:

- **Consistência (*Consistency*)** - propriedade de um sistema distribuído no qual todos os nós possuem os dados mais atuais da rede;
- **Disponibilidade (*Availability*)** - propriedade do sistema que define o sistema está sempre operante, acessível e é capaz de responder corretamente a novas requisições;
- **Tolerância a partições (*Partition Tolerance*)** - propriedade do sistema distribuído de operar corretamente mesmo que um grupo de nós falhe. Se for possível haver comportamento malicioso dos nós, o sistema continua funcionando corretamente com alguns nós honestos, mesmo na presença de um grupo de nós maliciosos.

O teorema prova que é impossível para um sistema distribuído atingir plenamente todas as três propriedades [59]. Por isto, na prática, sistemas distribuídos privilegiam uma ou duas propriedades, sacrificando as demais. A replicação contribui para a tolerância a falhas. A consistência é obtida com o uso de protocolos de consenso

que garantem que todos os participantes possuem os mesmos dados. No Bitcoin e em diversas outras aplicações [1, 2, 42], a consistência é sacrificada em nome da disponibilidade e da tolerância a partições. Isto significa que a consistência não é obtida simultaneamente com as outras duas propriedades, mas sim gradativamente, com o tempo, à medida que os blocos são validados pelos nós da rede. Correntes de blocos baseadas em protocolos tolerantes a falhas bizantinas privilegiam fortemente a consistência, em detrimento da disponibilidade [40, 60–63].

2.3.2 Premissas de um Ambiente de Corrente de Blocos

A construção de um protocolo de consenso de correntes de blocos baseia-se em premissas fundamentais assumidas sobre o ambiente distribuído. As premissas devem cobrir os elementos essenciais do sistema, como o número máximo de falhas de participantes do consenso, o modelo de sincronia da rede, o modelo de falha dos participantes do consenso, o modelo de atacante, etc. e proveem os casos de exceção que o protocolo deve tratar. As premissas do ambiente impactam diretamente a complexidade de um protocolo de consenso através de um compromisso. A adoção de premissas fortes para gerar protocolos mais simples restringem a aplicação a cenários controlados e normalmente menos realistas. Premissas fracas são aplicáveis a mais casos reais, porém geram protocolos de alta complexidade devido à maior quantidade de casos de exceção. Uma premissa representa uma idealização do mundo real e deve ser avaliada constantemente para cada caso de implementação [49, 52, 64]. A seguir, apresenta-se uma lista não-exaustiva de três premissas fundamentais a todo sistema baseado em correntes de blocos: o modelo de falha, o modelo de sincronia de rede e a quantidade máxima de falhas toleradas no sistema.

O modelo de falha de uma corrente de blocos define o tipo de falha que pode ocorrer nos participantes. No modelo de falhas de parada ou desastrosas (*crash failures*), os participantes podem parar de responder às mensagens do consenso devido a panes, perda de conectividade ou quedas de energia [23, 65]. No modelo de falhas bizantinas, os participantes do consenso podem, além de não responder, responder às mensagens de forma arbitrária e/ou maliciosa para subverter o sistema [57, 66]. No modelo de falhas bizantinas, um processo falho pode exibir qualquer comportamento, incluindo panes, omissão de envios e entregas de mensagens, ou emissão deliberada de mensagens falsas. Portanto, a falha bizantina é um modelo mais robusto que engloba a falha de parada ou desastrosa, e, consequentemente, um protocolo que seja tolerante a falhas bizantinas é também tolerante a falhas desastrosas [13]. Devido à característica fundamental de desconfiança mútua entre os participantes de uma corrente de blocos, o modelo de falhas bizantinas é amplamente mais utilizado em sistemas baseados em correntes de blocos [1, 2, 60, 61]. No entanto, algumas

implementações em ambientes privados utilizam o modelo de falhas desastrosas e optam por prevenir comportamento malicioso através de outros mecanismos [40, 67]. Restringir o modelo de falha a falhas de parada ou desastrosas gera protocolos mais simples e não tolerantes a comportamento malicioso; utilizar o modelo de falhas bizantinas gera protocolos mais complexos e seguros [64].

A quantidade máxima de falhas toleradas define a confiabilidade do sistema baseado em corrente de blocos. O desenvolvedor de uma corrente de blocos deve decidir por um valor k para que, com n participantes independentes, no máximo $f < n/k$ participantes estejam em estado de falha simultaneamente, onde $k \in \{2, 3, ..., n\}$ e os demais $n - f$ participantes são corretos. Em correntes de blocos públicas, como o Bitcoin e Ethereum [1, 2], assume-se $k = 2$ e, então, até $n/2$ ou 50% dos participantes do consenso podem estar em estado de falha sem comprometer o funcionamento do sistema. Em implementações federadas e privadas baseadas em protocolos tolerantes a falhas bizantinas [40, 60, 61, 63] assume-se $k = 3$ e, então, a tolerância máxima é de até $n/3$ ou 33,3% da rede em estado de falha. No entanto, desenvolvedores de protocolos de consenso podem escolher suportar tolerâncias mais baixas para prover maior segurança em detrimento da robustez de uma rodada de consenso [60]. O percentual máximo de nós em falha é uma decisão de projeto e o Apêndice A.1 analisa com mais detalhes os impactos de cada percentual. O Apêndice ainda demonstra por que 33% é a tolerância máxima para protocolos de consenso baseados em acordo bizantino e discute os benefícios de tolerar menos falhas do que o máximo possível.

O modelo de sincronia de rede do sistema define o tipo de sincronia existente entre os participantes e impacta diretamente o funcionamento dos protocolos de consenso. A literatura apresenta três principais categorias de sincronia de rede [13, 64, 68]:

- **Redes síncronas**, nas quais existe garantia de entrega das mensagens do consenso sem atraso ou com atraso limitado e bem definido. Em específico, uma rede cujas mensagens chegam ao destino com atraso de até Δ é chamada de uma rede Δ-síncrona. Este modelo permite utilizar temporizadores (*timeouts*) para identificar falhas e é o que mais facilita o desenvolvimento de protocolos de consenso. No entanto, normalmente é uma premissa pouco realista, pois na prática deve haver a execução de um serviço de sincronização de relógios centralizado e boa conectividade de rede a todo momento;

- **Redes parcialmente síncronas**, nas quais existe garantia de entrega das mensagens do consenso com atraso limitado, porém indeterminado [69]. Dentre os diversos subtipos de sincronia parcial de rede, destacam-se dois para aplicação em correntes de blocos: i) sincronia fraca (*weak synchrony*), na qual assume-se que o atraso Δ da mensagem não crescerá mais do que o horizonte

temporal, e assim a rede se estabilizará com o tempo; e ii) sincronia eventual ou após um tempo (*eventual synchrony*), que assume que a rede se comportará como uma rede Δ-síncrona após um período indeterminado porém finito de tempo. Nas duas formas, o protocolo de consenso considera que a rede se comportará como uma rede síncrona e estável caso o horizonte de tempo seja suficientemente longo. Este é o modelo majoritariamente utilizado pelos protocolos de consenso devido à sua aplicabilidade em redes reais, sobretudo as redes com garantia de entrega baseadas na pilha TCP/IP. O uso do protocolo TCP na camada de transporte garante a entrega das mensagens com atraso variável devido às possíveis retransmissões de pacotes e aos múltiplos caminhos percorridos, essencialmente provendo uma rede parcialmente síncrona.

- **Redes assíncronas**, nas quais existe garantia de entrega das mensagens do consenso, porém não há nenhuma garantia sobre o atraso. O atraso de entrega é indefinido, ilimitado e possivelmente infinito. Este é o modelo mais genérico e que engloba todas as redes existentes. No entanto, a maior dificuldade deste modelo é identificar participantes em falha, uma vez que o atraso de entrega é ilimitado. A assincronia aumenta consideravelmente a complexidade do protocolo de consenso e cria casos de exceção que podem impossibilitar o consenso mesmo sob a presença de uma falha [56].

2.3.3 Modelos de Consenso em Corrente de Blocos

Um dos principais desafios de consenso em sistemas distribuídos é o resultado de não-garantia do consenso conhecido como FLP[7]. O resultado prova que, em um sistema com modelo assíncrono e com n participantes identificados, o consenso não possui solução determinística mesmo na presença de uma única falha de parada [56]. Isto significa que, dado tempo suficiente e uma única falha de participante, os demais participantes não obtêm consenso e o sistema fica estagnado em um estado global.

Há duas abordagens possíveis para contornar o problema FLP de obtenção de consenso em correntes de blocos: relaxar as garantias do consenso, comprometendo a consistência do sistema, ou relaxar o modelo de sincronia, assumindo pelo menos o modelo eventualmente síncrono. Como consequência das duas abordagens, o problema do consenso pode ser tratado, respectivamente, de maneira probabilística ou determinística.

Consenso probabilístico. O consenso probabilístico assume a consistência eventual da rede, isto é, que, na ausência de novas transações, todo participante eventualmente atinge o mesmo estado global [70, 71]. Isto representa um enfraquecimento da propriedade de acordo, uma vez que os participantes podem divergir sobre

[7]O teorema recebe esta sigla em homenagem a seus autores: Fisher, Lynch e Patterson.

os dados mais recentes até que a consistência seja atingida. Este comportamento provê *finalidade probabilística* ao sistema, na qual a finalidade de um bloco[8] depende das confirmações por blocos seguintes e de mecanismos de desempate quando há mais de uma proposta de bloco simultaneamente. A propriedade de terminação, entretanto, é garantida.

Consenso determinístico. Em redes estáveis, como redes permissionadas síncronas ou parcialmente síncronas, é possível obter consistência determinística, ou consistência forte, apoiando-se nas garantias de entrega de mensagens da rede. Este tipo de consenso utiliza os protocolos de consenso clássicos baseados em quórum e tolerantes a falhas de parada ou a falhas bizantinas [23, 24] para prover *finalidade determinística* ao sistema. Neste caso, a finalidade de um bloco é garantida assim que o bloco é aprovado por consenso, eliminando a necessidade de mecanismos de desempate. O acordo entre participantes é garantido, mas se a rede não oferecer as condições de sincronia e estabilidade necessárias para que haja consenso, a terminação do protocolo de consenso é comprometida. Assim, o consenso determinístico sempre assegura a consistência do sistema, possivelmente sacrificando seu progresso.

Este capítulo abordou os princípios fundamentais de correntes de blocos e discutiu os detalhes do problema do consenso em sistemas baseados em correntes de blocos. A seguir, a dissertação apresenta diversos protocolos de consenso de acordo com as premissas de ambiente e os tipos de consenso detalhados neste capítulo.

[8]Um bloco finalizado é um bloco que não está mais sujeito a abandono por mecanismos de desempate.

Capítulo 3

Protocolos de Consenso

Este capítulo apresenta e compara os principais protocolos de consenso em corrente de blocos[1]. Dentre as categorias de protocolos de consenso existentes, destacam-se os protocolos baseados em prova, os protocolos baseados em quórum, os protocolos híbridos e os protocolos baseados em grafos acíclicos direcionados, que propõem uma nova estrutura de dados e não se enquadram nas demais categorias.

3.1 Consenso Probabilístico: Protocolos Baseados em Prova

Os protocolos de consenso mais comuns em sistemas públicos e não-permissionados são os baseados em algoritmos de prova, nos quais um participante que propõe um bloco deve apresentar uma prova de que pode liderar o consenso [1, 73, 74]. Os algoritmos baseados em prova proveem consenso probabilístico e seguem modelos similares à prova de trabalho de Nakamoto [1]. Protocolos de consenso probabilístico baseiam-se em difundir uma decisão local para os vizinhos e, eventualmente, atingir todos os participantes da rede. Em correntes de blocos, isto significa que um líder, ou um vencedor de um desafio, entre os participantes propõe o bloco e o adiciona à corrente de blocos antes de difundi-lo na rede. Caso o bloco esteja correto, garante-se que os demais participantes eventualmente o validarão e atingirão o mesmo estado global. Consensos probabilísticos possuem como principal vantagem a escalabilidade, uma vez que não é necessário conhecer todos os participantes da rede para se obter consenso. Portanto, este tipo de consenso é mais adequado a correntes de blocos públicas com muitos participantes. Em consensos probabilísticos, dois ou mais participantes podem propor blocos corretos simultaneamente, causando uma bifurcação na corrente de blocos. Nesse caso, todo protocolo de consenso probabilístico deve adotar um algoritmo de desempate para definir uma

[1]Este capítulo baseia-se nos minicursos [29] e [72]

verdade global. A regra de selecionar o ramo da bifurcação que possui a cadeia mais longa no Bitcoin é a regra de desempate mais conhecida em correntes de blocos [1].

3.1.1 Prova de Trabalho (*Proof of Work* – PoW)

Nakamoto propôs a prova de trabalho (*Proof of Work* – PoW) como um algoritmo de consenso baseado em uma competição na qual os mineradores devem resolver independentemente um desafio matemático computacionalmente custoso para provar que gastaram recursos computacionais. A probabilidade de um minerador minerar um bloco é, portanto, proporcional à sua capacidade computacional. Este desafio depende apenas do estado mais recente da corrente de blocos e portanto é totalmente paralelizável. A dificuldade do desafio é ajustada periodicamente[2] de acordo com a capacidade de mineração da rede para tentar garantir uma taxa constante de mineração de blocos[3]. Ao resolver o desafio, o minerador vencedor pode submeter o bloco e a prova de trabalho em difusão para todos os seus vizinhos. Não exite restrições de participação no consenso; qualquer máquina da rede pode tentar gerar um novo bloco. O nó que vence o desafio recebe uma boa recompensa pelo sucesso como forma de incentivo pela corretude do trabalho realizado. Assim, os nós maliciosos tendem a seguir a ordem instituída pelos nós honestos, pois os ganhos em agir honestamente são mais vantajosos do que os ganhos com ações maliciosas [1].

Apesar da inovação, a prova de trabalho empregada no Bitcoin e no Ethereum apresentam limitações evidentes de desempenho. O Bitcoin possui uma latência de consenso de aproximadamente dez minutos e uma vazão de transações de apenas 7 transações por segundo. O Ethereum possui uma latência de 15 segundos e uma vazão de 15 transações por segundo. Os sistemas atuais requerem altas taxas de transações por segundo com baixa latência, o que consiste em um grande desafio para os protocolos baseados em prova de trabalho [15, 28]. Em resposta às limitações de desempenho do consenso baseado em prova de trabalho, diversos protocolos de consenso foram propostos como alternativas ao protocolo do Bitcoin. Dentre eles, destaca-se a prova de posse, considerada pelo Ethereum como o substituto natural da prova de trabalho para prover um consenso escalável sem custo energético elevado [75, 76].

3.1.2 Prova de Posse (*Proof of Stake* – PoS)

A prova de posse é uma sub-categoria de algoritmos baseados em prova para correntes de blocos públicas que utiliza a quantidade de recursos em posse (*stake*)

[2]O Bitcoin reajusta a dificuldade do desafio computacional a cada 2016 blocos ou cerca de duas semanas.

[3]No Bitcoin, a taxa alvo é de um bloco a cada 10 minutos.

de cada participante como base para escolher o bloco a ser adicionado na corrente de blocos. A prova de posse surgiu principalmente como uma alternativa ao alto gasto energético característico da prova de trabalho de Nakamoto [1] e introduziu o conceito chamado de "mineração virtual"(*virtual mining*) por alguns autores [13, 77]. Comparado à prova de trabalho, na qual a probabilidade de um participante propor um bloco é proporcional somente a seu poder computacional (*hashpower*), na prova de posse a probabilidade de propor um bloco é proporcional à quantidade de ativos investidos por um participante no momento do consenso. Devido à ausência do ato de "minerar"característico da prova de trabalho, um participante de protocolos PoS é normalmente chamado de nó validador (*minter* ou *stakeholder*). As principais vantagens da prova de posse em comparação à prova de trabalho incluem maior segurança, menor risco de centralização e maior eficiência energética [13, 64, 75]. Em um algoritmo genérico baseado em prova de posse, a rede armazena, de forma distribuída, o conjunto de validadores aptos a participar da próxima rodada de consenso. Qualquer participante que possuir ativos pode se tornar um validador ao disponibilizar seus ativos como depósito. Então, inicia-se um processo de consenso que utiliza pesos proporcionais aos depósitos realizados por cada participante em relação ao total depositado.

O protocolo de consenso baseado em posse pode seguir duas abordagens principais para definir o bloco proposto: i) uma prova de posse pseudo-aleatória (*chain-based PoS*), na qual um participante com mais posses possui maior chance de ser o líder e adiciona um novo bloco imediatamente à corrente; ou ii) uma prova de posse baseada em acordo bizantino (*BFT-based PoS*), na qual a definição do líder também ocorre de maneira aleatória e de acordo com as posses, mas, posteriormente, os participantes trocam mensagens para validar o bloco que será adicionado à corrente de blocos, a exemplo de um protocolo BFT. Além das duas diferentes abordagens, há diversos detalhes em cada algoritmo de consenso e muitas maneiras de gerar incentivos aos validadores que participam do processo. Logo, existem muitos tipos (*flavors*) de prova de posse. Alguns autores classificam os algoritmos baseados em PoS em duas categorias de acordo com o tipo de abordagem: prova de posse pseudo-aleatória(*chain-based PoS*) e prova de posse baseada em acordo bizantino(*BFT-based PoS*) [68, 75, 77]. Outros autores citam duas categorias adicionais que admitem protocolos com centralização parcial em validadores confiáveis especiais: prova de posse baseada em comitês (*commitee-based PoS*) e prova de posse delegada (*Delegated Proof of Stake* – DPoS) [13]. Esta dissertação considera os protocolos que misturam os conceitos de consenso probabilístico e consenso determinístico como protocolos híbridos e os apresenta na Seção 3.3. Portanto, os parágrafos abaixo apresentam apenas a prova de posse "pura", ou pseudo-aleatória.

A prova de posse pseudo-aleatória (*chain-based PoS*) é a classe mais antiga de al-

goritmos PoS, proposta a partir de 2011 pelos desenvolvedores do Bitcoin como uma alternativa ao mecanismo de mineração de blocos da prova de trabalho. Este tipo de PoS herda características similares à prova de trabalho de Nakamoto [1], como a seleção pseudo-aleatória de um participante para adicionar um bloco, a regra da maior cadeia e a finalidade probabilística. Assim como na mineração da prova de trabalho, "minerar"na prova de posse baseada em corrente consiste em encontrar um cabeçalho de um bloco cujo *hash* seja menor do que uma alvo de dificuldade pré-estabelecida. No entanto, este tipo de prova de posse provê dois mecanismos para prevenir gasto energético excessivo no processo: i) uma janela de tempo máxima na qual o participante proponente deve resolver o desafio, para prevenir que haja gasto computacional por tempo indeterminado; e ii) a redução proporcional da dificuldade do desafio computacional de acordo com quantidade de ativos que um participante investiu no momento do consenso. Através destes novos mecanismos, participantes que possuam maiores investimentos têm suas dificuldades reduzidas e possuem maior chance de resolver o desafio com menor número de operações. Ainda que o processo de validação seja similar ao procedimento da prova de trabalho, a dificuldade média de atingir o alvo do desafio computacional é significativamente menor que a do Bitcoin. Portanto, o PoS evita a competição baseada somente em força bruta característica da prova de trabalho e, consequentemente, reduz os gastos energéticos.

O Peercoin [21] e o Nxt [20] são as primeiras criptomoedas a implementar a prova de posse baseada em algoritmos probabilísticos. Assim como na prova de trabalho de Nakamoto, em ambos os protocolos cada participante deve calcular um *hash* criptográfico para atingir um alvo, mas limitado a uma janela de tempo e cuja dificuldade diminui de acordo com a posse do participante. Formalmente, o desafio é encontrar um número aleatório único (*nonce*) n tal que $hash(n) < alvo * posse$, onde o alvo é um valor fixo iniciado por zeros e a posse é a quantidade de recursos que o participante possui na rede.

A principal diferença entre o Peercoin e Nxt são os mecanismos de incentivo à participação. Geralmente, o valor da posse é proporcional somente à quantidade de recursos de um participante. No entanto, para estimular a atividade de participantes com poucos recursos, um esquema de aumento de valor pode ser usado para ajustar o valor de uma posse antiga e ainda não selecionada. O Peercoin implementa uma métrica de justiça chamada "idade da moeda"(*coin age*) para aumentar linearmente o valor da posse no consenso conforme passam-se as rodadas. No fim de uma rodada, somente o valor da posse do vencedor da rodada retorna ao seu valor inicial. O resultado é que, mesmo que o participante possua poucos recursos e pouca chance de vencer uma rodada, suas chances de propor um bloco aumentam com o tempo e a rede torna-se menos centralizada em nós com posses maiores. O protocolo

limita os ganhos de valor a um período de 90 dias para impedir que os participantes depositem posses deliberadamente pequenas e mesmo assim gerem blocos ao esperar tempo suficiente.

Em comparação ao Peercoin, o Nxt incentiva mais a atividade de participantes com muitos recursos. O protocolo não valoriza a posse conforme as rodadas de consenso ocorrem; em vez disso, aumenta o valor da posse somente na janela de tempo da mesma rodada de consenso. Assim, o valor de uma posse aumenta significativamente apenas quando os participantes demoram a resolver o desafio computacional. Além disso, no Nxt: i) os participantes não escolhem o quanto de recursos depositam, e o depósito é sempre o total de recursos de cada participante; e ii) a recompensa por gerar um bloco é proporcional à quantidade de transações válidas, de forma que o participante maximiza seus ganhos ao agir honestamente. Os mecanismos para previnir tanto a monopolização do consenso quanto ataques que exploram deliberadamente a valorização da posse através de múltiplos pequenos depósitos. No entanto, o Nxt é mais suscetível do que o Peercoin à centralização em participantes com muitos recursos.

Apesar dos benefícios de redução energética, a prova de posse baseado em algoritmos probabilísticos introduz um novo desafio-chave: o problema de "nada a perder"(*nothing at stake*), no qual a decisão ótima é validar todos os caminhos possíveis quando ocorre uma bifurcação na corrente de blocos.

O problema de "nada a perder"(*nothing at stake*)

Nas primeiras implementações da prova de posse [20, 21, 73, 78], apenas possuir ativos é suficiente para participar e obter vantagem no processo de consenso. Essas implementações baseiam-se na ideia de que é desnecessário e ineficiente "apostar"ativos através de depósitos que são destruídos ao final do processo de proposta de um bloco. No entanto, a não-exigência de depósitos permite o ataque de "nada a perder"(*nothing at stake*), no qual os participantes podem utilizar seus ativos para participar simultaneamente na validação de blocos conflitantes quando uma bifurcação ocorre na rede. Em particular, este é o comportamento mais vantajoso, que será seguido por todo validador racional por dois motivos: i) ao contrário da prova de trabalho, não custa poder computacional para que um participante valide transações em múltiplas bifurcações. Portanto, torna-se computacionalmente eficiente validar várias bifurcações simultaneamente; ii) se um validador participa de múltiplas bifurcações, ele ganhará recompensas para qualquer bifurcação que seja considerada correta após o desempate. Isto significa que o comportamento que maximiza a probabilidade de ganhos é participar de todas as bifurcações possíveis. Consequentemente, todo participante racional que deseja maximizar seu lucro seguirá este comportamento.

O problema "nada a perder"(*nothing at stake*) pode ser modelado matematicamente como um problema de maximização de probabilidades. Sejam uma corrente de blocos bifurcada com dois caminhos conflitantes[4] A e B e um participante genérico m que possui uma parcela (*stake*) $r_m \in [0, 1]$ do total de recursos no sistema. A Figura 3.1 ilustra o cenário do problema com os caminhos conflitantes.

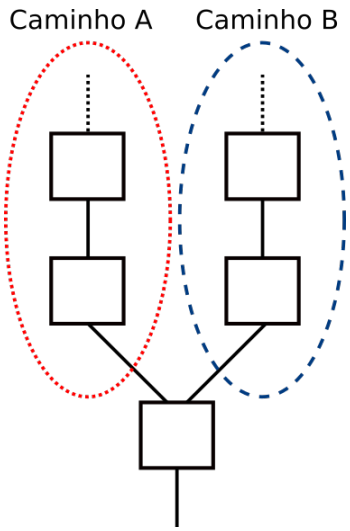


Figura 3.1: Uma corrente de blocos bifurcada com dois caminhos conflitantes A e B. Eventualmente um dos caminhos deve ser escolhido como correto para construir a corrente de blocos finalizada.

Definem-se os possíveis eventos:

- C_A: o sistema eventualmente finaliza[5] o caminho A e o caminho B é abandonado;
- C_B: o sistema eventualmente finaliza o caminho B e o caminho A é abandonado;
- $Val_m(C)$: o participante m utiliza seus recursos para validar o caminho C;
- $Venc_m$: o participante m vence a rodada de consenso e recebe as recompensas pré-acordadas pelo sistema.

Na prova de posse (*proof of stake*) clássica, não há gasto de recursos para validar um dos caminhos possíveis nem mecanismos de punição para evitar a validação de múltiplos caminhos. Assim, ainda que C_A e C_B sejam eventos mutuamente exclusivos, é perfeitamente possível que o participante m utilize todos os seus recursos para validar ambos os caminhos (i.e. $Val_m(A) \cap Val_m(B)$), realizando uma validação

[4]Caminhos conflitantes são caminhos que partem do mesmo bloco de origem e possuem mesma altura, de forma que não basta simplesmente aplicar a regra da maior corrente de Nakamoto [1].

[5]Finalizar um caminho significa prover ao caminho a propriedade de finalidade discutida na Seção 2.3.3, efetivamente considerando-o o caminho correto entre os caminhos conflitantes.

dupla (*double stake*) [75]. Em comparação, na prova de trabalho isto não é possível pois a capacidade computacional do participante seria dividida entre as validações de cada caminho. As probabilidades de que o participante m vença a rodada de consenso considerando cada possível cenário são:

$$p(Venc_m|Val_m(A) \cap \neg Val_m(B)) = r_m p(C_A) \tag{3.1}$$

quando o participante m valida apenas o caminho A,

$$p(Venc_m|\neg Val_m(A) \cap Val_m(B)) = r_m p(C_B) \tag{3.2}$$

quando o participante m valida apenas o caminho B, e

$$p(Venc_m|Val_m(A) \cap Val_m(B)) = r_m[p(C_A) + p(C_B)] \tag{3.3}$$

quando o participante m valida ambos os caminhos. Utilizando a propriedade de exclusão mútua entre C_A e C_B e a lei da probabilidade total, a Equação 3.3 pode ser simplificada pois $p(C_A) = 1 - p(C_B)$:

$$p(Venc_m|r_m(A) \cap r_m(B)) = r_m[p(C_A) + 1 - p(C_A)] = r_m \tag{3.4}$$

Como $r_m > r_m p(A)$ e $r_m > r_m p(B)$, o valor esperado de validar ambos os caminhos será sempre maior que escolher apenas um dos caminhos. Portanto, esta é a decisão que maximiza a probabilidade de vencer uma rodada de consenso e que, consequentemente, maximiza os ganhos do participante m. A consequência imediata deste resultado é que todo participante racional no sistema validará ambos os caminhos e a finalidade de um dos caminhos não é garantida mesmo sem a presença de atacantes. Além disso, realizar um gasto duplo torna-se muito mais fácil, uma vez que um atacante precisa apenas possuir mais recursos do que os participantes altruístas (i.e. os participantes que validarão apenas um dos caminhos possíveis). Em teoria dos jogos, este é o conhecido problema da tragédia dos comuns. Na prova de trabalho, este problema não ocorre pois dividir o poder computacional entre as bifurcações não aumenta a chance de minerar um bloco.

3.1.3 Alternativas Baseadas em Prova (*Proof-of-X* – PoX)

Outros algoritmos baseados em prova procuram mitigar as limitações de desempenho e o excesso de gasto energético presentes na prova de trabalho e os desafios da prova de posse [60, 73, 79]. A seguir está uma explicação resumida dos algoritmos mais conhecidos:

- **Delegated Proof of Stake (DPoS).** Os participantes utilizam seus ativos para eleger delegados em um quórum que define o bloco a ser adicionado. A quantidade de votos de um minerador é proporcional aos seus ativos [79]. A principal vantagem é o aumento na eficiência ao utilizar delegados e a principal desvantagem é a possível centralização do consenso em delegados populares;

- **Proof of Authority (PoA).** Similar ao DPoS, porém o conjunto de delegados (autoridades) é pré-determinado em acordo e suas identidades são públicas e verificáveis por qualquer participante da rede [80];. A principal vantagem é a fácil fiscalização das autoridades e a principal desvantagem é a centralização em autoridades sem possibilidade de eleição;

- **Proof of Capacity (PoC).** A probabilidade de propor um bloco é proporcional ao espaço de armazenamento cedido à rede por um participante do consenso. Quanto maior a capacidade de armazenamento em disco, maior o domínio sobre o consenso [49]. A principal vantagem é permitir um consenso baseado em discos, que possuem baixo custo e baixo gasto energético, e a principal desvantagem é a possível centralização em grandes centros de armazenamento, como serviços de nuvem;

- **Proof of Elapsed Time (PoET).** Cada participante do consenso recebe um temporizador aleatório decrescente e o nó cujo temporizador terminar primeiro propõe o próximo bloco [74]. Este protocolo de consenso funciona exclusivamente em *hardware* que suporta a tecnologia *Intel Software Guard eXtensions* (SGX). O Intel SGX garante, através de regiões privadas de memória, a distribuição aleatória de temporizadores e que nenhuma entidade tem acesso a mais de um participante do consenso. A principal vantagem é prover consenso seguro e eficiente sem grandes custos de processamento e a principal desvantagem é a dependência de hardware específico.

3.2 Consenso Determinístico: Protocolos Baseados em Quórum

Em redes bem-definidas, como redes permissionadas síncronas ou eventualmente síncronas, é possível obter consenso determinístico através de troca de mensagens. Neste tipo de consenso, todos os participantes devem votar pela aprovação ou rejeição de um bloco e atingir um consenso antes de adicionar o novo bloco à corrente. Como consequência, todos os participantes devem ser conhecidos, identificados e deve haver sincronia entre os nós para a troca de mensagens. O consenso baseado em quórum garante que a adição de um bloco à corrente ocorre ao mesmo tempo para todas as réplicas. Portanto, a consistência do sistema é garantida em qualquer momento e não existem bifurcações na corrente de blocos. As tolerâncias a falhas de parada e a comportamentos maliciosos são definidas de acordo com as especificações de cada protocolo de consenso. Um consenso determinístico baseado em quórum é mais eficiente em vazão de transações e latência de consenso do que algoritmos baseados em prova, porém sacrifica escalabilidade em número de participantes [60, 81].

Os protocolos de consenso determinístico são divididos em tolerantes a falhas de paradas (*Crash Tolerant Fault* - CFT) e tolerantes a falhas bizantinas (*Byzantine Fault Tolerant* - BFT). Na tolerância a falhas de parada, um participante pode parar de responder às mensagens. Na tolerância a falhas bizantinas, o participante malicioso pode, além de parar de responder, apresentar comportamento arbitrário. Protocolos BFT são capazes de atingir até dezenas de milhares de transações por segundo com baixa latência, sob a restrição operarem em redes com sincronia eventual e com poucos participantes do consenso [52, 64]. Isto faz com que os protocolos BFT sejam ideais para redes permissionadas e menos escaláveis nas quais os participantes podem agir de maneira maliciosa. Os protocolos BFT, em comparação a protocolos tolerantes apenas a falhas por parada (CFT), proveem uma confiança mais robusta, pois toleram comportamentos maliciosos na rede. No entanto, os protocolos BFT necessitam de mais réplicas para tolerarem a mesma quantidade de falhas bizantinas que os protocolos tolerantes a falhas de parada toleram. Também toleram menos falhas do que os clássicos 50% dos protocolos baseados em prova [1].

Em especial, a tolerância a falhas de parada assume as condições propostas no problema do parlamento em tempo-parcial, descrito por Lamport [23]:

1. todos os participantes são conhecidos e identificados;
2. todos os participantes podem trocar mensagens diretamente;
3. as mensagens trocadas chegam ao destino em tempo finito;
4. toda mensagem é assinada por seu emissor;

5. todo participante pode falhar, incluindo o líder do consenso;
6. as mensagens podem ser perdidas, reordenadas ou duplicadas;
7. o caminho entre dois participantes pode ser interrompido;
8. os participantes em falha podem deixar de responder às mensagens;
9. todos os participantes são honestos e não há comportamento malicioso;

O problema dos generais bizantinos, descrito por Lamport para exemplificar um protocolo BFT [57], modifica as condições 8 e 9 e inclui uma nova condição para prevenir comportamento malicioso:

8. participantes em falha podem deixar de responder, mentir ou omitir mensagens;
9. não há garantia de honestidade entre os participantes e pode haver comportamento malicioso;
10. as mensagens podem ser alteradas por um terceiro no meio do caminho;

A ideia principal dos protocolos CFT é eleger um líder forte e confiável que inicia e comanda uma rodada de consenso distribuindo o novo bloco proposto a todos os participantes do consenso. Os participantes do consenso respondem ao líder com suas avaliações e o líder computa e difunde a decisão final na rede. Os demais participantes detectam possíveis falhas no líder através de temporizadores e iniciam um processo de eleição caso o líder falhe. Nos protocolos BFT, a ideia é eleger um líder que apenas inicia e coordena uma rodada de consenso, mas que não detém com exclusividade as informações de validação da rede. Não se assume que o líder é confiável e, portanto, a ideia é diminuir ao máximo sua autonomia. Em protocolos BFT, os participantes do consenso também respondem ao líder com suas avaliações, mas as difundem para todos os outros participantes. Assim, todos os participantes do consenso se informam sobre o voto dos demais e possuem provas coletivas de qual deveria ser a decisão do sistema. Isto permite fiscalizar e prevenir possíveis ações maliciosas do líder. Ao receber uma quantidade suficiente de votos positivos, cada participante considera que o quórum foi alcançado e os participantes escrevem o novo bloco na corrente simultaneamente. Protocolos BFT toleram no máximo $f = \frac{n}{3} + 1$ falhas bizantinas de participantes. Protocolos CFT toleram no máximo $f = \frac{n}{2}$ de falhas de parada de participantes. Alguns protocolos toleram menos falhas devido a decisões de projeto [60, 61]. O Apêndice A.1 discute em detalhes a justificativa do limiar de falhas em acordos bizantinos.

Existem diversos protocolos de consenso baseados em quórum propostos para correntes de blocos na literatura. A seguir são apresentados um protocolo tolerante a falha de parada e três protocolos tolerantes a falhas bizantinas em um sistema parcialmente síncrono. O protocolo tolerante a falha de parada é o Raft [65], utilizado na plataforma Hyperledger Fabric [82], a plataforma mais utilizada para criar correntes de blocos empresariais. Os protocolos tolerantes a falhas bizantinas são: i) PBFT [83] - o primeiro protocolo BFT prático e o mais conhecido até hoje, que utiliza vetores de código de autenticação de mensagem (*message authentication code* - MAC) ao invés de assinaturas digitais para conseguir um desempenho similar aos protocolos que toleram apenas falhas de parada; ii) o protocolo Ripple [60], responsável pelo consenso do XRP, terceira maior criptomoeda em valor de mercado, e o primeiro a implementar um consenso BFT federado em escala global; e iii) o protocolo Stellar [84], considerado por muitos autores como a "continuação"do Ripple e que já desponta como o consenso da SCP, a 12ª maior criptomoeda em valor de mercado.

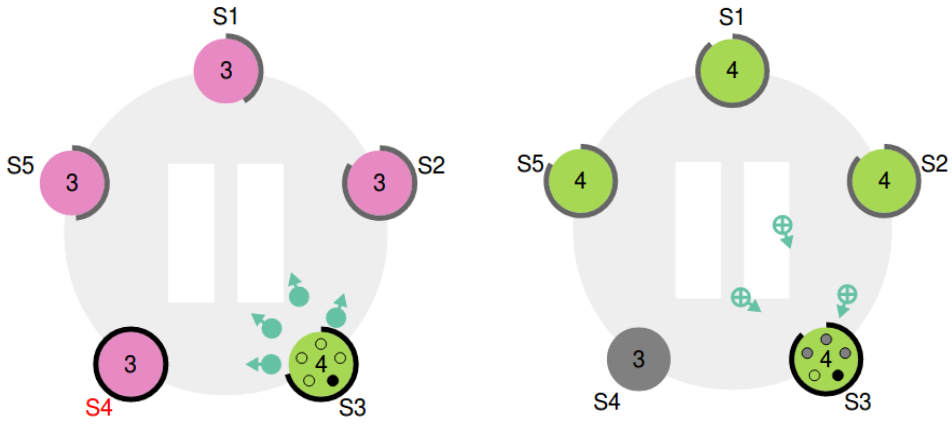
3.2.1 O Protocolo Raft

Raft[6] é um protocolo de consenso tolerante a falhas de parada (*crash*) proposto por Ongaro em 2014 [22]. Desenvolvido como uma alternativa de fácil compreensão ao protocolo Paxos [85], o Raft possui eficiência e resultados equivalentes ao Paxos, além de providenciar uma fundamentação mais adequada para construção de sistemas práticos. A simplicidade na compreensão é consequência da divisão do consenso em 3 fases: a eleição do líder que ocorre para definir o primeiro líder ou caso o atual falhe; a replicação do registro nas máquinas de estado de todos os nós; e a segurança do registro, garantindo que nenhum registro é apagado ou duplicado. O protocolo é baseado nas fortes premissas de que todos os nós são conhecidos, não existe comportamento bizantino na rede e os nós operam em um ambiente assíncrono com relógios defeituosos [86]. O Raft utiliza a replicação de registros utilizando a replicação de máquina de estados (RME) [87] que, em conjunto com o protocolo de consenso, garante a existência dos mesmos comandos na mesma ordem em todos as máquinas de estados dos nós. Dessa forma, o Raft garante a integridade dos registros e o funcionamento do consenso com até f participantes em estado de falha de parada dos $n = 2f + 1$ participantes da rede.

O Raft divide o tempo em mandatos de duração indefinida e numerados com inteiros consecutivos. A qualquer momento, cada nó está em um dos três estados: líder, seguidor ou candidato. O líder aceita as entradas do cliente, replica as entradas nos outros nós e informa aos nós quando é seguro aplicar as entradas às máquinas de

[6]O nome vem de *Reliable, Replicated, Redundant, And Fault-Tolerant.*

estado. O seguidor apenas responde a pedidos do líder e do candidato. O candidato é utilizado para eleger um novo líder caso o atual falhe, assim começando um novo mandato. A falha do líder é detectada utilizando o mecanismo de pulsação periódica (*heartbeat*) através de chamadas remotas de procedimento (*Remote Procedure Call* - RPC).



(a) Início da eleição do líder.

(b) Término da eleição do líder.

Figura 3.2: A Figura 3.2(a) representa o início da eleição de um novo líder, onde o tempo limite de eleição do seguidor S3 esgotou-se. O seguidor S3 vira um candidato, incrementa o número de mandato de 3 para 4, representado pela cor rosa e verde respectivamente, e envia pedidos de voto aos demais seguidores. Logo em seguida, os seguidores respondem o pedido de voto e incrementam seus números de mandato, assim elegendo o candidato S3 como o novo líder, conforme representado na Figura 3.2(b). Como o antigo líder S4 não respondeu as chamadas do novo líder S3, detectou-se uma falha em S4, representada pela cor cinza.

Para detectar falhas, cada nó possui um temporizador de duração escolhida aleatoriamente de um intervalo de 150 a 300 ms para marcar o tempo limite de eleição. Caso o tempo limite de eleição do seguidor seja atingido sem que haja sinal de comunicação do líder pelo mecanismo de pulsação, o seguidor assume que não existe um líder ou que o líder está em estado de falha e começa a eleição de um novo líder ao incrementar o mandato atual e se tornar um candidato. Em seguida, o candidato vota em si mesmo e envia pedidos de voto para o restante dos nós por RPC. A fase de eleição está representada na Figura 3.2. O tempo limite de eleição de cada seguidor é redefinido no início de toda eleição. O candidato permanece nesse estado até que consiga $f + 1$ votos, outro candidato se estabeleça como líder ou o tempo limite de eleição seja atingido. A escolha aleatória do tempo limite de eleição garante que a existência de múltiplos candidatos é rara.

Uma vez que o líder é eleito, ele começa a atender as solicitações dos clientes. Cada solicitação de cliente consiste em um comando para a máquina de estados replicada executar. Essas solicitações são guardadas em forma de registros contendo

o comando a ser executado, o número do mandato em que o líder recebeu a solicitação e um identificador que define a sua ordem no livro-razão. O líder registra a solicitação no seu livro-razão e informa, por RPC em paralelo, os seguidores para replicar o novo registro em seus livros-razão. Após receber $f + 1$ respostas positivas dos seguidores, o líder aplica o comando do cliente em sua máquina de estados, envia uma ordem por RPC em paralelo para todos os seguidores aplicarem a entrada em suas máquinas de estados e informarem a saída de suas máquinas de estados para o líder. A partir desse ponto, a solicitação do cliente está efetivada.

3.2.2 O Protocolo Prático de Consenso Tolerante a Falhas Bizantinas

O Protocolo Prático de Consenso Tolerante a Falhas Bizantinas (*Practical Byzantine Fault Tolerance* - PBFT) [83] foi o primeiro protocolo de consenso BFT que considera aspectos práticos, alcançando um desempenho semelhante aos protocolos para falhas por parada e resolve o problema dos generais bizantinos [57]. Neste protocolo, a autenticidade das mensagens é garantida através de vetores de MACs (*Message Authentication Codes*) ao invés de assinaturas digitais, aumentando o seu desempenho ao usar criptografia simétrica em vez de criptografia assimétrica. O PBFT executa em rodadas (*rounds*), sendo que em cada rodada (*round*) um participante do consenso é escolhido líder e tentará fazer da sua proposta de novo bloco para a corrente de blocos. O PBFT necessita de $n = 3f + 1$ participantes do consenso, com réplicas da corrente de blocos, para tolerar até f falhas bizantinas e utiliza quóruns de $2f + 1$ participantes de consenso em cada passo do protocolo. Também é necessário um modelo de sistema de comunicação parcialmente síncrono para garantir terminação. A Figura 3.3 ilustra os passos do protocolo em uma execução normal, i.e., quando o líder é correto e o sistema está em um período de sincronia. Os passos do protocolo são os seguintes:

- o líder envia uma mensagem de PRE-PREPARE para todos os participantes com a sua proposta de novo bloco, p;
- os participantes do consenso fazem algumas validações para definir se a proposta de novo bloco, p, enviada pelo líder é válida. Em caso afirmativo aceitam esta proposta e enviam uma mensagem de PREPARE contendo a proposta de novo bloco, p, para todos os outros participantes do consenso;
- quando um participante do consenso receber $\lceil \frac{n+f}{2} \rceil$ mensagens PREPARE para uma mesma proposta de novo bloco, p, ele envia uma mensagem de COMMIT contendo p para todos os outros participantes do consenso;

- quando um participante do consenso receber $\lceil \frac{n+f}{2} \rceil$ mensagens de COMMIT para uma mesma proposta de novo bloco, p, então a proposta p, é decidida.

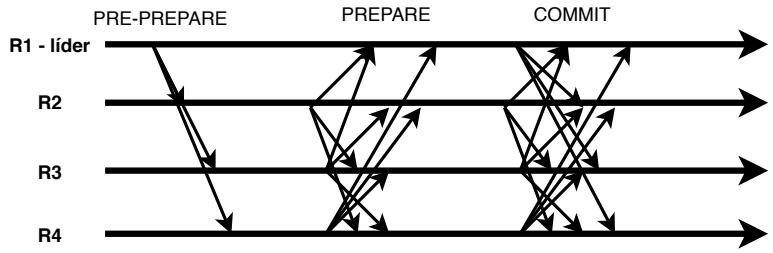


Figura 3.3: Execução normal do PBFT.

Quando o líder for malicioso e fizer propostas divergentes, um quórum de mensagens PREPARE ou COMMIT para uma mesma proposta não é obtido, o mesmo ocorre em um período de perturbação do sistema em que as mensagens não recebidas até um limite de tempo estipulado. Nestes casos, um protocolo de troca de visão é executado e uma nova rodada (*round*) é iniciada com um novo líder.

3.2.3 Protocolos de Consenso Bizantino Federado

Os protocolos de consenso tolerantes a falha bizantina procuram aumentar a vazão de transações, mas possuem o grande inconveniente de restringirem fortemente o estabelecimento de participantes do consenso. Estas redes são conhecidas como permissionadas[7]. Com o objetivo de atingir alta taxas de transações, mas, ao mesmo tempo, não apresentar um rigor tão forte na seleção dos participantes do consenso, surgiu o conceito de acordo bizantino federado (*Federated Bizantine Agreement* - FBA) que é um relaxamento em relação aos convencionais consensos tolerantes a falhas bizantinas. Diferentemente da premissa dos protocolos tolerantes a falhas bizantinas, em que não há confiança em nenhum dos participantes, os protocolos de consenso bizantino federado flexibilizam o rigor na seleção dos participantes do consenso e permitem ao participante escolher um conjunto de outros participantes nos quais confia. Assim, em vez de haver trocas de mensagens entre todos os participantes da rede, é suficiente para um participante que haja consenso apenas entre os participantes confiáveis. O Ripple implementa o conceito de acordo bizantino federado através de uma lista única e o Stellar através do conceito de fatias de quórum detalhadas em seguida.

Algoritmo de Consenso do Protocolo Ripple

O protocolo Ripple (*Ripple Protocol Consensus Algorithm* - RPCA) é um protocolo de consenso bizantino federado proposto em 2012 por David Schwartz e publicado

[7]Permissisoned Consensus Algorithms.

como artigo em 2014 [88]. A grande motivação do Ripple é fornecer uma plataforma de transferência de ativos tão simples quanto a transferência de arquivos hoje realizada pela Internet, assim criando a Internet de Valor [89]. Enquanto Bitcoin é muito usado por pessoas e organizações como moeda virtual, o sistema de pagamento pelo Ripple está sendo muito usado por bancos para transferências bancárias rápidas e com baixas taxas. O consenso Ripple é utilizado na criptomoeda XRP[8], a 3ª criptomoeda com maior valor de mercado[9]. A criptomoeda XRP possui uma taxa de 1.500 transações por segundo, 100 vezes maior que a taxa do Ethereum.

O protocolo de consenso Ripple possui 2 fases: a deliberação que ocorre em rodadas para definir a lista de transações a serem incluídas no livro-razão; e a validação que inclui as transações aprovadas da lista de transações no livro-razão, finalizando o consenso e efetivando as transações. O Ripple também usa a replicação de máquina de estados (RME) para replicar o livro-razão em todos os nós da rede e garantir um único estado global. Como um diferencial de tolerância a falhas bizantinas do protocolo, o Ripple é o único consenso BFT a garantir a integridade dos registros e o funcionamento do consenso com até f falhas bizantinas dos $5f + 1$ participantes da rede. Isso significa que precisaria de 80% dos participantes em conluio para dominar o consenso do Ripple, enquanto nos outros BFT precisariam de apenas 66% dos participantes.

A fase de deliberação é composta por rodadas de tempo fixo que seguem os seguintes itens:

- os validadores recebem as transações e as retransmitem para outros validadores que estejam na sua lista de validadores confiáveis (*Unique Node List* - UNL), onde cada validador possui sua própria UNL;
- cada validador propõe e difunde um conjunto de transações baseado nas transações recebidas de outros validadores da sua UNL;
- os validadores recebem as propostas dos validadores de suas UNL e comparam as transações com a sua própria proposta. Transações coincidentes recebem um voto;
- os itens anteriores se repetem até que o tempo da rodada termine;
- após o final da rodada, novas propostas são compostas com as transações que obtiveram um limiar de 50% dos votos do total de validadores da UNL;
- aumenta-se o limiar em 10% e inicia outra rodada. A deliberação termina somente quando o limiar atinge 80%;

[8]https://ripple.com/xrp
[9]Dados disponíveis em 25 de junho de 2019 pela https://coinmarketcap.com

- assim que o limiar atinge 80%, apenas as transações que obtiveram 80% dos votos do total de validadores da UNL irão compor o conjunto de transações final a ser considerado na fase de validação.

A fase de validação verifica se todos os validadores possuem o mesmo conjunto de transações proposto e atualiza a versão do livro-razão. O processo de verificação começa com todos os validadores aplicando o conjunto de transações proposto à versão atual da sua cópia do livro-razão. Após isso, cada validador calcula o *hash* da nova versão do livro-razão e o divulga em mensagens assinadas para os outros validadores. A rede reconhece e aceita a nova versão do livro-razão assim que 80% dos validadores assinaram e difundiram o mesmo *hash* para a versão do livro-razão, assim terminando o consenso.

Protocolo Stellar - Transferências Monetárias com Custo de Centavos

O Stellar[10] é uma plataforma de transação monetária rápida, confiável e de baixíssimo custo. Stellar conecta bancos, sistemas de pagamentos e pessoas, efetuando transferências entre duas localidades quaisquer do mundo e entre qualquer par de moedas. A moeda digital nativa do Stellar é chamada lumen (XLM) e é a 12ª criptomoeda em valor de mercado[11]. Proposto por David Mazières, professor e líder de um grupo de segurança na Universidade de Stanford, o protocolo de consenso Stellar (*Stellar Consensus Protocol* - SCP) é o principal diferencial[12], pois propõe um protocolo de consenso baseado em um acordo bizantino federado [84].

O Stellar propõe o acordo bizantino federado (*Federated Byzantine Agreement* - FBA) que é um modelo apropriado para um consenso a nível mundial, pois os participantes escolhem em que membros confiar, evitando assim uma autoridade central ou as negociações fechadas do protocolo de consenso bizantino convencional. No acordo bizantino federado, cada participante do consenso possui a flexibilidade de escolher um conjunto de membros conhecidos para confiar. O participante e o conjunto de membros em que confia formam uma fatia de quórum (*quorum slice*). Um nó concorda com um estado quando uma quantidade de membros em que confia concorda com o estado [84]. Nós podem selecionar fatias de quórum baseados em critérios arbitrários, como reputação ou arranjos financeiros, além de criar um número finito de fatias de quórum com diferentes limiares para cada. A Figura 3.4 mostra uma rede para acordo bizantino federado com múltiplas fatias de quórum. Na figura, as setas indicam a direção da confiança dos nós. Os nós 1, 2 e 3 formam

[10]https://www.stellar.org/

[11]Dado disponível em 11 de julho de 2019 em https://coinmarketcap.com/

[12]A rede Stellar foi lançada por Jed McCaled, um dos fundadores do Ripple, após discordâncias com os diretores do Ripple. O Stellar utilizava um consenso baseado no Ripple. Uma bifurcação na rede Stellar atribuída a uma falha de segurança no consenso Ripple/Stellar fez o Stellar buscar um mecanismo de consenso alternativo.

a fatia de quórum associada ao nó 1. Os nós 2, 3 e 6 formam a fatia de quórum associada ao nó 2. Os nós 3, 4 e 6 formam a fatia de quórum associada ao nó 3. Por fim, os nós 4, 5 e 6 formam a fatia de quórum associada aos nós 4, 5 e 6. Fatias de quórum devem compartilhar nós visando prevenir a criação de quóruns disjuntos, que possibilitam a votação em estados contraditórios ao não criar comunicação entre as diferentes fatias.

Assim como o Ripple, o protocolo Stellar é separado em duas etapas: nomeação e votação. A etapa de nomeação produz um conjunto de valores candidatos para uma rodada. Na etapa de votação, os nós escolhem qual dos valores gerados na etapa de nomeação vai ser a saída do protocolo.

A etapa de votação emprega um modelo em que os nós votam para aceitar ou eliminar valores candidatos. Nas duas etapas, o protocolo Stellar utiliza um modelo de votação federada, que tem como objetivo atingir um consenso sobre a corretude de um valor no cenário de múltiplas fatias de quórum. O modelo de votação federada separa essa decisão em dois passos. Primeiro, os nós trocam mensagens para decidir se efetivam ou eliminam o valor. Os nós não podem mudar os votos, mas podem aceitar diferentes saídas. Segundo, os nós verificam se um quórum votou no mesmo valor. Para encontrar um quórum, um nó deve:

- começar incluindo os nós da própria fatia de quórum;
- para cada membro da fatia, adicionar a fatia de quórum desses membros;
- repetir o passo anterior até atingir um ponto em que os próximos nós a serem adicionados já foram incluídos. Nesse ponto, o nó encontrou um quórum.

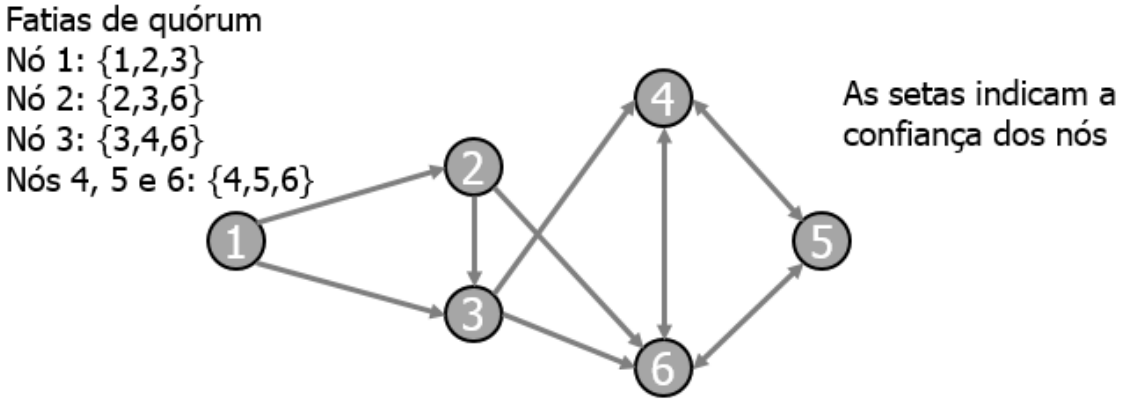


Figura 3.4: Fatias de quórum no protocolo Stellar.

Usando o cenário da Figura 3.4 como exemplo, o menor quórum que o nó 1 consegue encontrar contém todos os nós, enquanto que o menor quórum que o nó 5 consegue encontrar contém os nós 4, 5 e 6. Na rede Stellar, cada nó pode ter

múltiplas fatias de quórum, o que abre a possibilidade de encontrar diversos quóruns diferentes. Caso o nó consiga encontrar um quórum que votou no mesmo valor, ele passa a aceitar o valor como correto. Caso o quórum tenha votado e aceitado um valor diferente, o nó passa a aceitar o valor do quórum, mesmo que não tenha votado nele.

A etapa de nomeação (*nomination*) busca gerar os valores possíveis para a saída do consenso. Nesta etapa, cada nó pode nomear diversos valores como candidatos à saída do protocolo. Entretanto, quando o nó encontrar um quórum que aceite um dos valores nomeados como um candidato confirmado, o nó não pode mais nomear valores. Apesar de não poder mais nomear valores, o nó pode aceitar valores nomeados por outros participantes como candidatos à saída. Como esse processo pode gerar múltiplos valores confirmados como candidatos, os nós devem combinar deterministicamente os valores confirmados como candidatos. Usando o exemplo da rede Stellar, em que o valor nomeado é um conjunto de transações, os nós combinam os valores pegando a união dos conjuntos de transação. Esse processo resulta em uma convergência dos valores candidatos nos nós honestos.

A fase de votação (*balloting*) busca escolher diretamente a saída do protocolo de consenso, tendo como entrada os valores candidatos da etapa anterior. Como os nós não conseguem saber se o protocolo de nomeação acabou, é possível que os nós cheguem nessa etapa com valores diferentes, isto é, sem a etapa de nomeação convergir para um conjunto de valores. Isso pode fazer os nós votarem em diferentes valores como saída, o que pode fazer o protocolo ficar preso sem ter a possibilidade de conseguir um quórum que aceite qualquer um dos resultados votados. Além disso, os nós podem ter um atraso na resposta e a votação também ficaria presa. Para resolver esse problema, o protocolo determina uma cédula de votação (*ballot*) contendo um contador e o valor a ser votado. Assim, se um valor ficar preso, a cédula é incrementada para uma nova votação. Quando o quórum chegar a um acordo em qual cédula efetivar, o valor da cédula é usado como saída do protocolo.

Apesar de Ripple e Stellar serem protocolos de consenso federados, eles possuem filosofias distintas e focam em mercados diferentes. Ripple é fornecido por uma companhia de corrente de blocos privada que provê serviços para outras companhias. Companhias multinacionais e bancos podem se servir do Ripple para transferências monetárias internacionais. Por outro lado, Stellar visa transferências monetárias internacionais de indivíduos. Stellar é uma organização sem fins lucrativos que objetiva um sistema de transferência monetário de baixo custo e mais inclusivo.

3.3 Protocolos Híbridos

Os protocolos híbridos procuram combinar dois ou mais modelos de consenso com o objetivo de herdar as vantagens dos tipos de consenso usados. Os parágrafos a seguir apresentam exemplos desta classe de protocolos de consenso: o protocolo Casper FFG [26], o protocolo Tendermint [27] e o protocolo EOS [90]. O protocolo Casper FFG é o protocolo desenvolvido pelo Ethereum para substituir a prova de trabalho e que deve ser implementado por completo até 2020. O protocolo Tendermint é um dos protocolos híbridos mais famosos em correntes de blocos permisisionadas. O protocolo EOS é a base de uma nova criptomoeda que pretende substituir o Ethereum como principal sistema de contratos inteligentes e aplicações distribuídas.

3.3.1 Casper the Friendly Finality Gadget (Casper FFG)

O protocolo *Casper the Friendly Finality Gadget* (Casper FFG)[13] [26] foi proposto em 2017 por Vitalik Buterin como a primeira etapa da transição da criptomoeda Ethereum [2] para utilizar um protocolo de consenso baseado em posse. Este protocolo híbrido entre prova de trabalho e prova de posse provê finalidade, através do desempate por prova de posse, aos blocos propostos por mineradores que utilizam prova de trabalho. Assim, para a proposição de blocos o Casper usa o PoW do Ethereum como fonte de blocos e resolve as bifurcações que ocorrem através de votação com pesos proporcionais à posse de cada participante. Portanto, o Casper FFG é, na verdade, um protocolo de finalização que pode ser utilizado em conjunto com qualquer outro protocolo de consenso que provenha consenso probabilístico, sobretudo os baseado em prova. Basta que exista um fluxo constante adições de blocos à corrente de blocos. A adoção atual no Ethereum prevê um protocolo híbrido PoW/PoS utilizando a prova de trabalho *ethash*, já existente na plataforma, para propor blocos e o Casper FFG para finalizá-los.

Antes do processo de finalização de blocos, quando ainda existem bifurcações causadas por atrasos de rede ou ataques, a estrutura de corrente de blocos é, na verdade, uma árvore de blocos chamada no Casper FFG de *BlockTree*. Novos blocos gerados pelo protocolo de proposta são adicionados constantemente à árvore de blocos. Para reduzir a sobrecarga de comunicação do protocolo, o protocolo define uma árvore de blocos menor a partir da árvore de blocos chamada de árvore de pontos de checagem (*CheckpointTree*) contendo apenas os blocos cujas alturas são múltiplos de 50. Todo o restante do protocolo é realizado sobre a árvore de *checkpoints*. Cada validador realiza um depósito com a quantidade de ativos de deseja "apostar"no protocolo de consenso.

[13] O protocolo Casper é proposto como uma evolução do algoritmo GHOST (fantasma em inglês) e Casper então seria o Gasparzinho, o fantasminha camarada.

O voto no Casper FFG consiste em cinco campos de informação: um ponto de checagem justificado s, um ponto de checagem-alvo t, suas alturas $h(s)$ e $h(t) > h(s)$, e a assinatura do validador que enviou o voto. Um ponto de checagem justificado é um para o qual existem mais de 2/3 de votos válidos registrados que aprovam o ponto de checagem. Todos os votos são difundidos na rede e possuem pesos proporcionais ao depósito realizado pelo validador que assinou o voto. Se mais de 2/3 dos participantes emitem o mesmo voto contendo s e t, cria-se um "enlace de supermaioria"(*supermajority link*) entre os dois pontos de checagem, s é considerado finalizado e t é considerado justificado. Todos os blocos entre s e t na árvore de blocos original também são considerados finalizados. O nó raiz da árvore de pontos de checagem é considerado justificado por definição.

Para garantir a propriedade de acordo entre os validadores sobre os pontos de checagem, o Casper FFG utiliza um mecanismo de punição (*slashing*) para qualquer validador que descumpra um dos mandamentos do Casper: i) nenhum validador pode votar em dois pontos de checagem com a mesma altura e que, portanto, são conflitantes; e ii) nenhum validador deve votar em um par de pontos de checagem cujo caminho está contido em um caminho já votado pelo validador. Validadores que violem algum dos mandamentos estão sujeitos a perda total do seu depósito e possível banimento de participação em futuras rodadas do consenso. Como todos os votos são assinados por seus respectivos validadores, as evidências da violação são registradas e a detecção de malfeitores é realizada facilmente. A implementação atual do Casper FFG é realizada através de um contrato inteligente no Ethereum[14].

3.3.2 O Protocolo Tendermint

O Tendermint [27, 61] é um protocolo de consenso tolerante a falhas bizantinas, que não requer mineração por prova de trabalho e oferece proteção contra gasto duplo, simultaneamente apresentando uma taxa de milhares de transações por segundo e segurança baseada em fatores intrínsecos. O protocolo atua de modo similar a prova de posse ou participação (*proof of stake*), com os nós responsáveis por adicionar novos blocos a rede oferecendo parte de seus ativos como um modo de atuarem no processo de efetivação. Diferentemente da prova de posse ou participação, o Tendermint foi desenvolvido tendo ataques de nada a perder (*nothing at stake*) em mente, implementando medidas de segurança para proteger a rede destes ataques. Como o protocolo é tolerante a falhas bizantinas, garante-se a segurança da corrente de blocos contra até 1/3 de participantes bizantinos.

Cada bloco adicionado a corrente é constituído por 3 partes: o cabeçalho, os validadores do bloco anterior e as transações inclusas nesse bloco. Para a deter-

[14]Disponível em https://github.com/ethereum/EIPs/blob/master/EIPS/eip-1011.md

minação do *hash* do bloco, primeiramente utiliza-se uma função resumo em cada uma dessas partes; a seguir, esta função é utilizada novamente sobre os *hashes* obtidos. Este *hash* do bloco é utilizado em uma Árvore de Merkle (Merkle Tree) de modo a permitir a verificação de qualquer bloco na corrente, garantindo assim sua auditabilidade.

Validadores são contas que possuem moedas em um depósito de títulos (*bond deposit*), com essas moedas determinando o poder de votação do validador. Para adicionar moedas ao depósito, o usuário deve transmitir uma transação de título (*bond transaction*), indicando à rede quantas moedas serão utilizadas com este propósito. Esses validadores participam do consenso difundindo assinaturas criptográficas, ou votos, para concordar na adição de um bloco à corrente. Essas moedas podem ser transferidas e gastas, desde que não estejam armazenadas no depósito de títulos; para removê-las desse depósito, deve-se transmitir uma transação de desvínculo (*unbond transation*) com este objetivo, e aguardar por um período pré-determinado denominado período de desvínculo (*unbonding period*). O funcionamento das transações de título e desvínculo é apresentado na Figura 3.5. O poder de votação da rede é igual a soma do poder de votação de todos os participantes.

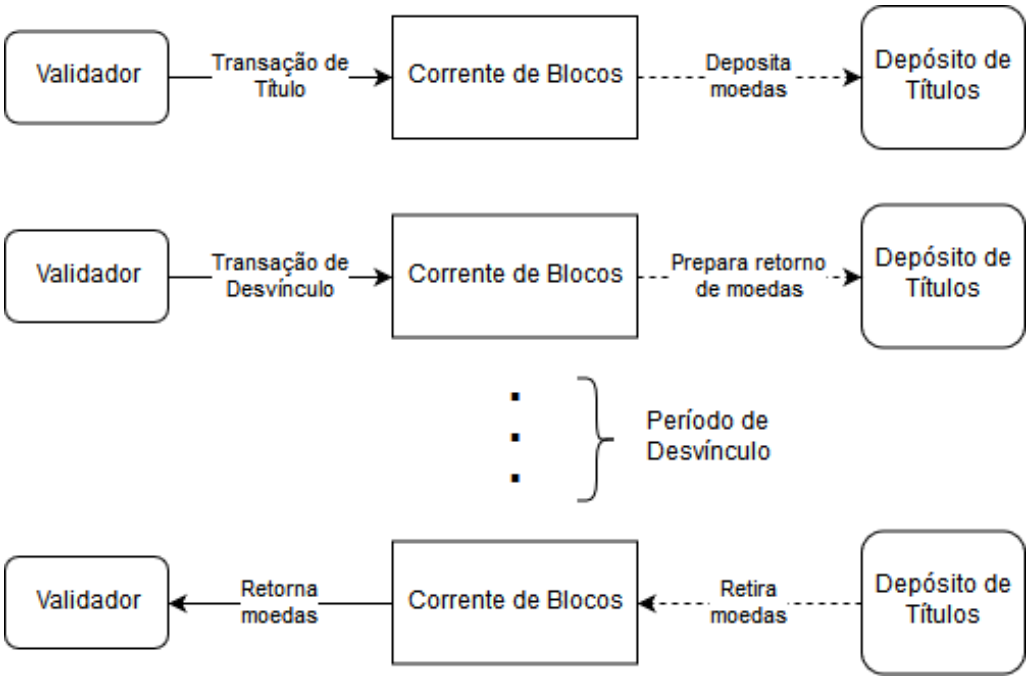


Figura 3.5: Transações de título e desvínculo do protocolo Tendermint, utilizadas para modificar o poder de votação de cada validador.

O processo de adição de novos blocos a corrente se dá através de rodadas. Cada rodada é composta por 3 passos: proposta (*propose*), pré-voto (*prevote*) e pré-efetivação (*precommit*), com os validadores enviando votos em cada passo da rodada. Existem 3 tipos de votos: o pré-voto (*prevote*), a pré-efetivação (*precommit*)

e a efetivação (*commit*). Um bloco é dito efetivado pela rede quando assinado e difundido por mais de 2/3 do poder de votação dos validadores.

As etapas que constituem uma rodada estão apresentadas na Figura 3.6. Cada rodada é feita de modo que um proponente (*proposer*) seja escolhido entre os validadores, onde os que possuem maior poder de votação são escolhidos com maior frequência. Inicialmente o proponente escolhido anuncia o bloco através de uma proposta, com os outros nós confirmando sua validade, e que este foi recebido a tempo, através de um pré-voto. Quando mais de 2/3 do poder de votação da rede realizar o pré-voto para este bloco, realiza-se então a pré-efetivação antes que se inicie o processo de efetivação do bloco. Esta também ocorre quando mais de 2/3 do poder de votação da rede concordar na pré-efetivação do bloco.

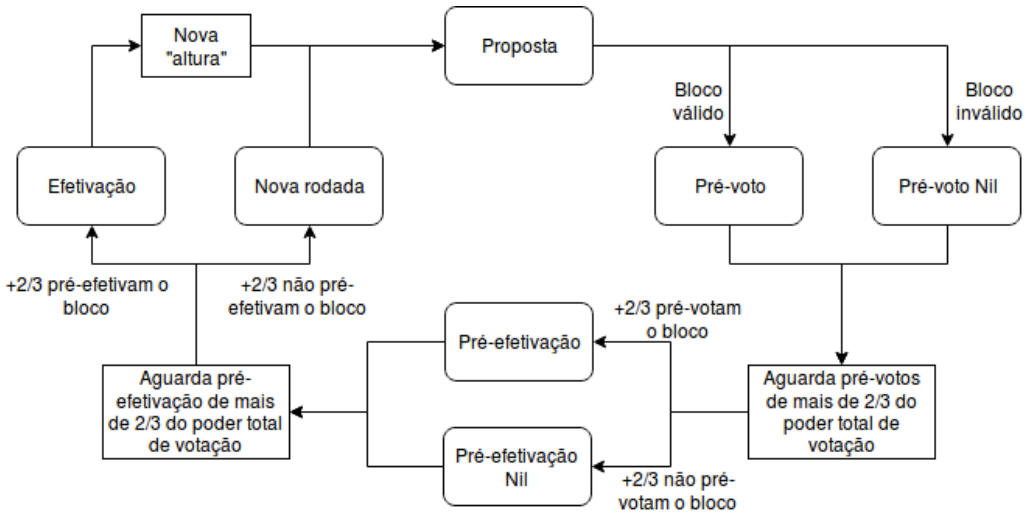


Figura 3.6: Etapas de uma rodada do protocolo Tendermint.

Um bloco é considerado efetivado quando pelo menos 2/3 do poder total de votação assina votos de efetivação para esse bloco. Uma bifurcação (*fork*) ocorre quando dois ou mais blocos na mesma "altura" são assinados por uma maioria de 2/3 do poder de votação, indicando que pelo menos 1/3 dos validadores votaram com duplicidade. Quando isto ocorre, uma transação de evidência (*evidence transaction*) é gerada, destruindo as moedas de título dos validadores culpados. O período de desvínculo garante que validadores maliciosos ainda possuirão suas moedas no depósito de título quando a transação de evidência for gerada, desse modo assegurando que a punição pelo ataque será devidamente realizada. Essa medida evita que validadores votem para efetivar diferentes blocos em uma única rodada, devido ao risco de ocorrer prejuízo financeiro, desencorajando assim ataques de nada a perder.

3.3.3 O Protocolo EOS

O protocolo EOS[90], utilizado pela criptomoeda EOS e desenvolvido por Dan Larimer [15], implementa conceitos tanto da prova de posse delegada quando da tolerância a falhas bizantinas (*Byzantine Fault Tolerance - Delegated Proof of Stake*). Isso oferece ao protocolo uma alta taxa de transações, atualmente suportando até 4000 transações por segundo, graças ao DPoS; assim como uma maior segurança em frente a ataques bizantinos, oferecida pelo BFT.

Nesta moeda, participantes da rede que possuem *tokens* (*token holders*) podem utilizá-los para eleger produtores de blocos (*block producers*) através de um sistema de votação, com esses produtores atuando de modo similar a delegados no DPoS. Ao votar em um produtor de blocos, a quantia em tokens enviada pelo voto é adicionado ao "montante em votos" do produtor para fins de consenso. Qualquer participante da rede pode se candidatar para tornar-se um produtor de blocos, com o único requisito sendo receber ao menos um voto de outro *token holder*. Entretanto, usualmente os votos são dados a entidades que investem diretamente no crescimento da criptomoeda, com exemplos incluindo grandes consórcios como o EOS New York e o EOS Beijing.

A adição de blocos à corrente é feita através de rodadas, com cada rodada selecionando os 21[16] produtores de blocos que possuem o maior montante em votos para validar as transações. Durante uma rodada, blocos são produzidos em fatias de tempo (*time slices*) de 0,5 segundo cada, com essas fatias sendo atribuída aos produtores. Cada produtor é atribuído um total de 6 fatias de tempo, sendo responsável por produzir um bloco para cada uma delas; caso a fatia de tempo expire sem que um bloco seja produzido, esse bloco é pulado. A ordem que os produtores de blocos são selecionados para o envio dos blocos é determinada no começo da eleição, após ser alcançado um consenso entre, no mínimo, 15 dos produtores. Desse modo, cada rodada resultará em até 126 blocos adicionados à corrente, e dura um total de 63 segundos. As principais etapas de uma rodada estão ilustradas na Figura 3.7.

A finalização dos blocos é feita com tolerância a falhas bizantinas (BFT), de modo que todos os produtores devem assinar para cada bloco válido recebido. Quando 15 dos 21 produtores (mais de 2/3 dos delegados) concordarem em um bloco, ele é adicionado à corrente, processo que geralmente demora de 2 a 3 segundos. No caso de *forks*, os participantes não devem assinar mais de um bloco na mesma altura; é possível verificar quando um produtor valida ambos, desse modo agindo

[15] Dan Larimer criou a plataforma BitShares (https://bitshares.org/) em outubro de 2015, cofundou a corrente de blocos social Steem (https://steem.com/) e, atualmente, preside a empresa Block.one, que desenvolve a plataforma EOS.

[16] A quantidade total de produtores de blocos foi selecionada partindo de um trabalho anterior do autor, onde participantes da rede podiam votar para aumentar ou reduzir a quantidade de validadores ativos.

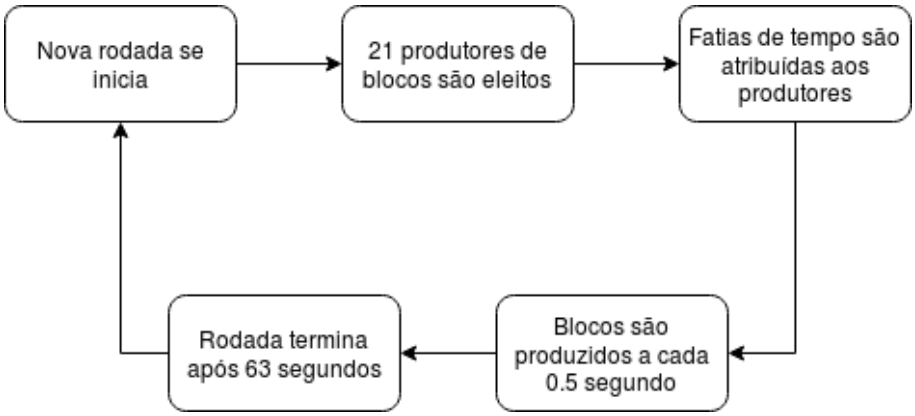


Figura 3.7: Rodada do protocolo de consenso EOS.

maliciosamente, verificando-se o conteúdo dos blocos que compõem a *fork*. Dado que os produtores de bloco são eleitos a partir de votos que recebem de outros participantes da rede, ser identificado como um produtor não confiável por realizar ataques de nada-a-perder potencialmente levará este participante a perder votos e, assim, sua posição como produtor. Desse modo, é mais vantajoso agir como um participante legítimo, evitando ataques de nada-a-perder.

Caso um produtor permaneça mais de 24 horas sem produzir blocos, ele é removido da seleção no início de cada rodada. Ele somente passará a ser considerado novamente após notificar a corrente de blocos de sua intenção de voltar a produzir blocos. Desse modo, melhora-se a eficiência da rede removendo-se produtores que se provaram não confiáveis.

3.4 Protocolos Baseados em Grafo Acíclico Direcionado

A maior parte dos sistemas baseados em correntes de blocos implementa algum tipo de protocolo de consenso baseado em prova, baseados em quórum ou protocolos híbridos. No entanto, recentemente surgiram novos protocolos que propõem substituir a estrutura de dados da corrente de blocos por uma nova estrutura baseada em grafo acíclico direcionado (*directed acyclic graph* - DAG) das transações do sistema. A principal ideia de utilizar DAG, que tem como maior representante a criptomoeda IOTA [28], é paralelizar a validação de transações e eliminar a necessidade de prover uma corrente de blocos única que deve sempre ser validada em consenso. A aprovação e a validação de transações anteriores são feitas pelos próprios usuários, possivelmente *offline*, e, quanto mais usuários enviando transações, mais desempenho e segurança o sistema possui.

3.4.1 IOTA Tangle

O IOTA[17] [28] é uma criptomoeda construída para atender a micro-pagamentos máquina a máquina (*machine to machine* - M2M) característicos de um ambiente de Internet das Coisas. Seus mecanismos de pagamento e protocolo de consenso, formalizados por Popov em 2016 [28], baseiam-se em uma estrutura de dados inovadora chamada *Tangle*. A principal inovação do *Tangle* é a estrutura de livro-razão distribuído, que reúne as transações da rede em um grafo acíclico direcionado em vez de utilizar uma corrente de blocos. Além disso, o Tangle elimina a distinção entre clientes e mineradores: todos os usuários do sistema devem realizar trabalho para emitir uma nova transação. Uma característica notável do *Tangle* em comparação ao consenso em corrente de blocos é que diferentes participantes na rede podem ter as diferentes visões das transações. Esta característica contrasta fortemente com a visão global da corrente de blocos, na qual todas as transações são idênticas em qualquer participante.

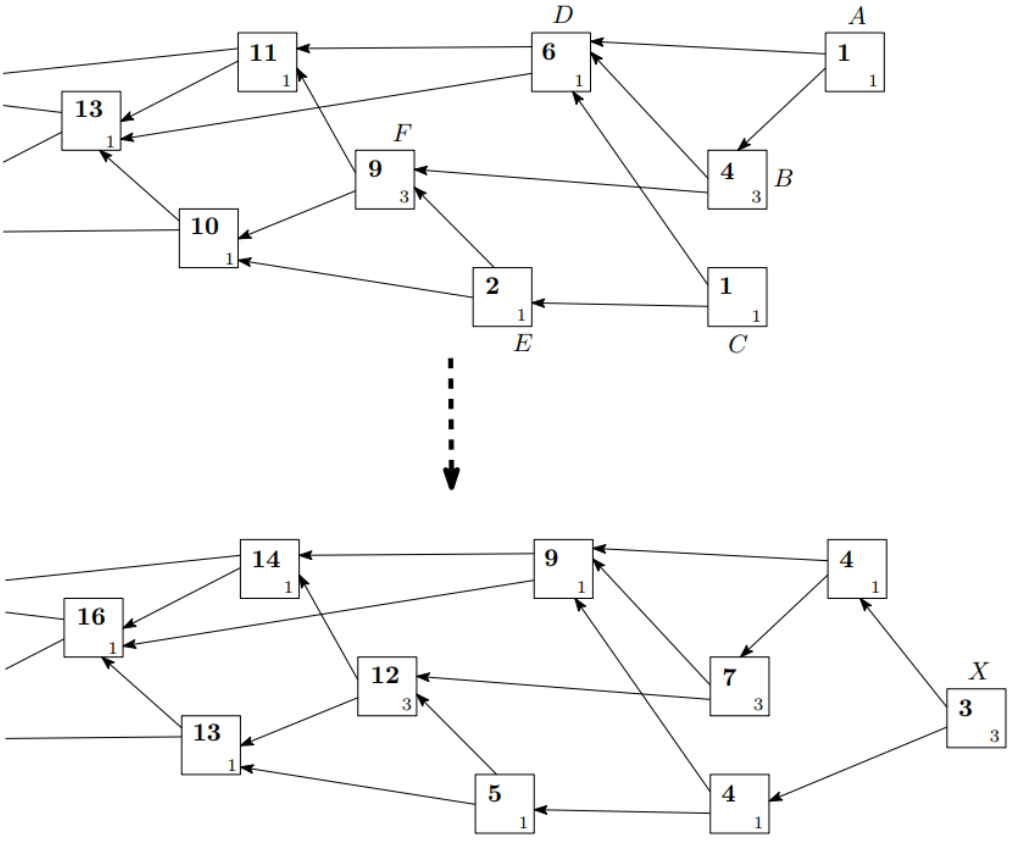


Figura 3.8: A adição de uma ponta, X, na estrutura de dados do *Tangle*. O peso individual de cada transação é representado no canto direito inferior e o peso acumulado é representado pelo número em negrito.

A Figura 3.8 mostra um exemplo da estrutura de dados do *Tangle*. Cada vértice

[17]Internet of Things Association.

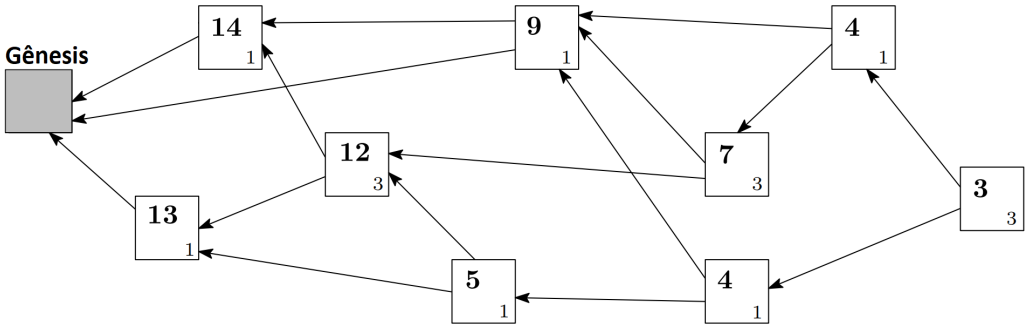


Figura 3.9: Um exemplo de estrutura de dados baseada em grafos acíclicos direcionados (DAG) do *Tangle*. Cada vértice do grafo representa uma transação e cada aresta representa o resultado da validação de uma transação. O número no canto inferior direito de cada transação representa o peso individual da transação, e o número ao centro, em negrito, representa o peso acumulado da transação.

do grafo representa uma transação e cada aresta representa o resultado da validação de uma transação. O usuário deve confirmar ao menos duas transações não-confirmadas para adicionar sua transação à *Tangle*[18]. As transações não-confirmadas são chamadas de "pontas"(*tips*) do *Tangle*. Para adicionar uma transação à rede, o usuário adiciona os resumos das duas pontas escolhidas à sua transação, resolve um desafio baseado em prova de trabalho, e difunde o resultado na rede. A prova de trabalho, neste caso, tem dificuldade bem menor que a do Bitcoin e serve apenas como um mecanismo para prevenir *spam* de transações. O procedimento de adição de uma transação cria duas novas arestas direcionadas no grafo que confirmam as transações anteriores e funcionam como uma versão generalizada da sequência de funções resumo da corrente de blocos. O IOTA Tangle não recompensa financeiramente os validadores de transações, pois o incentivo para validar transações é adicionar sua própria transação à rede. Todas as moedas trocadas na rede são criadas na primeira transação do sistema, chamada de transação gênesis.

Um dos conceitos fundamentais no IOTA é o peso de transações: o peso individual e peso acumulado. O peso individual de uma transação é um peso proporcional aos recursos gastos por um usuário para emitir a transação. A ideia dos pesos individuais é diferenciar transações mais importantes, que possuem maior peso individual, de transações com menor importância, que possuem menor peso individual. O peso acumulado de uma transação é definido pelo seu peso individual mais a soma de todos os pesos individuais das transações que aprovam, direta ou indiretamente, a transação. Uma transação aprova diretamente outra se há uma aresta conectando as duas, e indiretamente se há pelo menos um caminho composto de mais de uma aresta entre as duas transações.

As transações no IOTA não precisam de uma aprovação por consenso para serem

[18]Na implementação do IOTA, o número de confirmações necessárias para adicionar uma transação à rede é exatamente dois.

adicionadas ao grafo. Todas as transações são adicionadas, bastando que o usuário resolva o desafio computacional necessário para encadear as transações. No entanto, quando há pontas conflitantes, cada usuário precisa decidir quais transações serão consideradas válidas para escolher duas pontas que serão aprovadas por sua nova transação. As pontas inválidas são ignoradas pelo usuário. O principal mecanismo para identificar pontas válidas é realizar múltiplas rodadas do algoritmo de seleção de pontas (*tip selection algorithm*) e verificar qual das duas pontas conflitantes tem a maior probabilidade de ser escolhida. Por exemplo, se uma ponta foi escolhida 95 vezes em 100 rodadas do algoritmo de seleção, a ponta tem confiança de 95%. O IOTA Tangle atualmente utiliza um algoritmo de seleção baseado em passeios aleatórios (*random walks*) e cadeias de Markov e Monte Carlo (*Markov-chain and Monte Carlo* - MCMC) que prioriza transações de maior peso acumulado.

Quando existem mais de duas pontas válidas disponíveis para seleção, o usuário escolhe as duas pontas de acordo com o peso acumulado de cada ponta.

Capítulo 4

Provendo uma Infraestrutura de Software Fatiada, Isolada e Segura de Funções Virtuais através da Tecnologia de Corrente de Blocos

As redes móveis de próxima geração fornecem um modelo de conectividade com múltiplos serviços de rede adaptados para atender a demanda de cada segmento de cliente. As redes definidas por *software* (*Software-Defined Networking* - SDN) e a virtualização de funções de rede (*Network Function Virtualization* - NFV) são as principais tecnologias que utilizam a virtualização para fornecer a capacidade de programação da rede. Assim, as tecnologias NFV e SDN criam uma cadeia de funções de rede (*Service Function Chain* - SFC) fim-a-fim [91] para fornecer serviços sob demanda e adaptados a cada aplicação. Apesar de a associação de NFV e SDN fornecer a agilidade e o baixo custo desejados pelas telecomunicações, surgem novos desafios de segurança [92]. Além disso, o impacto de possíveis ataques aumenta porque os ataques aos hospedeiros de funções de rede podem comprometer simultaneamente milhares de usuários [93]. Portanto, é de grande importância reduzir os possíveis vetores de ataque a funções virtuais de rede (*Virtual Network Functions* - VNF) e fornecer um gerenciamento de configuração seguro e confiável. Garantir o isolamento entre fatias de rede é essencial para evitar ataques comuns em infraestruturas compartilhadas. Além disso, os inquilinos de cada fatia compartilham a mesma infraestrutura de nuvem, e as cadeias podem envolver funções virtualizadas instanciadas em domínios de provedores concorrentes. O ambiente multi-inquilino e multi-domínio aumenta a possibilidade de ataques dentro da nuvem, ao mesmo tempo que dificulta a responsabilização dos provedores de serviços quando ocorre uma falha. É necessário garantir que a cadeia de serviços, formada por uma cadeia

de funções virtuais de rede, seja construída de maneira confiável em um ambiente sem confiança entre os pares. Em um ambiente multi-inquilino formado por provedores concorrentes, é vantajoso para um provedor criar uma ataque para prejudicar um concorrente. Logo, a capacidade de auditoria é obrigatória para identificar uma configuração de VNF defeituosa ou comprometida, e a tecnologia de corrente de blocos atende a esta necessidade, pois fornece as características necessárias de não repúdio e imutabilidade do histórico de configuração de uma função virtual.

Esta dissertação propõe utilizar a tecnologia de corrente de blocos para registrar, como transações assinadas, todos os comandos que criam, modificam, configuram, migram ou destroem as funções de rede de cada fatia da rede. Portanto, todos os problemas de funcionamento da rede podem ser verificados e um erro pode ser atribuído corretamente a um provedor de serviço em um ambiente de concorrência, multi-inquilino, e sem confiança.

Em trabalhos anteriores, avaliou-se o desempenho do uso de correntes de blocos na virtualização de funções de rede para proteger comandos de gerenciamento, atualizações e migração de funções virtuais de rede com garantias de anonimidade [48, 94].

Este capítulo[1] foca no uso de correntes de blocos para fatiamento da rede (*network slicing*). Fatias de rede suportam requisitos de serviço para atender redes veiculares tolerantes a atraso, internet das coisas (*Internet of Things* - IoT), indústria 4.0 e serviços críticos como saúde eletrônica (*e-Health*), cidades inteligentes (*smart cities*) e redes elétricas inteligentes (*smart grids*). O cenário extraordinariamente diverso requer diferentes correntes de blocos com características específicas adaptadas ao serviço requerido. Por isso, este artigo propõe atender aos diferentes requisitos de cada fatia de rede através de diferentes categorias de correntes de blocos. A estrutura de dados, o protocolo de consenso e o protocolo de comunicação das corrente de blocos são adaptados a cada funcionalidade de fatia de rede específica. O trabalho apresenta uma arquitetura baseada em correntes de blocos para criar fatias de rede fim-a-fim seguras e adaptadas para cada caso de uso. Um protótipo de caso de uso que segue a arquitetura proposta com diferentes tipos de correntes de blocos é implementado usando a plataforma de código aberto Hyperledger Fabric [40]. O protótipo implementa dois contratos inteligentes (Hyperledger *chaincode*) com formatos de transação específicos para proteger o gerenciamento de fatias de rede e as operações de configuração de VNFs. Cada fatia de rede é executada em um canal Hyperledger isolado. Os resultados mostram que é possível proteger a construção de fatias de rede, mas que estruturas de dados otimizadas são necessárias para aumentar a taxa de transações necessárias para atender às fatias.

[1] Este capítulo se baseia nos artigos [95] e [96].

4.1 Fatiamento Seguro de Redes através de Corrente de Blocos

A utilização de corrente de blocos é necessária em ambientes distribuídos em que os participantes não conseguem chegar a um acordo sobre uma autoridade centralizada que governe todos os procedimentos sensíveis de rede. Em centros de dados virtualizados nos quais vários serviços de nuvem são orquestrados, a presença de uma VNF mal-intencionada em um serviço pode afetar toda a cadeia pela qual os pacotes são roteados sem o conhecimento do administrador da nuvem. Além disso, se um invasor tiver acesso ao orquestrador, o registro de operações pode ser manipulado para ocultar uma ameaça. O uso de corrente de blocos, apesar de envolver uma quantidade maior de processamento como um todo, permite gerenciar e atualizar VNFs de forma distribuída e segura, onde as transações podem ser verificadas localmente por cada nó com a garantia de não repúdio e integridade.

4.1.1 Modelo de Atacante

Esta dissertação considera o modelo de atacante de Dolev *et al.*, no qual um atacante pode ler, enviar e descartar uma transação endereçada à corrente de blocos, ou qualquer pacote da rede [97]. O atacante pode se conectar passivamente à rede e capturar trocas de mensagens ou injetar, reproduzir, filtrar e trocar informações ativamente. Os ataques podem ter como alvo inquilinos, VNFs, a corrente de blocos em si e a rede.

Ataques à corrente de blocos objetivam impedir que uma transação ou bloco legítimo sejam adicionados à corrente de blocos. Para que um ataque à corrente de blocos seja bem-sucedido, o atacante deve controlar uma parcela significativa da rede para afetar o protocolo de consenso. Um protocolo de consenso tolerante a falhas mitiga esse tipo de ataque. Os ataques que exigem corrupção ou adulteração de transações são impossíveis quando todas as transações incluem seu *hash* assinado correspondente.

Ataques a inquilinos ou VNFs consistem em tentar obter informações de configuração ou personificar o alvo. Os ataques de personificação não são possíveis, pois todas as transações enviadas para a corrente de blocos são assinadas por seus emissores. A encriptação de informações confidenciais mitiga ataques que buscam obter informações de configuração, nos quais o atacante precisa obter a chave privada da vítima. Este trabalho não aborda o caso em que um inquilino ou VNF é comprometido através de invasão de terminal ou sequestro de chave. A arquitetura proposta, no entanto, elimina a necessidade de qualquer serviço de escuta ativo em um VNF, e utiliza terminais em modo somente leitura para mitigar vetores de ataque. Além

disso, a arquitetura proposta permite a auditoria de todas as transações passadas. Portanto, se um invasor tentar modificar a corrente de blocos usando pares de chaves roubados, a tentativa será registrada. Após a descoberta de um incidente, o inquilino ou provedor pode facilmente substituir os pares de chaves roubados, restabelecendo a segurança e evitando mais danos.

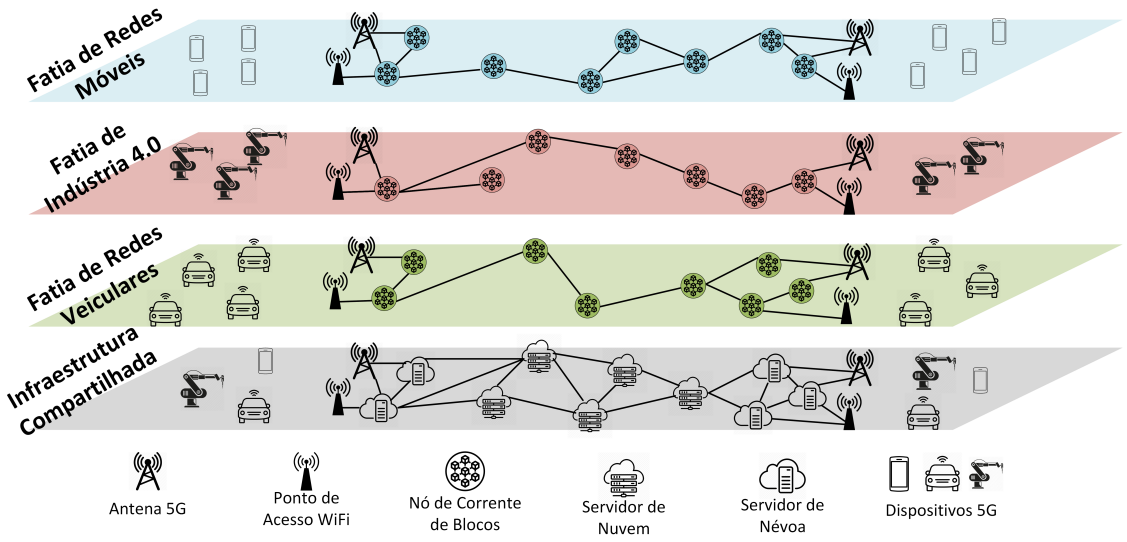


Figura 4.1: Fatias de rede isoladas através de corrente de blocos em uma infraestrutura física compartilhada. Cada fatia de rede é adaptada às necessidades de um caso de uso.

Ataques à rede representam a tentativa de isolar um único inquilino, um grupo de inquilinos ou um grupo de VNFs da rede, impedindo assim que a rede execute transações ou leia conteúdo da corrente de blocos. Esta categoria de ataque contempla ataques clássicos de rede, que podem ser mitigados através do estabelecimento de caminhos redundantes entre a corrente de blocos e VNFs ou inquilinos. Este trabalho assume uma rede pública redundante, como a Internet, que interconecta todos os participantes. A suposição dificulta o isolamento de uma única entidade se o invasor não estiver em sua rede adjacente. A mitigação completa de ataques de rede está fora do escopo deste trabalho. A arquitetura proposta foca nos ataques à corrente de blocos e transações antecipadas. No entanto, ao eliminar os serviços de escuta das VNFs, a arquitetura elimina os ataques de negação de serviço da camada de aplicação, que são uma ameaça comum em ambientes de nuvem compartilhados.

4.2 A Arquitetura Proposta

Diferentes casos de uso exigem funcionalidades específicas e incorrem em diferentes características de corrente de blocos para cada fatia de rede. Em vez de tentar

abordar todos os casos de uso, este trabalho propõe classes de corrente de blocos que atendem a muitos casos de uso. Assim, uma simples taxonomia de fatias baseadas em corrente de blocos composta de quatro categorias de fatias pode abordar uma infinidade de possíveis casos:

- **Fatias de administrador único.** Essa categoria trata casos de uso nos quais a fatia de rede inteira é administrada por uma única entidade. Neste caso, as correntes de blocos atuam apenas como um banco de dados distribuído, no qual as decisões da rede são controladas por uma autoridade central, como em redes privadas.

- **Fatias multi-domínio tolerantes a falhas de parada ou desastrosas**. Essa categoria trata casos de uso nos quais nós individuais na rede podem falhar, mas a rede está livre de comportamentos mal-intencionados. Essa categoria fornece alta eficiência para ambientes descentralizados de vários domínios que garantem a segurança da rede por meio de políticas baseadas em contrato. Este cenário é ideal para redes de até dezenas de administradores. Exemplos incluem comunicação horizontal e concordância entre os controladores SDN para implementar um serviço. Este tipo de fatia é baseia-se em correntes de blocos federadas e utiliza os protocolos RAFT e PAXOS.

- **Fatias multi-domínio tolerantes a falhas bizantinas.** Essa categoria usa protocolos de consenso tolerantes a falhas bizantinas (*Byzantine Fault Tolerance* - BFT) para proteger a rede contra comportamentos maliciosos. Os protocolos BFT fornecem eficiência razoavelmente alta para ambientes com até algumas centenas de nós identificados. Casos de uso incluem ambientes NFV em vários domínios e com vários inquilinos que implementam serviços fim-a-fim. Esta categoria baseia-se em correntes de blocos federadas robustas a ataques de conluio [48, 63].

- **Fatias públicas totalmente descentralizadas.** Esse tipo de fatia de rede fornece escalabilidade de milhares de nós, sacrificando a eficiência e a taxa de transferência. Essas fatias dependem de protocolos baseados em provas que determinam um estado global por meio de consenso probabilístico. As fatias de rede baseadas em provas fornecem alta escalabilidade, pois não precisam conhecer todos os nós da rede para obter consenso. Portanto, esse tipo de fatia é mais adequado para redes públicas com muitos dispositivos, como fatias de IoT.

Este trabalho propõe uma arquitetura na qual cada categoria fatia de rede baseada em corrente de blocos aborda um ou mais casos de uso do 5G, criando redes

isoladas com segurança e confiança. A Figura 4.1 descreve um cenário que usa corrente de blocos para três fatias de rede: uma fatia de rede móvel, uma fatia de indústria 4.0 e uma fatia de redes veiculares. Para garantir a justiça no protocolo de consenso, cada centro de dados pode hospedar no máximo um nó de corrente de blocos por fatia de rede. Os nós de corrente de blocos em uma fatia são invisíveis para qualquer entidade fora da fatia. O trabalho propõe fornecer a capacidade de auditoria de criação e gerenciamento de fatias, registrando todas as operações de orquestração da VNF em uma corrente de blocos de gerenciamento. A corrente de blocos de gerenciamento registra as operações de orquestração que criam ou modificam uma fatia de rede. Toda operação é assinada pelo cliente que solicitou a modificação. Os participantes da corrente de blocos devem validar cada operação por consenso e fornecer uma prova assinada irrefutável de que a transação foi aceita antes que as operações sejam realizadas. A solicitação assinada combinada com o registro permanente fornecido pela corrente de blocos garante que um comportamento malicioso seja rastreável. Assim, a proposta de corrente de blocos gerencial garante a procedência, a prestação de contas e a rastreabilidade das falhas em um ambiente NFV multi-inquilino e multi-domínio. Além disso, o trabalho propõe o uso de contratos inteligentes para fornecer automação e transparência em ambientes sem confiança distribuídos, em vez de confiar em um determinado nó para receber e processar transações. As propriedades de automação e transparência dos contratos inteligentes são ideais para criar fatias de rede fim-a-fim seguras que envolvem VNFs em vários domínios concorrentes, pois garantem que todos os nós da rede obedeçam ao mesmo conjunto de regras e que o código executado seja visível para qualquer nó participante.

A arquitetura proposta, representada na Figura 4.2, compreende quatro componentes principais: uma interface de usuário, o módulo de gerenciamento e orquestração NFV (NFV-MANO), um módulo de servidor de criação de corrente de blocos e um módulo de gerenciamento de corrente de blocos. Os módulos compõem o módulo de gerenciamento global, que é responsável por conectar o cliente aos serviços oferecidos. Nesta arquitetura, o cliente interage com o módulo de gerenciamento global por meio de uma interface de usuário para criar/modificar uma fatia de rede segura ou para solicitar informações de fatia, como configurações de VNFs e cadeias de funções. O cliente especifica as características da fatia, como VNFs desejadas e restrições de posicionamento, e a categoria de corrente de blocos correspondente. O módulo de interface do usuário converte as especificações em operações de criação de fatias/corrente de blocos e as envia como transações assinadas para aprovação na corrente de blocos de gerenciamento. Depois que as transações são aprovadas, o módulo NFV-MANO e o módulo de criação de corrente de blocos verificam a corrente de blocos de gerenciamento para obter operações pendentes. Os módulos executam

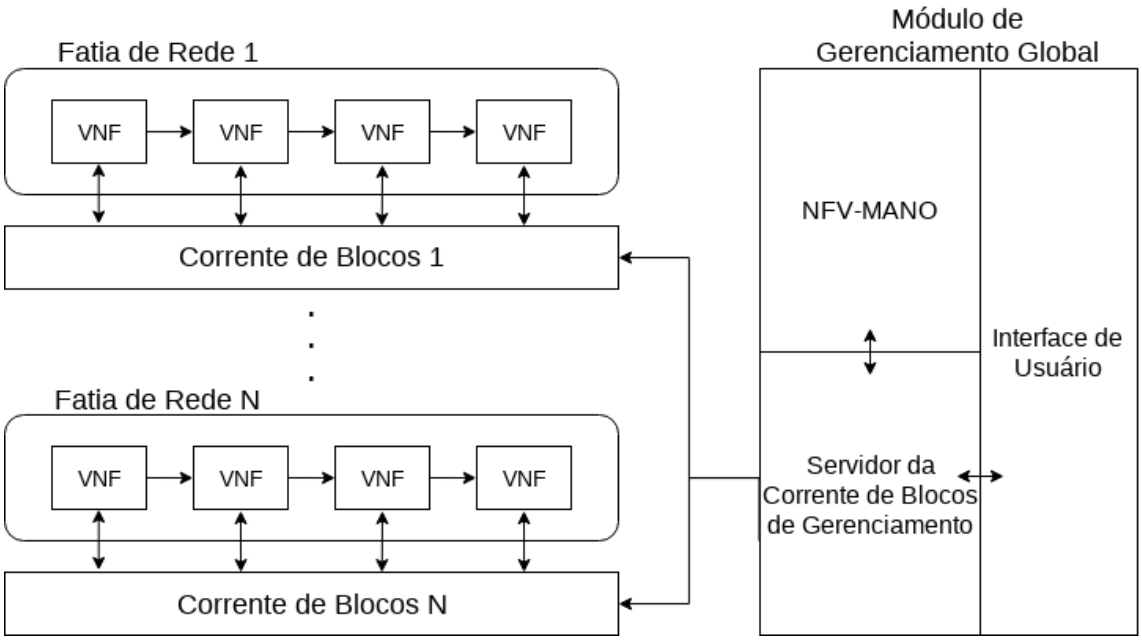


Figura 4.2: A arquitetura proposta baseada em corrente de blocos para fatiamento de rede. O usuário interage com o módulo de gerenciamento global para criar fatias de rede seguras. Cada VNF em uma fatia de rede é conectada a uma corrente de blocos responsável por registrar solicitações de configuração e informações relevantes, conforme especificado pelo usuário.

as operações para criar novas fatias seguras e emitem transações de resposta assinadas para a corrente de blocos de gerenciamento. O cliente pode então interagir com o módulo de interface do usuário para verificar se a fatia segura solicitada foi criada com sucesso.

4.3 O Protótipo baseado no Hyperledger Fabric

Este trabalho desenvolve um protótipo da arquitetura proposta que usa a plataforma Hyperledger Fabric [40]. O Hyperledger Fabric é uma plataforma de código aberto para implementar correntes de blocos entre organizações em ambientes sem confiança. Cada organização mantém uma réplica da corrente de blocos e pode acrescentar blocos através de um protocolo de consenso. A arquitetura modular do Hyperledger Fabric permite aos administradores de rede projetar sistemas baseados em sub-redes isoladas que suportam correntes de blocos específicas. As correntes de blocos no Hyperledger Fabric suportam protocolos de consenso plugáveis que proveem a personalização proposta para atender a casos de uso e modelos de confiança específicos. Os desenvolvedores das correntes de blocos podem configurar permissões de leitura e escrita para criar redes privadas, federadas ou públicas. O Hyperledger Fabric fornece um serviço de identidade e associação de membros que gerencia os identificadores dos usuários e provedores, e autentica todos os participantes na rede

para criar redes permissionadas. Nós e canais são os conceitos chave mais importantes de uma rede baseada em corrente de blocos permissionada do Hyperledger Fabric.

Os nós representam as entidades que participam do processamento de uma transação ou mantêm uma cópia da corrente de blocos. O Hyperledger Fabric provê três tipos de nós: clientes, pares (*peers*) e ordenadores. Um cliente representa um usuário que envia transações aos pares para validação e assinatura, e transmite propostas de transação assinadas para o serviço de ordenação. Os pares são um elemento fundamental da rede porque executam propostas de transação, validam transações e mantêm os registros na corrente de blocos. Os pares também instanciam contratos inteligentes e armazenam o estado global, uma representação sucinta do estado mais recente da corrente de blocos. Os nós ordenadores formam coletivamente o serviço de ordenação, que é responsável por estabelecer a ordem total e o empacotamento de todas as transações em um bloco usando um protocolo de consenso. Os ordenadores não participam da execução da transação nem validam transações. O desacoplamento das funcionalidades de ordenação e validação aumenta a eficiência da rede, pois permite o processamento paralelo de cada fase. A Figura 4.3 descreve um exemplo de uma corrente de blocos permissionada com quatro organizações no Hyperledger Fabric. Cada organização recebe transações de clientes e as retransmite para os ordenadores após a validação pelos pares. Cada organização possui um único ordenador, garantindo a justiça no protocolo de consenso.

Caminhos de mensagens diferentes, chamados canais, isolam as correntes de blocos. Um canal Hyperledger Fabric é uma sub-rede de comunicação privada e isolada entre um subconjunto de nós da rede específicos para fornecer privacidade e confidencialidade às transações. Todos os dados transmitidos em um canal, incluindo transações, contratos inteligentes, configurações de associação e informações de canal, são invisíveis e inacessíveis a qualquer entidade externa a um canal. A funcionalidade do canal é ideal para a proposta de oferecer correntes de blocos personalizadas para diferentes serviços de rede, pois permite que os administradores dos canais estabeleçam diferentes formatos de bloco e transação, além do protocolo de consenso, para cada canal. Portanto, os canais podem ser usados para oferecer fatias de rede protegidas por correntes de blocos configuradas de forma específica. Formatos de transação são definidos em contratos inteligentes, chamados de *chaincode* no Hyperledger Fabric, escritos em Go, Node.js ou linguagem Java.

4.3.1 Avaliação do Protótipo

O trabalho implementa duas correntes de blocos que representam, respectivamente, a corrente de blocos de gerenciamento e um exemplo de corrente de blocos

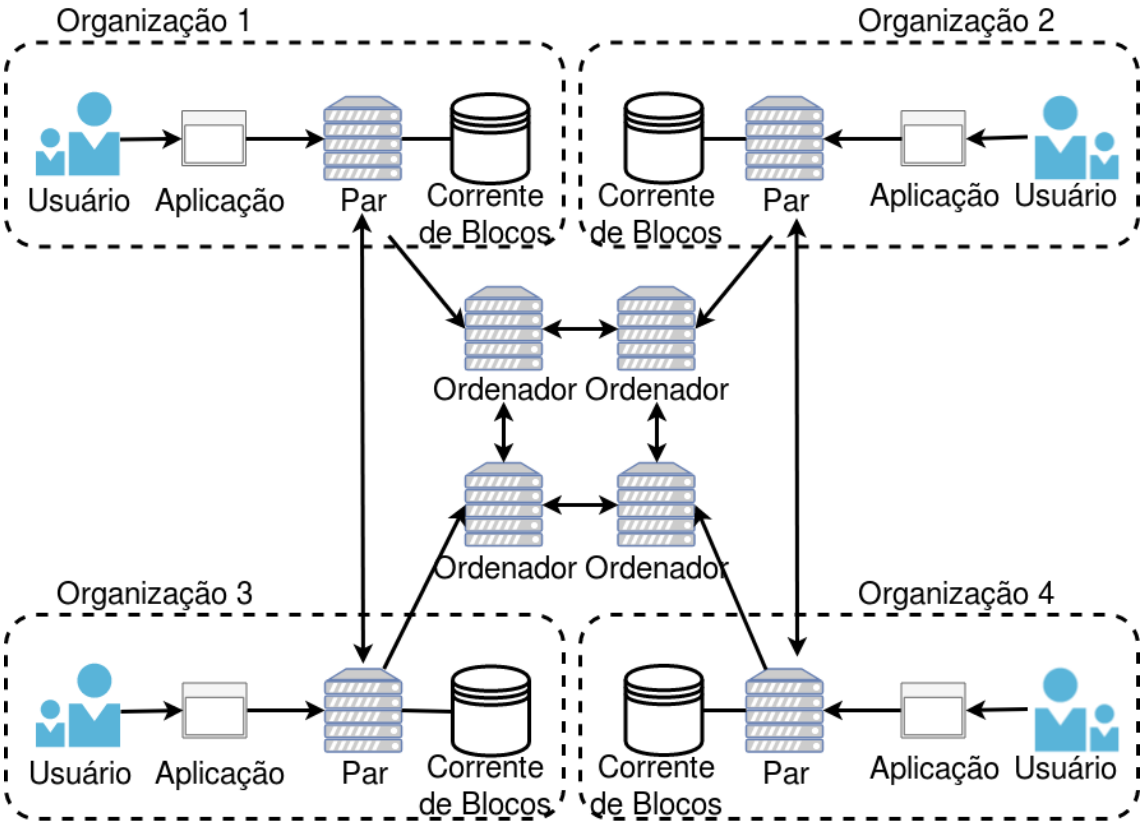


Figura 4.3: Uma corrente de blocos permissionada do Hyperledger Fabric. Usuários de cada organização usam aplicações para emitir transações as retransmitem para os ordenadores para o ordenamento global e a adição em um bloco. Depois de um bloco ser proposto e aceito pelos ordenadores através do consenso, os pares atualizam a corrente de blocos e o estado global.

para proteger uma fatia de rede. O exemplo de corrente de blocos protege a atualização de configuração da VNF e a migração em uma fatia de rede por registrar permanentemente configurações na corrente de blocos. Um consórcio com três organizações controla as duas implementações de corrente de blocos. Cada uma das três organizações controla um nó que possui direitos administrativos sobre a rede. Todas as três organizações recebem transações de um número variável de nós clientes na rede de corrente de blocos. Um computador Intel Core i7-7700 CPU 3.60GHz com 64 GB RAM cria todos os nós como contêineres Docker. Contêineres constroem múltiplos ambientes isolados de espaços de usuários que permitem a otimização de recursos computacionais da rede de corrente de blocos.

O trabalho implementa dois contratos inteligentes[2] escritos em Go, que são executados em todos os nós da rede [48, 94]. O primeiro contrato inteligente, parcialmente descrito na Lista 1, gerencia autonomamente o gerenciamento e a orquestração de VNF através de transações de instrução e resposta. Quando um cliente solicita uma fatia, o servidor da corrente de blocos de gerenciamento de emite uma tran-

[2]O código completo está disponível em https://github.com/gta-ufrj/hpsr-smart-contracts

sação de instrução com o comando de instrução. O contrato coloca a transação de instrução em uma fila de transações pendentes de instrução. O código notifica o módulo NFV-MANO, que executa a instrução pendente e envia a saída do comando para o servidor da corrente de blocos de gerenciamento. O módulo NFV-MANO emite uma transação de resposta que inclui um campo contendo o identificador da transação de instrução correspondente. Isso fornece a rastreabilidade de cada transação executada na corrente de blocos e, portanto, a responsabilização de entidades maliciosas.

```
struct instructionTransaction
{
    command                 string
    transactionType         string
    transactionName         string
    issuer                  string
}
initialize queue
initInstruction (instruction <command,name,issuer>)
{
    if instruction is not unique or instruction is not well-formatted:
        return error
    putState (instruction.name, instruction)
    put (transactionID, queue)
    notify orchestrator
    return success
}
```

Listing 4.1: Parte do pseudo-código do contrato inteligente que emite transações de instrução. O campo de comando contém a operação de orquestração a ser executada pelo módulo NFV-MANO. O contrato estabelece uma fila de instruções pendentes a serem processadas pelo orquestrador NFV.

O segundo contrato inteligente, descrito na Lista 2, define e atualiza uma configuração de uma VNF. Um cliente emite uma transação para a corrente de blocos conectada a cada VNF em uma fatia de rede. A transação contém um texto descritivo com a configuração associada no campo de descrição, assim como os dados de configuração no campo de configuração.

```
struct configurationTransaction
{
    configurationIdentifier string
    versionIdentifier       string
    description             string
    configuration           string
    transactionType         string
    transactionName         string
```

```
    issuer                    string
}
initConfiguration (configuration <description,configuration,name,issuer
    >){
    if configuration is not unique or configuration is not well-formed:
        return error
    putState (configuration.name, configuration)
    return success
}
```

Listing 4.2: Pseudo-código parcial para emitir transações de configuração. O campo configurationIdentifier contêm um identificador único para a configuração.

O protótipo usa os certificados de autoridade (*Certificate Authorities* - CA) do Hyperledger Fabric para criar e gerenciar certificados digitais em cada nó da rede de corrente de blocos. Certificados digitais garantem auditabilidade e que somente nós certificados e autorizados podem participar da rede de corrente de blocos.

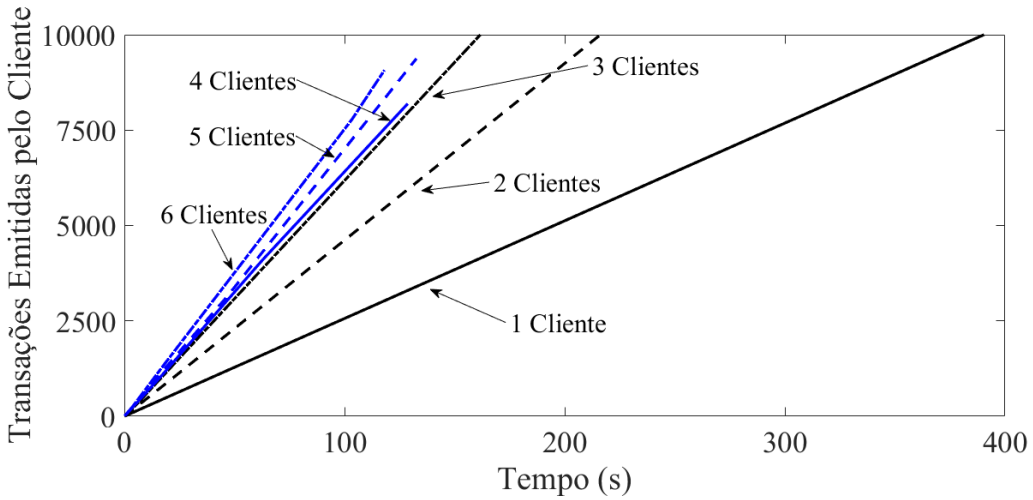


Figura 4.4: Tempo total decorrido para processar as transações na corrente de blocos à medida que o número de transações emitidas e o número de clientes aumentam. Os resultados mostram que a vazão aumenta com o número de clientes que emitem transações.

O protótipo implementa cada nó da rede do Hyperledger Fabric como um contêiner em um único computador e envia transações simultaneamente. O trabalho avalia a vazão de transação enviando um número crescente de transações e medindo o tempo decorrido que os clientes precisam para propor todas as transações para a rede baseada em corrente de blocos. A Figura 4.4 apresenta o resultado da avaliação da taxa de transação do cliente. A taxa de transação na fatia de rede atinge um valor de pico de 71,31 transações por segundo no lado do cliente.

O trabalho ajusta as configurações de criação de bloco de acordo com avaliações de desempenho realizadas anteriormente no Hyperledger Fabric [98, 99]. O trabalho

define o tamanho preferido do bloco sem cabeçalho em 99 MB, o número máximo de transações em um bloco em 10 e o tempo limite para inicializar uma rodada de consenso em dois segundos. Se alguma das condições for atendida, o nó ordenador inicia uma nova rodada de consenso e envia a proposta de novo bloco. Depois de alcançar o consenso, os nós acrescentam o novo bloco à sua cópia da corrente de blocos.

A tecnologia de fatia de redes fornece serviços fim-a-fim customizados encadeando funções virtuais de rede entre infraestruturas de nuvens concorrentes em um ambiente distribuído multi-inquilino e multi-domínio sem confiança entre os pares. A alta programabilidade resultante proveniente da virtualização da rede expõe todo o tráfego a um maior número de ameaças. Portanto, é obrigatório precisamente definir e localizar falhas e usos indevidos para identificar agentes maliciosos que podem comprometer simultaneamente o bom comportamento e a qualidade de serviço de milhares de usuários.

O trabalho propõe uma arquitetura baseada em corrente de blocos para proteger fatias de redes customizadas. A multiplicidade de características diferentes exigidas por cada fatia impõe o uso de várias correntes de blocos personalizadas com diferentes números de participantes, estruturas de dados, tipo de transações, taxa de transferência de transação, tipo de consenso, tipo de redes, etc.

O trabalho implementa um protótipo de prova de conceito usando duas correntes de blocos que garantem proteger a criação de fatia e a atualização e migração de funções virtualizadas de rede. O trabalho usa a plataforma Hyperledger Fabric que facilita a criação de várias correntes de blocos em diferentes canais. Para trabalhos futuros, pretende-se usar estruturas de dados otimizadas para melhorar a taxa de transação e usar um protocolo de consenso diferente.

Capítulo 5

Trabalhos Relacionados

Diversos trabalhos exploram o estado da arte de corrente de blocos aplicada a problemas de redes de comunicação e redes de quinta geração (5G) [100–106]. Os trabalhos se concentram no uso de corrente de blocos como um repositório de dados incrementais replicados, no qual todas as transações realizadas são assinadas e registradas com criptografia de chaves assimétricas. Yahiatene *et al.* e Ortega *et al.* propõem o uso de corrente de blocos como um mecanismo para fornecer segurança em redes veiculares [100, 101]. Eles argumentam que a corrente de blocos pode fornecer autenticidade e confiança com baixa latência para permitir redes seguras. Thuemler *et al.* e Capossele *et al.* discutem os requisitos necessários ao 5G para prover saúde eletrônica (*e-Health*) com base em experiências reais [102, 103]. Os resultados do artigo mostram que a corrente de blocos fornece a privacidade e proteção de dados necessários para proteger registros médicos em um ambiente sem confiança. Rawat *et al.* propõem uma solução baseada em corrente de blocos para redes sem fio virtuais que permite a confiança em provedores de nuvem [104]. Rosa e Rothemberg fornecem diretrizes para incorporar os aplicativos distribuídos baseados em corrente de blocos em cenários com múltiplos domínios administrativos [105]. Boudguiga *et al.* apresentam uma solução para atualizar dispositivos IoT através de informações armazenadas no corrente de blocos [106]. A proposta apresentada nesta dissertação fornece uma taxonomia de fatias baseadas em corrente de blocos que engloba e generaliza todas as aplicações de corrente de blocos como casos de uso em fatias de rede.

Outros trabalhos investigam o problema de vulnerabilidades de segurança em ambientes NFV multi-inquilino e multi-domínio [107, 108]. Eles mostram que a confiança nos provedores de nuvem é incerta e que o comprometimento de uma única VNF no núcleo da rede põe em risco todo o serviço fim-a-fim. Zaowat e Hasan propuseram o SECAP, uma estrutura baseada em corrente de blocos para armazenar com segurança uma árvore de proveniência de aplicativos em nuvem [109]. A estrutura protege os registros das mudanças de estado do aplicativo. Bozic *et al.*

propõem uma arquitetura para gerenciar estados de execução de máquinas virtuais usando um sistema baseado em corrente de blocos [47]. O sistema usa uma estrutura de corrente de blocos para registrar as instruções do hipervisor de virtualização do sistema na forma de transações.

Com relação à segurança do fatiamento de rede, Bordel *et al.* propõem uma solução baseada em geradores de números pseudo-aleatórios para fornecer segurança dentro de fatias de rede para dispositivos IoT e estações-base em sistemas 5G [110]. Khettab *et al.* propõem utilizar tecnologias NFV e SDN para proteger fatias de rede de múltiplos domínios instanciando funções de rede de segurança, como *firewalls* e sistemas de detecção de intrusão [111]. Os trabalhos, no entanto, não abordam possíveis comportamentos maliciosos de administradores de rede.

Outros trabalhos propõem o uso de corrente de blocos para prover confiança nos administradores de rede e intermediários responsáveis pelo fatiamento da rede. Valtanen *et al.* analisam o uso de corrente de blocos em provedores de fatia de rede para coletar, configurar e alocar recursos em automação industrial [112]. O artigo aponta as vantagens de usar corrente de blocos em casos de uso do 5G. Backman *et al.* propõem o uso de corrente de blocos para gerenciar recursos de rede 5G virtualizados em um cenário de múltiplos administradores [113].

Esta dissertação propõe uma arquitetura para construção de correntes de blocos que protegem e permitem a auditabilidade da criação de fatias de redes e também a atualização das funções de rede das fatias isoladas. A proposta atende ao ambiente multi-inquilino e multi-domínio sem confiança entre os pares.

Capítulo 6

Conclusões

A tecnologia de fatia de redes fornece serviços fim-a-fim customizados através encadeando funções virtuais de rede entre infraestruturas de nuvens concorrentes em um ambiente distribuído multi-inquilino e multi-domínio sem confiança entre os pares. A alta programabilidade resultante proveniente da virtualização da rede expõe todo o tráfego a um maior número de ameaças. Portanto, é obrigatório precisamente definir e localizar falhas e usos indevidos para identificar agentes maliciosos que podem comprometer simultaneamente o bom comportamento e a qualidade de serviço de milhares de usuários.

Esta dissertação propôs uma arquitetura baseada em corrente de blocos para proteger fatias de redes customizadas. A multiplicidade de características diferentes exigidas por cada fatia impõe o uso de várias correntes de blocos personalizadas com diferentes números de participantes, estruturas de dados, tipo de transações, taxa de transferência de transação, tipo de consenso, tipo de redes, etc. O trabalho implementa um protótipo de prova de conceito usando duas correntes de blocos que garantem proteger a criação de fatia e a atualização e migração de funções virtualizadas de rede. O trabalho usa a plataforma Hyperledger Fabric que facilita a criação de várias correntes de blocos em diferentes canais. Para trabalhos futuros, estruturas de dados otimizadas serão usadas para melhorar a taxa de transação e usar um protocolo de consenso diferente.

No entanto, a tecnologia de corrente de blocos é muito recente, pois tem apenas dez anos. O fato desta tecnologia prover uma camada de confiança distribuída faz com que ela seja considerada disruptiva e também a tecnologia mais importante depois da Internet. As criptomoedas já são um sucesso e existem predições que afirmam que continuará a crescer fortemente o volume de "dinheiro virtual"registrado em correntes de blocos. Os parágrafos a seguir abordam os principais desafios e problemas em aberto das correntes de blocos assim como as oportunidades de pesquisa na área.

6.1 Oportunidades e Aplicações da Tecnologia de Corrente de Blocos

Além da transferência de ativos utilizando criptomoedas, a tecnologia de corrente de blocos pode ser aplicada em diversas áreas. Praticamente em todas aplicações que requerem intermediários a corrente de blocos pode eliminar este intermediário provendo a camada de confiança distribuída. Também para fazer registros permanentes e imutáveis.

Papel da Corrente de Blocos em uma Economia Compartilhada - A economia do compartilhamento, ou compartilhada, é um modo de consumo onde bens e serviços não são de posse de um único usuário. Estes bens e serviços são temporariamente disponibilizados por outros indivíduos (terceiros), que recebem um incentivo financeiro para oferecer o serviço, através de uma plataforma que atua como intermediário entre o provedor do serviço e o consumidor. Os principais exemplos de empresas que utilizam este modelo são a Uber e a Airbnb, com outros exemplos incluindo: Cohealo, BlaBlaCar, JustPark, Skillshare, RelayRides e Landshare. Estas empresas conseguem seu lucro tomando uma parte dos ganhos dos usuários que provêm o serviço, ao fornecer uma plataforma capaz de conectar pessoas com interesses coincidentes.

A tecnologia de corrente de blocos permite prover confiança entre o provedor de serviço e o usuário do serviço sem necessidade de uma empresa intermediária. As empresas centralizadoras são assim eliminadas [114]. Além disso, contratos inteligentes permitem definir e impor regras para um acordo entre duas partes, possibilitando a fácil responsabilização de qualquer partícipe caso ocorra uma violação desse contrato. Plataformas de economia compartilhada implementadas com corrente de blocos são desse modo administradas coletivamente por todos seus usuários. Como é do interesse dos membros da plataforma que esta continue funcionando, há um incentivo para que os participantes atuem com honestidade.

Um exemplo de aplicação da corrente de blocos na economia compartilhada é a plataforma descentralizada de carona solidária Arcade City, que propõe um serviço similar ao Uber. Todo o processamento necessário para tratar do compartilhamento de caronas e outras funções é executado pela corrente de blocos. Como não existem intermediários, os custos das caronas podem ser reduzidos significativamente. Segurança é garantida usando um sistema de classificação no próprio aplicativo, garantindo que avaliações feitas por usuários encontrem-se sempre disponíveis e inalteradas graças a características da corrente de blocos.

Corrente de Blocos para Cidades Inteligentes - O uso de corrente de blocos em sistemas de Cidades Inteligentes possui restrições de segurança e privacidade que ainda são desafios [115]. As correntes de dados públicas, como o Bitcoin, efetuam

transações anônimas identificadas por chaves públicas. No entanto, a anonimidade não é absoluta, pois todas as transações são visíveis para todos os participantes da corrente de blocos e, assim, as atividades do usuário podem ser rastreadas. Combinando as informações dessas transações com alguns dados externos permite revelar a identidade real do usuário. Uma vez que a identidade do usuário é descoberta, todas suas ações serão rastreadas e dados confidencias, como padrões de compra e venda, serão vazados. Portanto, estes sistemas não garantem a privacidade dos dados dos usuários o que viola um dos princípios da segurança da informação [116].
O Uso de Corrente de Blocos na Área de Saúde - A alta produção e transferência de dados na área da saúde podem ser beneficiadas pelo uso de corrente de blocos. Sistemas baseados em corrente de blocos permite que diferentes serviços de saúde possam compartilhar e armazenar dados e registros médicos em uma rede permissionada [43]. Enquanto o uso de contratos inteligentes permite que pacientes controlem o acesso e a transferência de seus dados privados, a cópia da corrente de blocos em diferentes locais e a auditabilidade atendem à demanda de acesso de diferentes interessados ao mesmo documento.

A corrente de blocos também possui aplicações na rastreabilidade de dados na área da saúde. Sistemas baseados em corrente de blocos permitem o estabelecimento de uma rede privada e segura entre serviços de saúde para armazenar prescrições de remédios [5]. As mudanças no estado ou posição do medicamento, assim como a transferência de propriedade são registradas no livro-razão distribuído, formando um registro histórico de dados do remédio desde a origem. A auditabilidade permite o acesso das entidades credenciadas às informações, facilitando a identificação de medicamentos falsos ou desviados.

6.2 Atividades em Organismos de Normalização

A aplicabilidade da tecnologia de corrente de blocos tem se demonstrado enorme. Assim, é essencial o pronto estabelecimento de normas internacionais para prover diretrizes relacionadas à uma nova tecnologia que propiciem um maior envolvimento e mais investimentos das indústrias. Em relação aos tópicos abordados nesta dissertação diferentes organismos de normalização criaram grupos de trabalho com objetivos variados.

A organização internacional de normalização (*International Organization for Standartization* - ISO) criou um comitê técnico que visa normalizar a tecnologia de corrente de blocos e a tecnologia de livro-razão distribuído [117]. O comitê possui projetos para formalizar os riscos de segurança, ameaças e vulnerabilidades da tecnologia, além de normalizar a arquitetura de referência, taxonomia, contratos inteligentes e a proteção de privacidade e de informações pessoais. A *Internet Rese-*

arch Task Force (IRTF) criou o grupo de pesquisa da infraestrutura descentralizada da Internet (*Decentralized Internet Infrastructure Research Group* - DINRG) que estuda os serviços de infraestrutura beneficiados pela descentralização [118]. A *World Wide Web Consortium* (W3C) criou o grupo da comunidade de corrente de blocos (*Blockchain Community Group*) que tem como objetivo normalizar os formatos das mensagens em sistemas baseados em corrente de blocos [119]. A comunidade usa o formato de mensagem definido na ISO 20022 como base para as normas. O grupo também busca definir diretrizes para o uso de armazenamento. A união internacional de telecomunicações (*International Telecommunications Union* - ITU), através do grupo de foco na aplicação da tecnologia de livro-razão distribuído (*Focus Group on Applications of Distributed Ledger Technology* - ITU-T FG DLT), pretende normalizar os serviços interoperáveis baseados na tecnologia de livro-razão distribuído [120]. O grupo de trabalho de corrente de blocos da Europa (*Europe Blockchain Working Group*) da associação internacional de segurança para comunicação institucional do comércio (*International Securities Association Trade Communication* - ISITC) discute a adoção da tecnologia de livro-razão distribuído pela indústria de serviços financeiros [121].

6.3 Desafios e Problemas em Aberto

Os desafios em corrente de blocos são enormes: escalabilidade, vazão de transações, segurança e *software*, algoritmos criptográficos eficientes, entre outros.

Escalabilidade e Vazão de Transações - Todos os protocolos de consenso e tipos de corrente de blocos apresentados possuem limitações em termos de taxa de transferência, latência de transação ou quantidade de nós [28]. Os sistemas baseados em correntes de blocos ainda são incapazes de atingir o desempenho dos atuais sistemas centralizados de transferência de ativos. Enquanto as plataformas do PayPal e Visa processam aproximadamente 2000 transações por segundo em média e até 56.000 transações em pico [122, 123] com tempos de resposta da ordem de segundos [16], o Ethereum processa um máximo de 20 transações por segundo e o Bitcoin atinge uma vazão de apenas 7 transações por segundo. Ainda, o processo de mineração da prova de trabalho no Bitcoin gera gastos insustentáveis de energia sem retorno proporcional, atingindo em média 50 TW consumidos por ano [17]. Algumas correntes de blocos permissionadas atingem taxas da ordem de milhares de transações por segundo, porém reduzem o número de participantes possíveis [61, 88, 124].

Outro desafio de correntes de blocos é a eficiência no armazenamento e na busca de transações, pois, para prover segurança descentralizada, é necessário que todos os participantes do consenso processem todas as transações da rede e armazenem todo

o histórico de transações. A verificação de transações é fundamental para garantir segurança em um ambiente sem confiança entre os pares como o de corrente de blocos. No entanto, verificar a existência ou corretude de uma transação pode envolver uma busca custosa em uma lista crescente de todas as transações presentes em todos os blocos da rede. Além disso, a quantidade de dados armazenados em cada participante da corrente de blocos cresce constantemente, uma vez que nenhuma transação ou bloco jamais é descartado. As principais implementações de correntes de blocos implementam estruturas auxiliares chamadas de árvores de Merkle para contornar o problema[1]. O trilema entre descentralização, segurança e escalabilidade dificulta que os sistemas baseados em correntes de blocos atinjam o desempenho de sistemas atuais e que atendam às necessidades do futuro.

Revogação de Chaves criptográficas - Um dos principais desafios de sistemas baseados em correntes de blocos é a perda de chaves privadas, pois todas as transferências de ativos na rede são realizadas através de transações assinadas. A ausência de uma autoridade central confiável que armazene as identidades dos participantes da rede dificulta a revogação de chaves perdidas ou roubadas e consiste em um risco constante de perda de ativos. Em especial, é impossível revogar uma chave perdida em correntes de blocos públicas, nas quais o único identificador de um usuário é um endereço baseado na sua chave pública correspondente. Estima-se que mais de 30% dos *bitcoins* minerados estão inutilizados devido à perda de chaves, totalizando uma perda média de mais de 1000 *bitcoins* ou 8 milhões de dólares por dia [125].

Gerenciadores de chaves de correntes de blocos públicas implementam mecanismos de múltiplas assinaturas (*multisignature* ou *multisig*) para prevenir a perda de ativos decorrente da perda de uma chave privada. O mecanismo *multisignature* cria 3 pares de chaves assimétricas e exige a assinatura de pelo menos duas para emitir uma transação. As chaves podem ser armazenadas em locais diferentes e, possivelmente, controladas por entidades diferentes. Caso uma chave seja perdida, é possível assinar com as duas restantes. Nenhuma chave pode emitir uma transação individualmente. O mecanismo permite tolerar perdas de até uma chave ou repartir a posse de um ativo, pois, caso as chaves pertençam a entidades diferentes, pelo menos duas entidades devem concordar para emitir uma transação. Caso uma chave seja perdida no mecanismo de assinaturas múltiplas, basta criar um novo endereço e emitir uma transação assinada transferindo o ativo com as duas chaves restantes. Sistemas baseados em correntes de blocos permissionadas podem implementar mecanismos baseados em identidades pessoais para criar listas locais de revogação de chaves que devem ser alteradas através de consenso [40].

Segurança da Corrente de Blocos - Falhas de segurança em *software* de corrente de blocos podem causar um desastre. Existe uma enorme carência de profissionais

[1]A estrutura e fundamentos de árvores de Merkle são apresentados no Apêndice A.2.

que programem de forma segura e de ferramentas que consigam analisar, validar e testar a segurança de um *software* de corrente de blocos. Existem mais de 700 criptomoedas, no entanto, não existe nenhuma garantia que os softwares usando nas criptomoedas é seguro. Até mesmo, a maioria das implementações dos algoritmos de consenso não passaram por algum tipo de prova. Uma falha em um contrato inteligente pode ser catastrófica. Um exemplo foi a falha da chamada recursiva (*recursive call bug*) no acesso ao objeto (*data access object* - DAO) que causou o roubo de mais de 50 milhões de dólares no Ethereum[2].

No consenso com prova de trabalho (*Proof of Work* - PoW) usado no Bitcoin e Ethereum existe o conhecido ataque de 51% ou ataque de maioria, que se refere quando um atacante ou grupo de atacantes possui mais de 50% do poder computacional da rede, porque, neste caso, os atacantes podem fazer gasto duplo. Embora um ataque de 51% nunca tenha sido executado com sucesso no Bitcoin[3], no entanto, ele foi demonstrado que funciona em moedas alternativas do bitcoin (*altcoin*)[4,5].

Ainda em relação ao mecanismo de consenso por prova de trabalho, Eyal and Sirer da Universidade de Cornell apresentaram em 2013 o ataque de mineração egoísta (*selfish mining*) [126]. Este ataque se aplica ao consenso de prova de trabalho quando mineradores em conluio procuram aumentar seus lucros para receberem mais incentivos. A ideia chave deste ataque é que um conjunto de mineradores em conluio achem o resultado do desafio para formar um bloco, mas mantém a o resultado em segredo, forçando uma bifurcação de forma intencional. O conjunto de mineradores em conluio continuam minerando o ramo da bifurcação que eles forçaram sem divulgar novos blocos, enquanto mineradores honestos vão minerar o outro ramo. Caso os mineradores em conluio, mais de 25% da capacidade total, resolvam mais desafios, e não divulguem o resultado, eles passam a ter mais vantagem para conseguir obter o ramo mais longo. Quando o ramo minerado de forma pública pelos mineradores honestos se aproxima em tamanho do ramo minerado pelos malfeitores em conluio, os mineradores egoístas divulgam os blocos. Os mineradores honestos perderão dinheiro porque o ramo que eles mineraram não vai vingar.

Assiste-se hoje uma colaboração das indústrias sem precedentes mesmo entre empresas que eram concorrentes. O interesse na tecnologia de corrente de blocos cresce exponencialmente e o profissionais especialistas desta área estão muito requi-

[2]https://medium.com/@MyPaoG/explaining-the-dao-exploit-for-beginners-in-solidity-80ee84f0d470

[3]O conjunto de mineradores "Ghash.io"ultrapassou 50% do poder computacional da rede bitcoin em julho de 2014. Este conjunto de mineradores se comprometeu a reduzir o poder computacional para um valor sempre menor que 40%.

[4]A moeda Bitcoin Gold, na época a 26^a maior moeda, sofreu um ataque de 51% em maio de 2018. Os atacantes efetuaram gasto duplo por diversos dias e roubaram mais de US$18 milhões em Bitcoin Gold.

[5]As correntes de blocos Krypton e Shift, baseadas em Ethereum, sofreram ataques de 51% em agosto de 2016.

sitados e estão entre os mais bem pagos. Muitos desafios ainda persistem e muitos avanços são esperados para os próximos anos. Com certeza é uma área com enormes perspectivas e oportunidades.

A corrente de blocos pode resolver problemas em diversos setores da sociedade como economia, saúde, transporte, educação, entre outros. A tecnologia de corrente de blocos é uma poderosa ferramenta que oferece muitas possibilidades de atacar problemas de injustiça, desigualdade, carência, entre outras. No entanto, as soluções para problemas humanos na maioria das vezes não são técnicas, mas sim humanas. A corrente de blocos não corrige seres humanos.

Referências Bibliográficas

[1] NAKAMOTO, S. "Bitcoin: A peer-to-peer electronic cash system". 2008. Disponível em: <https://bitcoin.org/bitcoin.pdf>. Acessado em 31 de agosto de 2019.

[2] WOOD, G. "Ethereum: A secure decentralised generalised transaction ledger". 2014. Disponível em: <http://bitcoinaffiliatelist.com/wp-content/uploads/ethereum.pdf>. Acessado em 31 de agosto de 2019.

[3] COIN MARKET. *Cryptocurrency Global Charts: Total Market Capitalization*. Relatório técnico, Coin Market, 2019. Disponível em: <https://coinmarketcap.com/charts/>. Acessado em 31 de agosto de 2019.

[4] INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATÍSTICA (IBGE). "Produto Interno Bruto - PIB". 2019. Disponível em: <https://www.ibge.gov.br/explica/pib.php>. Acessado em 31 de agosto de 2019.

[5] ZHANG, P., SCHMIDT, D. C., WHITE, J., et al. "Chapter One - Blockchain Technology Use Cases in Healthcare". In: Raj, P., Deka, G. C. (Eds.), *Blockchain Technology: Platforms, Tools and Use Cases*, v. 111, *Advances in Computers*, Elsevier, pp. 1 – 41, 2018. doi: https://doi.org/10.1016/bs.adcom.2018.03.006. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0065245818300196>.

[6] ESPOSITO, C., DE SANTIS, A., TORTORA, G., et al. "Blockchain: A Panacea for Healthcare Cloud-Based Data Security and Privacy?" *IEEE Cloud Computing*, v. 5, n. 1, pp. 31–37, Jan 2018. ISSN: 2325-6095. doi: 10.1109/MCC.2018.011791712.

[7] XIE, J., TANG, H., HUANG, T., et al. "A Survey of Blockchain Technology Applied to Smart Cities: Research Issues and Challenges", *IEEE Communications Surveys Tutorials*, 2019. ISSN: 1553-877X. doi: 10.1109/COMST.2019.2899617.

[8] RAHMAN, M. A., RASHID, M. M., HOSSAIN, M. S., et al. "Blockchain and IoT-Based Cognitive Edge Framework for Sharing Economy Services in a Smart City", *IEEE Access*, v. 7, pp. 18611–18621, 2019. ISSN: 2169-3536. doi: 10.1109/ACCESS.2019.2896065.

[9] MORA, O. B., RIVERA, R., LARIOS, V. M., et al. "A Use Case in Cybersecurity based in Blockchain to deal with the security and privacy of citizens and Smart Cities Cyberinfrastructures". In: *2018 IEEE International Smart Cities Conference (ISC2)*, pp. 1–4, Sep. 2018. doi: 10.1109/ISC2.2018.8656694.

[10] PIERONI, A., SCARPATO, N., DI NUNZIO, L., et al. "Smarter City: Smart Energy Grid Based on Blockchain Technology", *International Journal on Advanced Science, Engineering and Information Technology*, v. 8, n. 1, pp. 298–306, 2018.

[11] GLASER, F. "Pervasive decentralisation of digital infrastructures: a framework for blockchain enabled system and use case analysis". In: *50th Hawaii International Conference on System Sciences*, 2017.

[12] CACHIN, C., VUKOLIĆ, M. "Blockchains Consensus Protocols in the Wild", *arXiv preprint arXiv:1707.01873*, 2017.

[13] XIAO, Y., ZHANG, N., LOU, W., et al. "A Survey of Distributed Consensus Protocols for Blockchain Networks", *CoRR*, v. abs/1904.04098, 2019. Disponível em: <`http://arxiv.org/abs/1904.04098`>.

[14] POPOV, S. "The tangle". 2016. Disponível em: <`https://iota.org/IOTAWhitepaper.pdf`>. Acessado em 31 de agosto de 2019.

[15] EYAL, I., GENCER, A. E., SIRER, E. G., et al. "Bitcoin-ng: A scalable blockchain protocol". In: *13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)*, pp. 45–59, 2016.

[16] BITCOINWIKI. "Bitcoin Scalability". 2019. Disponível em: <`https://en.bitcoin.it/wiki/Scalability`>. Acessado em 31 de agosto de 2019.

[17] DIGICONOMIST. "Bitcoin Energy Consumption Index". 2019. Disponível em: <`https://digiconomist.net/bitcoin-energy-consumption`>. Acessado em 31 de agosto de 2019.

[18] ELETROBRAS. *Relatórios de Sustentabilidade Socioambiental.* Relatório técnico, Eletrobras S.A., 2017. Disponível em: <`http://www.eletronuclear.gov.br/Quem-Somos/Governanca/Documents/`

Relat%C3%B3rios%20e%20Balan%C3%A7os/Relat%C3%B3rios%20de%20Sustentabilidade/RelSustentabilidade2017_Completo_DE_REV5.pdf>. Acessado em 31 de agosto de 2019.

[19] MINISTÉRIO DE MINAS E ENERGIA. *Anuário Estatístico de Energia Elétrica 2017*. Relatório técnico, Ministério de Minas e Energia do Brasil, 2017. Disponível em: <http://epe.gov.br/sitespt/publicacoes-dados-abertos/publicacoes/PublicacoesArquivos/publicacao-160/topico-168/Anuario2017vf.pdf>. Acessado em 31 de agosto de 2019.

[20] NXT COMMUNITY. "NXT whitepaper". 2014. Disponível em: <https://nxtwiki.org/wiki/Whitepaper:Nxt>. Acessado em 31 de agosto de 2019.

[21] KING, S., NADAL, S. "PPCoin: Peer-to-peer crypto-currency with proof-of-stake", *self-published paper, August*, v. 19, 2012.

[22] ONGARO, D., OUSTERHOUT, J. "In Search of an Understandable Consensus Algorithm". In: *2014 USENIX Annual Technical Conference (USENIX ATC 14)*, pp. 305–319, Philadelphia, PA, 2014. USENIX Association. ISBN: 978-1-931971-10-2.

[23] LAMPORT, L. "The Part-Time Parliament", *ACM Transactions Computer Systems*, v. 16, n. 2, pp. 133–169, maio 1998. ISSN: 0734-2071.

[24] CASTRO, M., LISKOV, B. "Practical Byzantine Fault Tolerance". In: *Proceedings of the Third Symposium on Operating Systems Design and Implementation*, OSDI '99, pp. 173–186, Berkeley, CA, USA, 1999. USENIX Association. ISBN: 1-880446-39-1.

[25] BESSANI, A., SOUSA, J., ALCHIERI, E. "State machine replication for the masses with BFT-SMaRt". In: *IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2014.

[26] BUTERIN, V., GRIFFITH, V. "Casper the friendly finality gadget", *arXiv preprint arXiv:1710.09437*, 2017.

[27] KWON, J. "Tendermint: Consensus without mining". 2014. Disponível em: <https://tendermint.com/static/docs/tendermint.pdf>. Acessado em 31 de agosto de 2019.

[28] POPOV, S. "The Tangle", *cit. on*, p. 131, 2017. Disponível em: <http://www.descryptions.com/Iota.pdf>. Acessado em 31 de agosto de 2019.

[29] REBELLO, G. A. F., CAMILO, G. F., SILVA, L. G. C., et al. "Segurança na Internet do Futuro: Provendo Confiança Distribuída através de Correntes de Blocos na Virtualização de Funções de Rede". In: *Minicursos do SBRC'2019*, 2019.

[30] GREVE, F., SAMPAIO, L., ABIJAUDE, J., et al. "Blockchain e a Revolução do Consenso sob Demanda". In: *Minicursos do SBRC'2018*, v. 36, 2018.

[31] CHOHAN, U. W. "The Double Spending Problem and Cryptocurrencies". 2017. Disponível em: <http://dx.doi.org/10.2139/ssrn.3090174>. Acessado em 31 de agosto de 2019.

[32] RYAN, M. D. "Digital Cash". 2006. Disponível em: <http://www.cs.bham.ac.uk/~mdr/teaching/modules06/netsec/lectures/DigitalCash.html>. Acessado em 31 de agosto de 2019.

[33] CHAUM, D. "Blind signatures for untraceable payments". In: *Advances in cryptology*, pp. 199–203. Springer, Hong-Ning, 1983.

[34] DAI, W. "B-money". 1998. Disponível em: <http://www.weidai.com/bmoney.txt>. Acessado em 31 de agosto de 2019.

[35] BACK, A. "Hashcash - a denial of service counter-measure". 2002. Disponível em: <http://www.hashcash.org/papers/hashcash.pdf>. Acessado em 31 de agosto de 2019.

[36] FINNEY, H. "RPOW: Reusable Proofs of Work". 2005. Disponível em: <http://nakamotoinstitute.org/finney/rpow/theory.html>. Acessado em 31 de agosto de 2019.

[37] WÜST, K., GERVAIS, A. "Do you need a Blockchain?" In: *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*, pp. 45–54. IEEE, 2018.

[38] EYAL, I., SIRER, E. G. "Majority is Not Enough: Bitcoin Mining is Vulnerable", *Commun. ACM*, v. 61, n. 7, pp. 95–102, 2018. ISSN: 0001-0782. doi: 10.1145/3212998. Disponível em: <http://doi.acm.org/10.1145/3212998>.

[39] FRANTZ, C. K., NOWOSTAWSKI, M. "From institutions to code: Towards automated generation of smart contracts". In: *2016 IEEE 1st International Workshops on Foundations and Applications of Self* Systems (FAS* W)*, pp. 210–215. IEEE, 2016.

[40] ANDROULAKI, E., BARGER, A., BORTNIKOV, V., et al. "Hyperledger Fabric: a distributed operating system for permissioned blockchains". In: *Proceedings of the Thirteenth EuroSys Conference*, p. 30. ACM, 2018.

[41] WILKINSON, S., BOSHEVSKI, T., BRANDOFF, J., et al. "Storj: a peer-to-peer cloud storage network". 2018. Disponível em: <`https://storj.io/storj.pdf`>. Acessado em 31 de agosto de 2019.

[42] ALI, M., NELSON, J. C., SHEA, R., et al. "Blockstack: A Global Naming and Storage System Secured by Blockchains." In: *USENIX Annual Technical Conference*, pp. 181–194, 2016.

[43] AZARIA, A., EKBLAW, A., VIEIRA, T., et al. "MedRec: Using blockchain for medical data access and permission management". In: *Open and Big Data (OBD), International Conference on*, pp. 25–30. IEEE, 2016.

[44] LEE, K., JAMES, J. I., EJETA, T. G., et al. "Electronic voting service using block-chain", *Journal of Digital Forensics, Security and Law*, v. 11, n. 2, pp. 8, 2016.

[45] CHRISTIDIS, K., DEVETSIKIOTIS, M. "Blockchains and smart contracts for the Internet of Things", *IEEE Access*, v. 4, pp. 2292–2303, 2016.

[46] HUH, S., CHO, S., KIM, S. "Managing IoT Devices Using Blockchain Platform". In: *Advanced Communication Technology (ICACT), 2017 19th International Conference on*, pp. 464–467. IEEE, 2017.

[47] BOZIC, N., PUJOLLE, G., SECCI, S. "Securing Virtual Machine Orchestration with Blockchains". In: *1st Cyber Security in Networking Conference*, 2017.

[48] ALVARENGA, I. D., REBELLO, G. A. F., DUARTE, O. C. M. B. "Securing Management, Configuration, and Migration of Virtual Network Functions Using Blockchain". In: *IEEE/IFIP NOMS*, 2018.

[49] ZHENG, Z., XIE, S., DAI, H.-N., et al. "Blockchain Challenges and Opportunities: A Survey", *International Journal of Web and Grid Services*, v. 14, n. 4, pp. 352–375, 2018.

[50] XU, X., WEBER, I., STAPLES, M., et al. "A Taxonomy of Blockchain-Based Systems for Architecture Design". In: *2017 IEEE International Conference on Software Architecture (ICSA)*, pp. 243–252, April 2017. doi: 10.1109/ICSA.2017.33.

[51] SMOLENSK, M. "Lightstreams White Paper". 2018. Disponível em: <https://s3.amazonaws.com/lightstreams/lightstreams_whitepaper.pdf>. Acessado em 31 de agosto de 2019.

[52] VUKOLIĆ, M. "The Quest for Scalable Blockchain Fabric: Proof-of-Work vs. BFT Replication". In: *Open Problems in Network Security: IFIP WG 11.4 International Workshop, iNetSec 2015, Zurich, Switzerland, October 29, 2015, Revised Selected Papers*, pp. 112–125, Cham, Springer International Publishing, 2016. ISBN: 978-3-319-39028-4. doi: 10.1007/978-3-319-39028-4_9.

[53] DOUCEUR, J. R. "The sybil attack". In: *International Workshop on Peer-to-Peer Systems*, pp. 251–260. Springer, 2002.

[54] ATTIYA, H., BAR-NOY, A., DOLEV, D. "Sharing memory robustly in message-passing systems", *Journal of the ACM (JACM)*, v. 42, n. 1, pp. 124–142, 1995.

[55] HADZILACOS, V., TOUEG, S. *A Modular Approach to the Specification and Implementation of Fault-Tolerant Broadcasts*. Relatório técnico, Department of Computer Science, Cornell University, New York - USA, maio 1994.

[56] FISCHER, M. J., LYNCH, N. A., PATERSON, M. S. *Impossibility of distributed consensus with one faulty process.* Relatório técnico, Massachusetts Institute of Technology, 1982.

[57] LAMPORT, L., SHOSTAK, R., PEASE, M. "The Byzantine Generals Problem", *ACM Transactions on Programming Languages and Systems*, v. 4, n. 3, pp. 382–401, jul. 1982. ISSN: 0164-0925. doi: 10.1145/357172.357176. Disponível em: <http://doi.acm.org/10.1145/357172.357176>.

[58] SCHNEIDER, F. B. "Chapter 7: Replication Management Using the State-machine Approach. Distributed Systems, 2nd edn." 1993.

[59] BREWER, E. A. "Towards Robust Distributed Systems". In: *Proceedings of the Nineteenth Annual ACM Symposium on Principles of Distributed Computing*, New York, NY, USA, 2000. ACM. ISBN: 1-58113-183-6. doi: 10.1145/343477.343502. Disponível em: <http://doi.acm.org/10.1145/343477.343502>.

[60] SCHWARTZ, D., YOUNGS, N., BRITTO, A., et al. "The ripple protocol consensus algorithm", *Ripple Labs Inc White Paper*, v. 5, 2014.

[61] BUCHMAN, E. *Tendermint: Byzantine fault tolerance in the age of blockchains*. Tese de Doutorado, University of Guelph, 2016.

[62] MILLER, A., XIA, Y., CROMAN, K., et al. "The Honey Badger of BFT Protocols". In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pp. 31–42, New York, NY, USA, 2016. ACM. ISBN: 978-1-4503-4139-4. doi: 10.1145/2976749.2978399. Disponível em: <http://doi.acm.org/10.1145/2976749.2978399>.

[63] SOUSA, J., BESSANI, A., VUKOLIC, M. "A Byzantine Fault-Tolerant Ordering Service for the Hyperledger Fabric Blockchain Platform". In: *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 51–58, June 2018. doi: 10.1109/DSN.2018.00018.

[64] CACHIN, C., VUKOLIĆ, M. "Blockchain consensus protocols in the wild", *arXiv preprint arXiv:1707.01873*, 2017.

[65] ONGARO, D., OUSTERHOUT, J. "In search of an understandable consensus algorithm". In: *2014 USENIX Annual Technical Conference (USENIX ATC 14)*, pp. 305–319, 2014.

[66] DOLEV, D. "The Byzantine generals strike again", *Journal of algorithms*, v. 3, n. 1, pp. 14–30, 1982.

[67] HYPERLEDGER. "Hyperledger Fabric FAQ". 2018. Disponível em: <https://hyperledger-fabric.readthedocs.io/en/release-1.4/Fabric-FAQ.html#bft>. Acessado em 31 de agosto de 2019.

[68] BANO, S., SONNINO, A., AL-BASSAM, M., et al. "Consensus in the Age of Blockchains", *CoRR*, v. abs/1711.03936, 2017. Disponível em: <http://arxiv.org/abs/1711.03936>.

[69] DWORK, C., LYNCH, N., STOCKMEYER, L. "Consensus in the presence of partial synchrony", *Journal of the ACM (JACM)*, v. 35, n. 2, pp. 288–323, 1988.

[70] TSENG, L. "Bitcoin's consistency property". In: *2017 IEEE 22nd Pacific Rim International Symposium on Dependable Computing (PRDC)*, pp. 219–220. IEEE, 2017.

[71] DECKER, C., SEIDEL, J., WATTENHOFER, R. "Bitcoin meets strong consistency". In: *Proceedings of the 17th International Conference on Distributed Computing and Networking*, p. 13. ACM, 2016.

[72] REBELLO, G. A. F., CAMILO, G. F., SILVA, L. G. C., et al. "Correntes de Blocos: Algoritmos de Consenso e Implementação na Plataforma Hyperledger Fabric". In: *38º Jornada de Atualização em Informática (JAI) do XXXIX Congresso da Sociedade Brasileira de Computação (CSBC 2019)*, 2019.

[73] VASIN, P. "Blackcoin's Proof-of-Stake Protocol v2", *URL: https://blackcoin.co/blackcoin-pos-protocol-v2-whitepaper. pdf*, v. 71, 2014.

[74] OLSON, K., BOWMAN, M., MITCHELL, J., et al. "Sawtooth: An Introduction", *The Linux Foundation*, 2018.

[75] BUTERIN, V. "Proof-of-Stake FAQ". 2019. Disponível em: <`https://github.com/ethereum/wiki/wiki/Proof-of-Stake-FAQ`>. Acessado em 31 de agosto de 2019.

[76] TIKHOMIROV, S. "Ethereum: State of Knowledge and Research Perspectives". In: *Foundations and Practice of Security*, pp. 206–221. Springer International Publishing, 2018. ISBN: 978-3-319-75650-9.

[77] WANG, W., HOANG, D. T., XIONG, Z., et al. "A Survey on Consensus Mechanisms and Mining Management in Blockchain Networks", *CoRR*, v. abs/1805.02707, 2018. Disponível em: <`http://arxiv.org/abs/1805.02707`>.

[78] BENTOV, I., LEE, C., MIZRAHI, A., et al. "Proof of Activity: Extending Bitcoin's Proof of Work via Proof of Stake." *IACR Cryptology ePrint Archive*, v. 2014, pp. 452, 2014.

[79] KIAYIAS, A., RUSSELL, A., DAVID, B., et al. "Ouroboros: A provably secure proof-of-stake blockchain protocol". In: *Annual International Cryptology Conference*, pp. 357–388. Springer, 2017.

[80] ANGELIS, S. D., ANIELLO, L., BALDONI, R., et al. "PBFT vs proof-of-authority: applying the CAP theorem to permissioned blockchain". In: *Italian Conference on Cyber Security (06/02/18)*, January 2018. Disponível em: <`https://eprints.soton.ac.uk/415083/`>.

[81] DINH, T. T. A., WANG, J., CHEN, G., et al. "Blockbench: A framework for analyzing private blockchains". In: *Proceedings of the 2017 ACM International Conference on Management of Data*, pp. 1085–1100. ACM, 2017.

[82] CACHIN, C. “Architecture of the Hyperledger blockchain fabric”. In: *Workshop on Distributed Cryptocurrencies and Consensus Ledgers*, 2016.

[83] CASTRO, M., LISKOV, B. “Practical Byzantine Fault-Tolerance and Proactive Recovery”, *ACM Transactions on Computer Systems*, v. 20, n. 4, pp. 398–461, nov. 2002.

[84] MAZIÈRES, D. *The Stellar Consensus Protocol: A Federated Model for Internet-level Consensus*. Relatório técnico, Stellar Devlopment Foundation white paper, Apr. 2015. Disponível em: <`https://www.stellar.org/papers/stellar-consensus-protocol.pdf`>. Acessado em 31 de agosto de 2019.

[85] LAMPORT, L. “Paxos Made Simple”, *ACM SIGACT News*, v. 32, n. 4, pp. 18–25, 2001.

[86] HOWARD, H. *ARC: Analysis of Raft Consensus*. Relatório Técnico UCAM-CL-TR-857, University of Cambridge, Computer Laboratory, Jul. 2014. Disponível em: <`https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-857.pdf`>.

[87] SCHNEIDER, F. B. “Implementing Fault-tolerant Services Using the State Machine Approach: A Tutorial”, *ACM Comput. Surv.*, v. 22, n. 4, pp. 299–319, dez. 1990. ISSN: 0360-0300. doi: 10.1145/98163.98167. Disponível em: <`http://doi.acm.org/10.1145/98163.98167`>.

[88] SCHWARTZ, D., YOUNGS, N., BRITTO, A. “The Ripple Protocol Consensus Algorithm”, *Ripple Labs Inc White Paper*, 2014. Disponível em: <`https://ripple.com/files/ripple_consensus_whitepaper.pdf`>. Acessado em 31 de agosto de 2019.

[89] TRUONG, N. B., UM, T., ZHOU, B., et al. “Strengthening the Blockchain-Based Internet of Value with Trust”. In: *2018 IEEE International Conference on Communications (ICC)*, pp. 1–7, May 2018. doi: 10.1109/ICC.2018.8423014.

[90] LARIMER, D. “EOS.IO White Paper”. 2017. Disponível em: <`https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md`>. Acessado em 31 de agosto de 2019.

[91] HALPERN, J., PIGNATARO, C. *Service Function Chaining (SFC) Architecture*. RFC 7665, RFC Editor, October 2015. Disponível em: <`http://www.rfc-editor.org/rfc/rfc7665.txt`>. Acessado em 31 de agosto de 2019.

[92] MEDHAT, A. M., TALEB, T., ELMANGOUSH, A., et al. "Service Function Chaining in Next Generation Networks: State of the Art and Research Challenges", *IEEE Communications Magazine*, v. 55, n. 2, pp. 216–223, fev. 2017. ISSN: 0163-6804.

[93] BHAMARE, D., JAIN, R., SAMAKA, M., et al. "A survey on service function chaining", *Journal of Network and Computer Applications*, v. 75, pp. 138–155, 2016.

[94] REBELLO, G. A. F., ALVARENGA, I. D., SANZ, I. J., et al. "BSec-NFVO: A Blockchain-based Security for Network Function Virtualization Orchestration". In: *IEEE International Conference on Communications (ICC)*, 2019.

[95] REBELLO, G. A. F., CAMILO, G. F., SILVA, L. G. C., et al. *Provendo uma Infraestrutura de Software Fatiada, Isolada e Segura de Funções Virtuais através da Tecnologia de Corrente de Blocos*. Relatório técnico, Grupo de Teleinformática e Automação (GTA/COPPE/UFRJ), 2019. Disponível em: <`https://www.gta.ufrj.br/ftp/gta/TechReports/RCS19.pdf`>.

[96] REBELLO, G. A. F., CAMILO, G. F., SILVA, L. G. C., et al. "Providing a Sliced, Secure, and Isolated Software Infrastructure of Virtual Functions Through Blockchain Technology". In: *IEEE International Conference on High Performance Switching and Routing (HPSR 2019)*, 2019.

[97] DOLEV, D., YAO, A. "On the security of public key protocols", *IEEE Transactions on information theory*, v. 29, n. 2, pp. 198–208, 1983.

[98] THAKKAR, P., NATHAN, S., VISWANATHAN, B. "Performance benchmarking and optimizing hyperledger fabric blockchain platform". In: *IEEE MASCOTS*, pp. 264–276, 2018.

[99] GORENFLO, C., LEE, S., GOLAB, L., et al. "FastFabric: Scaling Hyperledger Fabric to 20,000 Transactions per Second", https://arxiv.org/pdf/1901.00910.pdf, Jan. 2019.

[100] YAHIATENE, Y., RACHEDI, A. "Towards a Blockchain and Software-Defined Vehicular Networks Approaches to Secure Vehicular Social Network". In: *IEEE Conference on Standards for Communications and Networking (CSCN)*, pp. 1–7, 2018.

[101] ORTEGA, V., BOUCHMAL, F., MONSERRAT, J. F. "Trusted 5G vehicular networks: Blockchains and content-centric networking", *IEEE Vehicular Technology Magazine*, v. 13, n. 2, pp. 121–127, 2018.

[102] THUEMMLER, C., ROLFFS, C., BOLLMANN, A., et al. "Requirements for 5G based telemetric cardiac monitoring". In: *14th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pp. 1–4, 2018.

[103] CAPOSSELE, A., GAGLIONE, A., NATI, M., et al. "Leveraging blockchain to enable smart-health applications". In: *IEEE 4th International Forum on Research and Technology for Society and Industry (RTSI)*, pp. 1–6, 2018.

[104] RAWAT, D. B., ALSHAIKHI, A. "Leveraging Distributed Blockchain-based Scheme for Wireless Network Virtualization with Security and QoS Constraints". In: *International Conference on Computing, Networking and Communications (ICNC)*, pp. 332–336, 2018.

[105] ROSA, R., ROTHENBERG, C. E. "Blockchain-Based Decentralized Applications for Multiple Administrative Domain Networking", *IEEE Communications Standards Magazine*, v. 2, n. 3, pp. 29–37, 2018.

[106] BOUDGUIGA, A., BOUZERNA, N., GRANBOULAN, L., et al. "Towards better availability and accountability for IoT updates by means of a blockchain". In: *IEEE EuroS&PW*, pp. 50–58, 2017.

[107] PATTARANANTAKUL, M., HE, R., SONG, Q., et al. "NFV Security Survey: From Use Case Driven Threat Analysis to State-of-the-Art Countermeasures", *IEEE Communications Surveys & Tutorials*, 2018.

[108] PALADI, N., MICHALAS, A., HAI-VAN, D. "Towards Secure Cloud Orchestration for Multi-Cloud Deployments". In: *EuroSys-CrossCloud*, 2018.

[109] ZAWOAD, S., HASAN, R. "SECAP: Towards Securing Application Provenance in the Cloud". In: *2016 IEEE 9th International Conference on Cloud Computing*, pp. 900–903, June 2016. doi: 10.1109/CLOUD.2016.0132.

[110] BORDEL, B., ORÚE, A. B., ALCARRIA, R., et al. "An intra-slice security solution for emerging 5G networks based on pseudo-random number generators", *IEEE Access*, v. 6, pp. 16149–16164, 2018.

[111] KHETTAB, Y., BAGAA, M., DUTRA, D. L. C., et al. "Virtual security as a service for 5G verticals". In: *IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, 2018.

[112] VALTANEN, K., BACKMAN, J., YRJÖLÄ, S. "Creating value through blockchain powered resource configurations: Analysis of 5G network slice brokering case". In: *IEEE WCNCW'18*, pp. 185–190, Apr. 2018. doi: 10.1109/WCNCW.2018.8368983.

[113] BACKMAN, J., YRJÖLÄ, S., VALTANEN, K., et al. "Blockchain network slice broker in 5G: Slice leasing in factory of the future use case". In: *Internet of Things Business Models, Users, and Networks*, pp. 1–8, Nov. 2017. doi: 10.1109/CTTE.2017.8260929.

[114] HAWLITSCHEK, F., NOTHEISEN, B., TEUBNER, T. "The limits of trust-free systems: A literature review on blockchain technology and trust in the sharing economy", *Electronic commerce research and applications*, v. 29, pp. 50–63, 2018.

[115] KHATOUN, R., ZEADALLY, S. "Smart Cities: Concepts, Architectures, Research Opportunities", *Commun. ACM*, v. 59, n. 8, pp. 46–57, jul. 2016. ISSN: 0001-0782. doi: 10.1145/2858789. Disponível em: <http://doi.acm.org/10.1145/2858789>.

[116] TALARI, S., SHAFIE-KHAH, M., SIANO, P., et al. "A Review of Smart Cities Based on the Internet of Things Concept", *Energies*, v. 10, n. 4, 2017. ISSN: 1996-1073. doi: 10.3390/en10040421. Disponível em: <http://www.mdpi.com/1996-1073/10/4/421>.

[117] ISO/TC 307. "Blockchain and distributed ledger technologies". 2016. Disponível em: <https://www.iso.org/committee/6266604.html>. Acessado em 31 de agosto de 2019.

[118] IRTF. "Decentralized Internet Infrastructure Research Group". 2017. Disponível em: <https://trac.ietf.org/trac/irtf/wiki/blockchain-federation>. Acessado em 31 de agosto de 2019.

[119] W3C. "Blockchain Community Group". 2016. Disponível em: <https://www.w3.org/community/blockchain>. Acessado em 31 de agosto de 2019.

[120] ITU-T FG DLT. "Focus Group on Application of Distributed Ledger Technology". 2017. Disponível em: <https://itu.int/en/ITU-T/focusgroups/dlt/Pages/default.aspx>. Acessado em 31 de agosto de 2019.

[121] RADFORD, D. *Europe Blockchain Working Group*. Standard, The International Securities Association for Institutional Trade Communication, jul. 2016. Disponível em: <https://isitc-europe.com/

isitc-europe-blockchain-working-group>. Acessado em 31 de agosto de 2019.

[122] VISA. "About VisaNet". 2019. Disponível em: <https://usa.visa.com/about-visa/visanet.html>. Acessado em 31 de agosto de 2019.

[123] PAYPAL. "Paypal Holdings, Inc. (PYPL) SEC Filing 10-K Annual report for the fiscal year ending Monday, December 31, 2018". 2019. Disponível em: <https://filings.last10k.com/sec-filings/1633917/000163391719000043/pypl201810-k.htm.pdf>. Acessado em 31 de agosto de 2019.

[124] KIAYIAS, A., RUSSELL, A., DAVID, B., et al. "Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol". In: Katz, J., Shacham, H. (Eds.), *Advances in Cryptology – CRYPTO 2017*, pp. 357–388, Cham, 2017. Springer International Publishing. ISBN: 978-3-319-63688-7.

[125] CHAINALYSIS. *Crypto Crime Report: Decoding increasingly sophisticated hacks, darknet markets, and scams.* Relatório técnico, Chainalysis Inc., 2019.

[126] EYAL, I., SIRER, E. G. *Majority is not Enough: Bitcoin Mining is Vulnerable.* Relatório técnico, Department of Computer Science, Cornell University, Nov. 2013. Disponível em: <https://arxiv.org/pdf/1311.0243.pdf>. Acessado em 31 de agosto de 2019.

[127] BRACHA, G. "Asynchronous Byzantine agreement protocols", *Information and Computation*, v. 75, n. 2, pp. 130–143, 1987.

Apêndice A

Tolerãncia a Falhas e Árvores de Merkle

A.1 Limite de Falhas em um Acordo Bizantino

A tolerância máxima a falhas bizantinas em um sistema de computação distribuída, descoberta por Lamport no conhecido problema dos generais bizantinos [57], é de até um terço do total de participantes do sistema. No entanto, existem protocolos de consenso que optam por tolerar menos falhas do que o máximo permitido visando preservar a segurança do sistema [13, 60]. A seguir, demonstra-se como obter o limite máximo de falhas encontrado por Lamport e discute-se os efeitos de diminuir intencionalmente o limite de falhas toleradas.

A.1.1 Tolerância Máxima a Falhas Bizantinas

Seja S um sistema genérico de computação distribuída com um conjunto de participantes $P = H \cup F$, onde H é o conjunto de participantes honestos e F é o conjunto de participantes em falha bizantina. O conjunto possui cardinalidade $|P| = n = h + f$, onde n é a quantidade total de participantes, h é quantidade participantes honestos e f é a quantidade de participantes em falha. Para garantir consenso no sistema, é necessário garantir as propriedades de consistência (*safety*) e vivacidade (*liveness*).

Garantia de consistência. Suponha que os participantes honestos são particionados em dois grupos iguais de $\frac{h}{2}$ participantes que divergem sobre a decisão do consenso[1]. Neste caso, os participantes desonestos podem enviar até f votos ao primeiro grupo, concordando com sua decisão e, simultaneamente, enviar até f votos ao segundo grupo concordando com uma decisão diferente. Caso o protocolo

[1]Esta é a configuração mais suscetível a um ataque bizantino pois os nós honestos possuem a mesma quantidade de votos para decisões distintas [127].

de consenso admita uma decisão com $\frac{h}{2} + f$ votos ou menos, as mensagens maliciosas dos nós desonestos acarretam em uma decisão diferente para cada grupo, a propriedade de acordo no sistema é violada, e o sistema torna-se inconsistente. Portanto, define-se o limite inferior de votos necessários para garantir acordo sob f falhas bizantinas:

$$L_i > \frac{h}{2} + f \tag{A.1}$$

onde L_i é o limitante inferior de votos, h é o número de participantes honestos e f é o número de participantes em falha bizantina.

Garantia de vivacidade. Além de enviar mensagens divergentes para violar a propriedade de acordo, os participantes desonestos podem tentar interromper o consenso parando propositalmente de responder às mensagens que recebem e, consequentemente, omitindo seus votos. Portanto, o limite inferior de votos necessários também deve levar em conta o pior caso, em que todos os participantes desonestos atuam em conluio para não responder. Para garantir vivacidade nesse caso, o limite deve ser igual ou menor que o número de nós honestos. Do contrário a ausência de resposta de qualquer nó desonesto acarreta um quórum insuficiente para a tomada de decisão do algoritmo, violando a propriedade de terminação e interrompendo a vivacidade do sistema. Portanto, tem-se que:

$$L_i \leq h \tag{A.2}$$

onde L_i é o limitante inferior de votos e h é o número de participantes honestos.

Corretude do consenso. Substituindo A.1 em A.2, surge o limiar de tolerância máxima contra falhas bizantinas para garantir simultaneamente consistência e vivacidade em qualquer sistema distribuído:

$$f < \frac{h}{2} \tag{A.3}$$

onde h é o número de participantes honestos e f é o número de participantes em falha bizantina. Portanto, a corretude do consenso sob falhas bizantinas é garantida se e somente se o número de nós desonestos é menor do que metade dos nós honestos. Equivalentemente, substituindo $h = n - f$ em A.3, encontra-se o conhecido limiar em função do total de participantes na rede:

$$f < \frac{n}{3} \tag{A.4}$$

onde n é o número total de participantes na rede, incluindo participantes honestos e em falha bizantina, e f é o número de participantes em falha bizantina.

A.1.2 Limitantes Inferiores à Tolerância Máxima

A maior parte dos protocolos de consenso baseados em acordo bizantino utiliza a tolerância máxima de até um terço dos participantes em falha para garantir a corretude do consenso [13, 68, 77]. Outros protocolos baseados em prova que assumem redes síncronas atingem limiares de até 50% utilizando finalidade probabilística [1, 2]. No entanto, tolerar o máximo possível de falhas bizantinas nem sempre é a decisão óbvia e existem benefícios em adotar um valor menor que o limite inferior. Em especial, o protocolo Ripple, utilizado pela terceira maior criptomoeda em valor de mercado em 2019, realiza o consenso com tolerância de até 20% de participantes em falha [60]. O motivo é o compromisso entre a quantidade de falhas bizantinas necessárias para interromper o consenso e a quantidade necessária de falhas bizantinas para tomar controle do consenso e aprovar transações conflitantes que realizam um gasto duplo.

Seja S o mesmo sistema genérico de computação distribuída definido na Seção A.1.1. A consequência lógica do limitante inferior descoberto em A.4 é que, para garantir corretude do consenso, basta que mais de dois terços dos participantes da rede concordem sobre a mesma decisão (i.e. $h > \frac{2n}{3}$). Este limiar é chamado de "corretude forte"(*strong correctness*) [60]. No entanto, como não existe uma forma canônica de diferenciar os participantes "bons"dos participantes "maus", se a quantidade de participantes honestos for menor que dois terços dos participantes da rede (i.e. $h < \frac{n}{3}$), o sistema inverte a condição dos participantes e considera que os participantes honestos são os participantes em falha bizantina. Esté é o limiar de "corretude fraca"(*weak correctness*) [60]. Neste caso, os participantes bizantinos tomam o controle do consenso e podem deliberadamente realizar ataques de gasto duplo. Caso o número de participantes honestos seja inferior ao limiar de corretude forte, mas superior ao limiar de corretude fraca, o consenso não ocorre pois a propriedade de acordo seria violada.

A mesma formulação estende-se a qualquer sistema tolerante a falhas bizantinas. Sem perda de generalidade, pode-se definir as três faixas possíveis de h para garantir corretude do consenso em função do máximo de falhas f_{max} em qualquer sistema tolerante a falhas bizantinas:

$$i) \quad h > n - f_{max} \tag{A.5}$$

na qual o consenso ocorre e os participantes honestos controlam o sistema;

$$ii) \quad f_{max} < h < n - f_{max} \tag{A.6}$$

na qual o consenso é interrompido por falta de acordo e nenhum dos tipos de

participante controla o sistema; e

$$iii) \quad h < f_{max} \tag{A.7}$$

na qual o consenso ocorre e os participantes em falha bizantina controlam o sistema. Portanto, os limiares de corretude forte e de corretude fraca dependem da quantidade máxima de falhas e podem ser generalizados para, respectivamente, $n - f_{max}$ e f_{max}. A consequência importante deste resultado é o compromisso entre robustez e segurança causado por alterações em f_{max}: aumentar a tolerância a falhas (i.e. aumentar f_{max}) significa melhorar a robustez do consenso, porém facilita que participantes em falha bizantina controlem o sistema.

Na maior parte dos protocolos, considera-se a tolerância de um terço suficiente para garantir robustez e segurança [61–63, 79]. O protocolo Ripple utiliza a tolerância de $f_{max} = \frac{n}{5}$ para garantir maior segurança em detrimento da robustez do protocolo [60]. Os protocolos baseados em prova assumem $f_{max} = \frac{n}{2}$ e procuram prover mecanismos de segurança externos ao protocolo para prevenir que participantes maliciosos controlem o sistema, como a necessidade de poder computacional e mecanismos de punição [1, 26, 75].

A.2 Árvores de Merkle e Verificação Simples de Pagamento

As árvores de Merkle são a principal estrutura auxiliar utilizada em correntes de blocos para verificar a integridade e não-repúdio de uma transação de maneira eficiente [1]. Nomeadas em homenagem a Ralph Merkle, que patenteou o conceito em 1979, as árvores de Merkle são estruturas de dados em forma de árvore na qual cada nó não-folha, denominado de "nó-galho,"é o resultado de um *hash* de seus respectivos nós filhos. Cada nó-folha da árvore é o resultado de um *hash* de um conjunto de dados que, no caso da corrente de blocos, representam as transações da rede. A Figura A.1 ilustra a estrutura de uma corrente de blocos que utiliza uma árvore de Merkle binária para armazenar transações. A estrutura da árvore pode ser construída calculando o *hash* de cada transação T para criar o nível mais baixo da árvore, e, posteriormente, utilizando os *hashes* de cada nível para construir o próximo nível, até chegar à raiz da árvore. A complexidade de construção de uma árvore de Merkle de um bloco é $O(n.log_2(n)$ onde n é o número de transações contidas no bloco. A árvore binária é o tipo de árvore de Merkle mais utilizado em correntes de blocos, porém algumas implementações utilizam tipos diferentes de árvore [2].

O uso de árvores Merkle permite obter uma prova de Merkle (*Merkle proof*), que

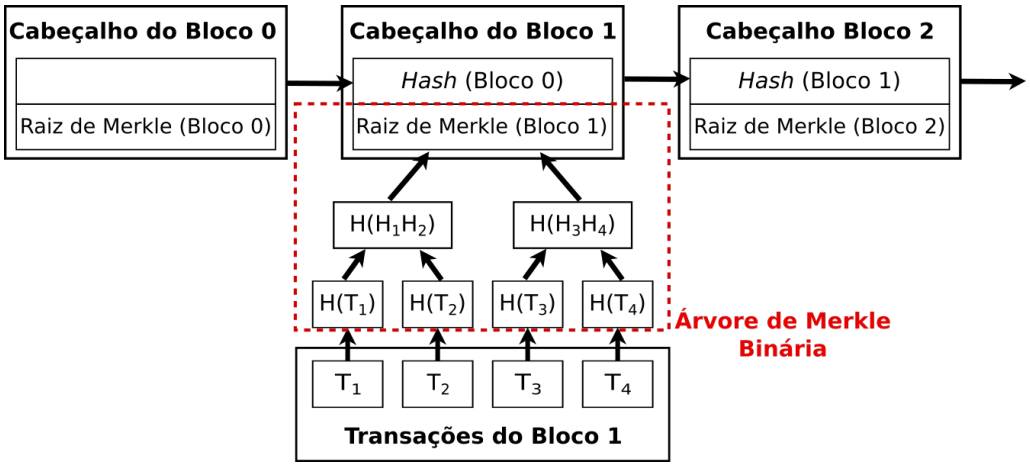


Figura A.1: Estrutura de uma árvore de Merkle binária utilizada em correntes de blocos. O uso de árvores de Merkle permite armazenar apenas a raiz da árvore sem prejuízo à verificação das transações.

verifica a presença e a corretude de uma transação em um bloco de maneira eficiente. Para verificar uma transação, basta fornecer os *hashes* necessários para reconstruir o caminho da árvore correspondente à transação verificada. A reconstrução de um caminho e, logo, a verificação de uma transação, tem complexidade $O(n)$. Assim, é possível verificar transações rapidamente mesmo em meio a milhares de transações, e não é necessário armazenar o bloco inteiro para verificar uma transação. Além disso, o uso de *hashes* diminui o tráfego na rede, pois não é necessário transferir a transação ou bloco completos. Isto permite criar "nós leves"e um mecanismo simplificado de verificação, pois, para verificar uma transação, basta armazenar o cabeçalho do bloco e solicitar a nós que possuem todas as transações os *hashes* do caminho até a transação. As provas de Merkle são a base do mecanismo de verificação simples de pagamento (*Simple Payment Verification* - SPV) proposto por Nakamoto e utilizado na maioria das implementações de corrente de blocos atuais [1, 2].