Uma Análise do Desempenho e Segurança de Protocolos de Consenso Baseados em Quórum para Corrente de Blocos

Gabriel Antonio F. Rebello, Gustavo F. Camilo, Lucas C. B. Guimarães, Lucas Airam C. de Souza, Otto Carlos M. B. Duarte

Universidade Federal do Rio de Janeiro - GTA/COPPE/UFRJ

Resumo. *Os protocolos de consenso dos sistemas baseados em corrente de blocos são a parte fundamental e mais complexa da corrente de blocos. O protocolo de consenso de prova de trabalho da moeda Bitcoin é de longe o mais usado, mas apresenta uma baixa vazão e um altíssimo custo energético. Este artigo analisa os principais protocolos de consenso de corrente de blocos baseados em quórum que são alternativas que apresentam uma maior vazão em número de transições por segundo e um baixíssimo custo energético. O artigo foca os protocolos tolerantes a falhas bizantinas por serem mais robustos a ataques de segurança. O artigo analisa e classifica os protocolos quanto às suas características principais e seus desempenhos, destacando os ataques específicos que afetam cada protocolo de consenso e apresentando as possíveis contramedidas.*

1. Introdução

A inovação disruptiva das correntes de blocos é prover confiança entre um grupo de participantes sem confiança mútua, de forma descentralizada e sem intermediários. A decisão centralizada é substituída por um acordo coletivo dos participantes obtido de forma distribuída. Para isso, o sistema deve ser capaz de incorporar novos blocos à corrente através de consenso[1] entre os participantes. A corrente de blocos é uma tecnologia com aplicação para garantir segurança em diversas áreas de pesquisa como Internet das coisas [Dedeoglu et al. 2020, Pinno et al. 2017], serviços médicos [Oliveira et al. 2019] e fatias de rede.Satoshi Nakamoto [Nakamoto 2008] revolucionou a área de transferência de ativos com a criptomoeda Bitcoin e sua proposta de consenso através da prova de trabalho (*Proof-of-Work* - PoW). No entanto, o consenso através da prova de trabalho é probabilístico porque pode gerar bifurcações na corrente de blocos, requer um consumo de energia excessivo, apresenta uma baixa vazão de transações e possui uma tendência de centralização das decisões em participantes com maior poder computacional.

O Bitcoin apresenta uma vazão de sete transações por segundo e uma hora de latência para se ter uma garantia correspondente a seis ciclos de geração de bloco. Estas características limitam a utilização do Bitcoin em aplicações que não podem esperar uma hora para serem concluídas ou por companhias de cartões de crédito, que processam até 56.000 transações por segundo [BitcoinWiki 2019]. Além disso, a prova de trabalho no Bitcoin consome anualmente mais de quatro vezes a energia gerada pelas usinas nucleares de Angra [Eletrobras 2017] e o gasto energético para processar uma única transação no Bitcoin é suficiente para abastecer uma residência média brasileira durante três

Este trabalho foi realizado com recursos do CNPq, CAPES, FAPERJ e FAPESP (18/23292-0, 2015/24514-9, 2015/24485-9 2014/50937-1).

[1]Consenso é um tipo de acordo obtido por consentimento entre todos os membros de um grupo.

meses [Digiconomist 2019, MME 2017]. Em resposta às limitações de desempenho do consenso baseado em prova de trabalho, diversos novos protocolos de consenso surgem como possíveis substitutos do protocolo do Bitcoin.

Este artigo classifica, analisa e compara os principais protocolos de consenso determinísticos baseados em quórum propostos como alternativa ao consenso por prova de trabalho. O artigo apresenta e analisa as características, a vazão em transações por segundo, a escalabilidade e a segurança de cada classe de protocolo de consenso determinístico. Diferente de outros artigos de análises de desempenho de protocolos, este artigo foca sobretudo na segurança dos protocolos analisados, especificando as principais vulnerabilidades de cada protocolo e as possíveis contramedidas. A análise é extensiva, cobrindo as principais os protocolos empregados nas principais criptomoedas existentes tais como: XRP, NEO, EOSIO e XLM.

O restante do artigo está organizado da seguinte forma. A Seção 2 discute os trabalhos relacionados. A Seção 3 apresenta os principais requisitos de um protocolo de consenso. A Seção 4 descreve e analisa a família de protocolos práticos de consenso tolerantes a falhas bizantinas. A Seção 5 apresenta e analisa os protocolos de acordo bizantino federado, citando o Ripple e o Stellar. A Seção 6 analisa e descreve as vulnerabilidades de protocolos da classe federados e a Seção 7 foca um protocolo híbrido; Por fim, a Seção 8 conclui o artigo e apresenta direções para trabalhos futuros.

2. Trabalhos Relacionados

A área de consenso para corrente de blocos atrai a atenção de diversos pesquisadores devido ao grande sucesso da moeda digital Bitcoin [Miers et al. 2019, Palma et al. 2019, Bessani et al. 2014]. No entanto, as vulnerabilidades e as contramedidas a ataques de segurança em protocolos de consenso baseados em quórum são pouco exploradas.

Vukolić compara as correntes de blocos baseadas em consenso determinístico e prova de trabalho [Vukolić 2015]. Angelis *et al.* avaliam o desempenho de protocolos de consenso de prova de autoridade e comparam com o protocolo PBFT [Angelis et al. 2018]. Entretanto, os trabalhos focam na escalabilidade dos protocolos de consenso, sem uma discussão aprofundada sobre as vulnerabilidades existentes em cada protocolo.

Xiao *et al.* [Xiao et al. 2020] e Joshi *et al.* [Joshi et al. 2018] apresentam diferentes protocolos de consenso determinísticos e probabilísticos para corrente de blocos. A discussão sobre ameaças e falhas de segurança existentes em protocolos de consenso baseados em quórum é breve e não apresenta contramedidas. Hasanova *et al.* analisam a segurança de diferentes correntes de blocos probabilísticas e determinísticas [Hasanova et al. 2019]. O trabalho, entretanto, foca as vulnerabilidades e contramedidas de protocolos baseados em prova e não provê uma análise profunda de protocolos determinísticos.

Wang *et al.* apresentam duas vulnerabilidades existentes no protocolo de consenso da criptomoeda NEO [Wang et al. 2020]. Christodoulou *et al.* analisam a segurança da criptomoeda Ripple na presença de diferentes números de adversários.

Este artigo apresenta de forma clara e concisa as principais características que

os protocolos de consenso baseados em quórum devem possuir. Além disso, o artigo descreve os principais protocolos de consenso desta classe usados em correntes de blocos, focando nas vulnerabilidades de segurança específicas e comparando cada protocolo.

3. O Consenso em Corrente de Blocos baseado em Quórum

Consenso é o processo pelo qual, a partir de um grupo de participantes independentes, todos os participantes corretos atingem a mesma decisão coletiva de aceitar ou recusar um novo bloco a ser inserido na corrente de blocos. Um dos participantes do consenso envia previamente a proposta de um novo bloco. Em consensos baseados em quórum, os participantes trocam mensagens com duas primitivas:

- $propor(P, b)$: propõe o novo bloco, b, ao conjunto de participantes do consenso P. Apenas um participante especial do consenso, o líder do consenso, pode enviar esta primitiva;
- $decidir(b)$: informa à rede que o participante validou e decidiu pelo bloco b.

O consenso ocorre toda vez que o líder propõe um novo bloco b e a maioria dos demais participantes validam e decidem pelo bloco proposto. Obter consenso não é trivial, pois falhas podem ocorrer tanto na entrega das mensagens, que podem sofrer atrasos ou serem perdidas, quanto na decisão dos participantes, que podem falhar devido a quedas de energia ou comportamento malicioso. Em um ambiente não-confiável, o consenso através da troca de mensagens ocorre se e somente se o protocolo garante as seguintes condições [Lamport et al. 1982].

- **Terminação (*termination*)**: todo participante correto[2] eventualmente toma uma decisão por um bloco b a ser inserido na corrente;
- **Acordo (*agreement*)**: o bloco b de todos os participantes corretos é idêntico;
- **Validade (*validity*)**: o bloco b decidido pelos participantes corretos é o bloco proposto pelo líder no início do consenso;
- **Integridade (*integrity*)**: o bloco b é proposto por um participante correto.

Os protocolos de consenso procuram garantir as quatro condições que, juntas, proveem as propriedades de consistência (*safety*) e vivacidade (*liveness*) ao protocolo. A garantia da condição de terminação provê vivacidade, pois as rodadas do consenso continuam acontecendo e o sistema sempre incorpora novos blocos. No entanto, isto não garante que os blocos são corretos. A condição de acordo provê uniformidade às decisões em todos os participantes, e as condições de validade e integridade representam a corretude da decisão, garantindo que o bloco foi proposto por um participante honesto. Juntas, a garantia de acordo, validade e integridade proveem a propriedade de consistência ao protocolo. A consistência não garante que o sistema sempre incorpora novos blocos, mas garante que os que são incorporados são sempre corretos. Portanto, um protocolo de consenso deve prover ambas as propriedades para garantir que o sistema funciona corretamente mesmo quando falhas ocorrem.

O protocolo de consenso pode tolerar falhas de dois tipos: falhas de parada (*crash faults*) ou falhas bizantinas (*byzantine faults*). Um participante em falha de parada não

[2]Um participante correto é um participante que não está em estado de falha de parada ou bizantina. Neste artigo os autores usam "honesto"como sinônimo de "correto"e "malicioso"como sinônimo de "em falha."

responde e não executa novas operações durante a execução do consenso. A falha bizantina é bem mais complexa, pois o participante em falha pode ser um agente malicioso que exibe um comportamento arbitrário, desviando do protocolo especificado e executando qualquer ação. O agente malicioso pode se comportar bem, respondendo corretamente, pode responder de forma errada ou pode não responder. Além disso, um participante em falha bizantina pode responder que aprova um bloco b para um participante e que aprova um bloco b' para outro participante. Assim, no modelo de falhas bizantinas não há informações precisas sobre o comportamento dos participantes ou se as informações do sistema estão corretas. O número máximo de participantes maliciosos[3] que um sistema baseado em quórum pode tolerar é de um terço do total de participantes na rede, incluindo participantes honestos e maliciosos. Este artigo foca os consensos tolerantes a falhas bizantinas, pois são mais robustos a comportamento malicioso.

Para obter o consenso por quórum, os sistemas de comunicação são fundamentais, pois os participantes do consenso devem trocar mensagens para chegar a um resultado. Fisher, Lynch e Paterson (FLP) publicaram em 1985 um dos trabalhos mais importantes envolvendo o problema do consenso [Fischer et al. 1985], conhecido como a impossibilidade de FLP. Eles provam que não é possível obter-se consenso em um sistema distribuído completamente assíncrono no qual pelo menos um participante pode falhar (qualquer tipo de falha). Logo, a maioria das propostas de consenso encontradas na literatura consideram sistemas de comunicação eventualmente síncronos que trabalham de forma assíncrona, não respeitando nenhum limite de tempo, a maior parte do tempo. Porém, durante períodos de estabilidade, o tempo para a entrega de mensagens é limitado. Este modelo é realístico, pois engloba o comportamento de redes de melhor esforço, como a Internet, que passam por tempos de estabilidade e tempos de perturbação, além de possuir uma solução determinística para o consenso [Fischer et al. 1985].

Ao contrário do consenso por prova de trabalho do Bitcoin, o consenso baseado em quórum requer o conhecimento da identidade e do número de participantes do consenso para computar o número de votos favoráveis a uma proposta de consenso. Portanto, todos os participantes precisam de permissão para participar do consenso. O consenso por quórum é um consenso permissionado. Além disso, os participantes permissionados têm que ser autenticados para evitar o ataque de Sybil. No Bitcoin, o ataque de Sybil não é grave porque o que é crucial é a capacidade de processamento. Porém, no consenso por quórum o que importa são os votos e, portanto, o ataque de Sybil é crucial. Assim, as mensagens trocadas pelos participantes devem ser autenticadas para se identificar e eliminar os participantes maliciosos A autenticação pode ocorrer através de criptografia assimétrica para gerar assinatura digitais das mensagens, que requer um sistema de infraestrutura de chaves públicas, ou através de criptografia simétrica e uso de código de autenticação de mensagens (*Message Authentication Codes* - MAC), que requer o compartilhamento prévio de segredos.

Os protocolos de consenso baseados em quórum podem ser divididos três grandes classes: o protocolo tolerante a falhas bizantinas prático e seus derivados, os protocolos tolerantes a falhas bizantinas delegados e os protocolos tolerantes a falhas bizantinas federados. As propostas apresentam diferentes características e desempenho, pois procu-

[3]Este problema é conhecido como problema dos generais bizantinos resolvido por Lamport [Lamport et al. 1982].

ram focar em uma certa propriedade como descentralização, com a rotação do líder, ou a vazão de transações, com divisão do consenso em federação. Por sua vez, os ataques de segurança vão procurar atrasar ou impedir a obtenção do consenso.

4. O Protocolo Prático Tolerante a Falha Bizantina

O Protocolo Prático Tolerante a Falha Bizantina (*Practical Byzantine Fault Tolerant* - PBFT) [Castro and Liskov 1999] é o primeiro protocolo de consenso que resolve o problemas dos generais bizantinos de forma prática[4]. O PBFT é prático pois otimiza a autenticação e comunicação entre os participantes, além de sobreviver corretamente em ambientes assíncronos como a Internet. Para incrementar o estado atual, o consenso centraliza todas as proposições de atualização no líder, denominado *primary*. Os demais participantes do consenso são réplicas, denominadas *backups*. Para garantir a tolerância a falha do líder, o protocolo possui uma mudança de perspectiva (*view-change protocol*) que garante a vivacidade. Os participantes são sequencialmente numerados. Assim, quando as réplicas detectam uma falha no líder e ativam o protocolo de mudança de perspectiva, o participante subsequente da fila circular assume a liderança e inicia-se uma nova rodada de consenso. Cada participante conhece todos os n participantes do consenso e suas respectivas assinaturas digitais. Assim, os participantes conhecem a quantidade mínima de votos para aprovar uma atualização, verificando facilmente a origem e a autenticidade das mensagens.

O consenso ocorre em um processo de 3 fases: pré-preparo, preparo e confirmação, como mostra a Figura 1. Ao receber a mensagem de pré-preparo (*pre-prepare*) do líder com a proposta do próximo estado, as réplicas verificam localmente o conteúdo e a sua validade. Caso a proposta seja válida, as réplicas assinam uma mensagem de preparo (*prepare*) e divulgam para todos os participantes. Os participantes aguardam o recebimento de $\lceil \frac{2n}{3} + 1 \rceil$ mensagens de preparo corretas para iniciar a próxima fase. A fase de preparo garante que os participantes honestos concordam sobre a proposta do líder, e o número de mensagens necessário deriva do número de falhas toleradas. Ao concluir a fase de preparo, os participantes enviam a confirmação (*commit*) contendo cada mensagem de preparo assinada por seus remetentes. A fase de confirmação garante que os participantes corretos entram em acordo sobre a transição e aplicam o próximo estado localmente quando recebem a mesma quantidade de votos da fase anterior.

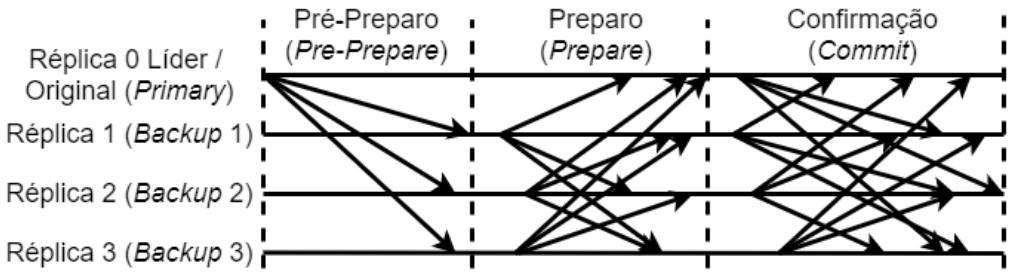

Figura 1. Diagrama de execução de uma rodada de consenso do protocolo PBFT com quatro participantes do consenso honestos.

[4]O PBFT foi apresentado como uma solução tolerante a falhas bizantinas através da replicação de arquivos do sistema NFS com apenas 3% de sobrecarga.

A principal limitação do protocolo PBFT é a escalabilidade em relação ao número de participantes do consenso devido à alta complexidade computacional na troca de mensagens [Amoussou-Guenou et al. 2019], pois os n participantes precisam trocar $O(n^2)$ mensagens para tolerar falhas maliciosas. Outra limitação do protocolo PBFT é a impossibilidade de adição de participantes em tempo de execução, pois, caso algum participante saia da rede permanentemente, o seu comportamento é considerado como uma falha. Este procedimento torna o PBFT inadequado para redes dinâmicas, nas quais participantes podem entrar e sair sob demanda. Um ataque importante ao PBFT é atrasar o tempo de finalização das rodadas na presença de participantes maliciosos [Amir et al. 2010]. Como a detecção de falha do líder se faz por temporização, o líder pode reduzir drasticamente a vazão do consenso atrasando a entrega de mensagens o máximo de tempo possível sem ser detectado.

Muitos autores propõem protocolos alternativos ao PBFT. O Rotativo (*Spinning*) é um protocolo de consenso que propõe a troca do líder a cada rodada [Veronese et al. 2009]. A proposta evita a centralização em um participante, que pode maliciosamente atrasar a entrega de mensagens para prejudicar a vivacidade do sistema. A rotação mitiga no mínimo o atraso malicioso na entrega de mensagens em $2f + 1$ rodadas.

O Protocolo Redundante Tolerante a Falha Bizantina (*Redundant Byzantine Fault Tolerance* - RBFT) é um protocolo de consenso robusto que identifica o atraso malicioso do líder na entrega de mensagens [Aublin et al. 2013]. Os participantes do consenso recebem as transações de clientes e estimam o tempo necessário para definir a proposta do próximo estado. Se uma rodada do consenso ultrapassa o limiar estimado, os participantes honestos solicitam a troca do líder. Entretanto, os clientes enviam mensagens para todos os participantes do consenso, enquanto nos outros protocolos os clientes enviam suas mensagens apenas para o líder.

5. O Protocolo Federado Tolerante a Falhas Bizantinas

A ideia principal do Protocolo Federado Tolerante a Falhas Bizantinas é fatiar o quórum e com isto aumentar significativamente a vazão de transações, diminuindo a quantidade de mensagens trocadas e o custo por transação. O acordo bizantino federado (*Federated Byzantine Agreement* - FBA) é uma forma de acordo bizantino no qual cada general bizantino é responsável por sua própria fatia de quórum (*quorum slice*). Enquanto os protocolos tradicionais do acordo bizantino restringem a admissão no consenso para prevenir ataques Sybil, o FBA concede a cada participante a liberdade de escolher em quem confiar. Assim, mesmo que um atacante crie diversas identidades, é necessário convencer uma grande quantidade de participantes legítimos a adicionar as identidades maliciosas em suas listas de participantes confiáveis. Os dois principais protocolos de consenso que implementam o FBA são o protocolo de consenso Stellar e o protocolo de consenso Ripple.

O protocolo de consenso Ripple (*Ripple Protocol Consensus Algorithm* - RPCA) [Schwartz et al. 2014] foi originalmente implementado em 2012 pela Ripple Labs. O objetivo do Ripple é oferecer segurança contra falhas bizantinas e uma alta taxa de transações por segundo, permitindo que a sua moeda XRP[5] não tivesse as mesmas

[5]Atualmente a XRP é a quarta maior criptomoeda em valor de mercado, com sua capitalização atingindo

limitações de vazão de transações que o Bitcoin e o Ethereum. A baixa latência necessária para a alta taxa de transações desejada ocorre devido à introdução de subredes de participantes considerados confiáveis dentro da rede principal, segundo o modelo FBA.

Para participar do RPCA como um validador, um participante deve executar um servidor capaz de aceitar e processar transações da criptomoeda XRP. Cada validador possui uma lista de nós únicos (*Unique Node List* - UNL), que é o conjunto de participantes considerados confiáveis pelo validador. Essa seleção ocorre de modo a reduzir o risco de ataques Sybil. No processo de obtenção do consenso, somente os votos dos servidores na UNL são levados em conta pelo validador para determinar quais transações devem ser adicionadas à corrente de blocos. O protocolo garante que todos os participantes alcançam corretamente o mesmo resultado no consenso e que é impossível ocorrer a validação de uma transação fraudulenta desde que a rede possua menos de 20% de participantes bizantinos.

O RPCA é dividido em duas etapas: consenso e validação. No consenso, cada validador inicialmente propõe um conjunto de transações que devem ser adicionadas à corrente. Cada validador verifica então as propostas dos participantes de sua UNL, e modifica a sua proposta de modo a adicionar transações que a maioria dos outros validadores na UNL propôs e remover transações que a maioria dos outros validadores na UNL não propôs. Esse processo se repete até que pelo menos 80% dos validadores na UNL concordem em um conjunto de transações válidas. Em seguida, inicia-se o processo de validação, no qual cada validador calcula individualmente o novo registro, utilizando as transações presentes no conjunto. O *hash* do novo registro é anunciando para todos os outros participantes da rede. Ocorre então a comparação dos resultados, onde o validador declara que sua versão do registro está validada somente se pelo menos 80% dos participantes em sua UNL receberam o mesmo *hash* dos outros participantes da rede. Caso isso não ocorra, e o validador possua um *hash* diferente do obtido pelos participantes em sua UNL, deve-se recalcular o *hash* ou baixar o registro validado pelos outros participantes.

O protocolo Ripple é vulnerável a ataques Sybil, dado que um atacante pode criar múltiplos validadores para tentar obter controle sobre as transações aprovadas pelo consenso. A contramedida para o ataque de Sybil são as UNL, de modo que somente validadores considerados confiáveis por outros nós da rede podem ter influência direta no consenso obtido. Para evitar o impacto de validadores maliciosos, o Ripple Labs mantém atualizada uma UNL padrão constituída por empresas e grupos confiáveis interessados no crescimento da moeda. Ainda, os validadores não recebem nenhuma forma de incentivo para atuarem no consenso, escolha feita para incentivar a presença de validadores interessados no progresso da moeda ao invés de validadores somente em busca desse incentivo.

Christodoulou *et al.* fazem uma análise do Ripple em ambientes adversos [Christodoulou et al. 2020]. Essa análise é feita alterando-se a porcentagem de participantes maliciosos na rede e a porcentagem de sobreposição de participantes nas UNLs. A primeira medida demonstra o impacto de participantes bizantinos no tempo necessário para se alcançar o consenso, enquanto a segunda demonstra o impacto da descentralização no protocolo nessa mesma medida. Pela análise feita, confirma-se que o tempo de convergência não é impactado para até 20% de participantes bizantinos; além disso, o impacto

valores superiores a 9 bilhões de dólares.

para valores maiores pode ser mitigado desde que sejam utilizadas UNLs com altas taxas de sobreposição. Outra medida considerada foi o indicador de saúde da rede (*Network Health Indicator* - NHI), que indica quantos registros candidatos foram validados durante o consenso. Os resultados demonstram que, mesmo com taxas de sobreposição de UNL de até 10%, o protocolo atinge bons valores de NHI desde que haja poucos participantes bizantinos.

O protocolo de consenso Stellar (*Stellar Consensus Protocol* - SCP), proposto em 2016 por David Mazières e utilizado pela criptomoeda Lumens (XLM), tem como principal vantagens aliar a baixa latência do consenso BFT à confiança flexível e o controle descentralizado do modelo FBA. Cada participante do consenso no SCP seleciona um ou mais conjuntos de outros participantes, chamados fatias de quórum (*quorum slices*), nos quais confia para trocar mensagens e decidir sobre um novo bloco. A cada rodada, o participante analisa as diferentes visões das diferentes fatias de quórum para tomar sua decisão. Um participante pode pertencer a diversas fatias de quórum e estabelecer limiares de decisão diferentes para cada fatia.

O protocolo Stellar separa a decisão de um bloco em duas etapas: a nomeação (*nomination*), em que os participantes propõem blocos, e a votação (*balloting*), em que participantes votam nos blocos a serem adicionados. O SCP atinge consenso em múltiplas fatias de quórum com um modelo de votação chamado votação federada (*federated voting*). Participantes da rede Stellar devem trocar mensagens declarando votos em um valor e, caso encontrem um quórum que vote ou aceite o mesmo valor, devem executar mais uma etapa de votação para confirmar o valor votado. Os participantes do SCP geram um conjunto de transações candidatas durante etapa de nomeação. Para validar suas transações, o participante deve alcançar um quórum completo que contém uma fatia para cada participante da rede. A Figura 2 ilustra uma rede que implementa o SCP, apresentando as fatias de quórum associadas a cada participante. Na figura, o participante N1 confia nos participantes N2 e N3, formando uma fatia de quórum associada ao participante N1. Para encontrar um quórum completo, o participante N1 deve começar na própria fatia e acrescentar as fatias de cada membro repetidamente até atingir o ponto em que os participantes a serem acrescentados já foram incluídos. Assim, o menor quórum completo que o participante N1 consegue encontrar contém todos os participantes, enquanto o menor quórum completo que o participante N5 consegue encontrar contém os participantes N4, N5 e N6. Ao encontrar um quórum que aceite o conjunto de transações, o participante passa a somente decidir sobre transações nomeadas por outros participantes. Como essa etapa gera múltiplos conjuntos de transações, os participantes da rede devem juntar as nomeações deterministicamente para convergir em um conjunto único de transações candidatas, por exemplo utilizando a união dos conjuntos de transação propostos.

Após a fase de nomeação, os participantes executam a fase de votação para definir o bloco. O sistema aplica a votação federada para decidir qual conjunto de transações gerado na etapa anterior forma o bloco finalizado. No entanto, o consenso pode travar caso os conjuntos de transações não tenham convergido na etapa anterior. O protocolo define cédulas com o conjunto de transações a ser votado e contadores para contornar esse problema. Assim, caso o consenso fique preso, os participantes incrementam o contador e passam para o próximo conjunto a ser votado até a rede chegar a um acordo.

O modelo de confiança flexível do SCP possui uma alta influência na segurança do

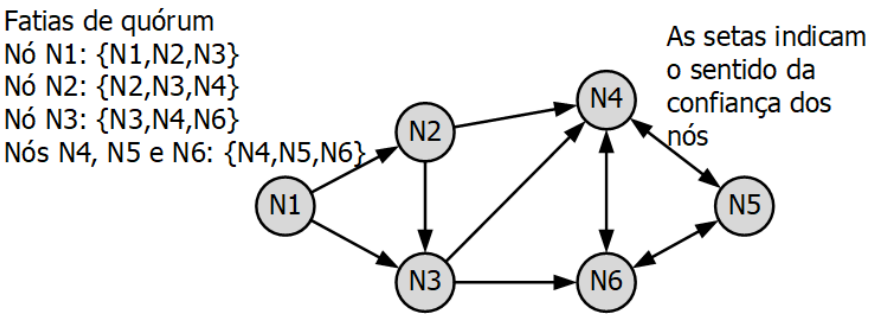

Figura 2. Fatias de quórum no Protocolo Stellar.

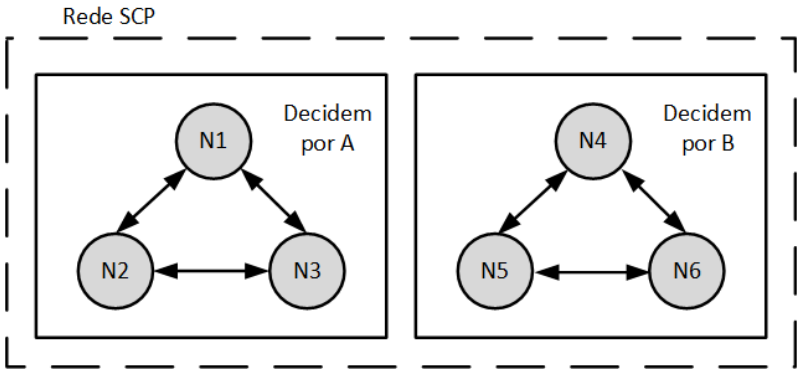

Figura 3. Cenário de uma rede que utiliza o protocolo Stellar e que não possui interseção entre quóruns. As setas indicam o sentido da confiança de cada participante. Como os quóruns não se intersectam, não há garantia de acordo na rede.

protocolo, uma vez que o funcionamento correto do protocolo depende de como as fatias de quórum se intersectam. O SCP perde a garantia de acordo quando não há interseção de quóruns completos, pois quóruns desconexos podem acordar em blocos contraditórios. A Figura 3 apresenta uma rede SCP em que não há garantia de acordo. No cenário da figura, o quórum completo formado pelos participantes N1, N2 e N3 decide por um bloco A como o bloco correto, enquanto o quórum completo formado pelos participantes N4, N5 e N6 decide por um bloco B como o bloco correto. Dessa forma, o sistema não chega em um valor acordado pelos participantes, comprometendo a propriedade do acordo.

Kim *et al.* analisam a rede Stellar para verificar a influência de cada participante no sistema e medir o grau de centralização [Kim et al. 2019]. Os autores utilizam uma versão modificada do algoritmo *PageRank* chamada *NodeRank*, que considera: (i) o número de fatias de quórum que contém o participante; (ii) se algum participante com alta influência possui o participante em sua fatia de quórum; (iii) o valor do limiar da fatia que contém o participante. O artigo verifica que os participantes mais importantes da rede Stellar são 3 validadores da Fundação Stellar. Ainda, os resultados mostram o risco de uma falha em cascata (*cascading failure*), em que alguns participantes falham devido à falha de alguns outros participantes. No caso da rede Stellar, os autores mostram que o consenso pode não atingir uma decisão caso 2 dos 3 participantes da Fundação Stellar falhem.

6. O Protocolo Delegado Tolerante a Falha Bizantina

O Protocolo delegado Tolerante a Falha Bizantina (*delegated Byzantine Fault Tolerant* - dBFT)[6] segue as mesmas fases dos protocolos BFT da Seção 4, porém centraliza o consenso em um número reduzido de participantes, proporcionando maior vazão e escalabilidade. O protocolo utiliza o conceito de reputação para escolher os nós participantes do consenso e, como o consenso Rotativo, o líder muda circularmente a cada rodada. No entanto, na implementação original do dBFT não existe a terceira fase, de confirmação, que existe no protocolo PBFT, o que torna o consenso vulnerável a falhas bizantinas. Os autores adicionaram ao dBFT a fase de confirmação, para garantir que os nós honestos concordam sobre a mudança de estado.

Segundo Wang *et al.* o protocolo dBFT da moeda NEO [Wang et al. 2020] exibe uma vulnerabilidade de segurança em sua implementação. O nó malicioso pode criar uma bifurcação (*fork*) determinística, conhecida como *spork*, explorando o protocolo de mudança de perspectiva. Um líder malicioso pode guardar mensagens e criar dois blocos válidos aprovados por pelos participantes honestos do consenso em perspectivas diferentes. Como os blocos são válidos, os participantes honestos podem aceitar qualquer um e criar dois estados diferentes na rede. Uma solução para esses problemas é descartar mensagens geradas antes da mudança de perspectiva, impossibilitando que participantes honestos aceitem dois estados válidos.

7. O Protocolo Híbrido Tolerante a Falhas Bizantinas e Prova de Posse Delegada

O Protocolo Híbrido Tolerante a Falhas Bizantinas e Prova de Posse Delegada (*Byzantine Fault Tolerance - Delegated Proof of Stake* - BFT-DPoS) procura associar a vantagem do alto desempenho dos protocolos da classe de prova de posse (PoS) com a segurança e o determinismo dos protocolos tolerantes a falhas bizantinas (BFT). Um exemplo desta classe é protocolo EOSIO [Larimer 2017] proposto e desenvolvido por Daniel Larimer[7] e utilizado na moeda digital EOS. O protocolo atinge o alto desempenho de mais de 3000 transações por segundo provido pelo DPoS, e as garantias de determinismo e segurança contra ataques bizantinos oferecidas pelo BFT. O protocolo possui duas fases: a eleição de delegados e a produção de blocos [EOSIO 2018].

Na primeira fase, os usuários que detêm tokens (*token holders*) podem utilizá-los para eleger produtores de blocos (*block producers*) através de um sistema de votação. Cada parte interessada do EOSIO pode votar em até 30 produtores de blocos por ação de votação. Os 21 produtores mais votados atuam como delegados que produzem blocos em nome das partes interessadas e participam do consenso BFT[8]. Qualquer participante da rede pode se candidatar para tornar-se um produtor de blocos desde que receba ao menos um voto de outro *token holder*. Entretanto, na prática o conjunto de produtores de

[6]Da HongFei e Erik Zhange propuseram o dBFT em 2014 com a criptomoeda Antshares, renomeada em 2017 para NEO [HongFei and Zhang 2018], que atualmente é uma das criptomoedas com maior capitalização de mercado.

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

[8]A quantidade total de 21 produtores é resultado de um trabalho anterior do autor no qual usuários podiam votar para definir a quantidade de produtores.

blocos varia pouco e consiste em entidades que investem diretamente no crescimento da criptomoeda, como os consórcios EOS New York e EOS Beijing.

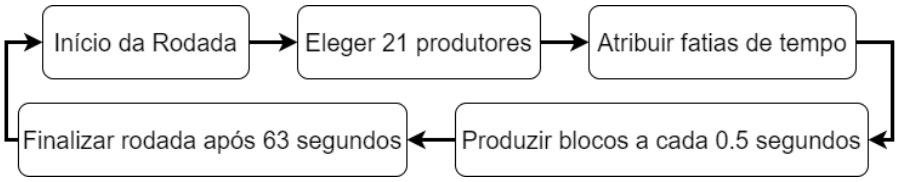

Figura 4. Uma rodada do protocolo de consenso EOSIO.

Após eleitos, os delegados eleitos iniciam uma rodada de produção de blocos através de acordo bizantino. Durante a rodada, cada delegado possui seis fatias de tempo (*time slots*) fixas de 0.5 segundo cada para produzir blocos. A ordem de produção de blocos é alfabética e, caso um produtor falhe em produzir um bloco dentro da sua fatia de tempo, o produtor seguinte ignora o bloco e continua o processo. Assim, bifurcações podem ocorrer e cada rodada adiciona até 126 blocos à corrente de blocos, durando um total de 63 segundos. A Figura 4 ilustra as principais etapas de uma rodada. Por fim, o protocolo resolve as bifurcações geradas através de tolerância a falhas bizantinas (BFT) antes de adicioná-los à corrente de blocos. Quando 15 dos 21 produtores, i.e. mais de 2/3 dos delegados, confirmam um bloco através de mensagens assinadas, o protocolo o adiciona à corrente de blocos.

A principal fragilidade do protocolo EOSIO é a centralização em apenas 21 delegados que são eleitos por votação. Este modelo possui vulnerabilidades claras: (i) Como cada voto tem peso proporcional à posse do participante, um conluio entre poucos participantes com grandes posses é suficiente para eleger delegados maliciosos. Evidências deste tipo de ataque já existem na prática, através de padrões de gangues de votação identificados na corrente de blocos do EOSIO [Zhao et al. 2020]; (ii) A rede é suscetível a ataques de gasto duplo com apenas $\lfloor \frac{n}{3} - 1 \rfloor = 6$ delegados maliciosos. Como cada participante pode votar em até 30 delegados simultaneamente, a eleição de 6 delegados maliciosos pode ser facilmente alcançada. A comunidade do EOSIO mitiga este problema elegendo frequentemente os mesmos participantes baseado na reputação que possuem. Esta prática, no entanto, centraliza o processo de produção de blocos em um conjunto fixo de entidades que podem ser alvos de ataques. Lee *et al.* demostram que um atacante é capaz de modificar o comportamento de um delegado confiável através de ataques de negação de serviço e sequestro de memórias [Lee et al. 2019]. Os desenvolvedores da EOSIO reconheceram as vulnerabilidades apresentadas e planejam corrigi-las no futuro [Lee et al. 2019, Block One 2018]. (iii) Após eleitos, os delegados possuem o mesmo poder independente da quantidade de votos recebidos. Esta característica minimiza o custo de ataques de conluio, pois os atacantes precisam apostar apenas um pequeno conjunto de posses suficiente para eleger os delegados menos votados.

Apesar de o consenso ocorrer apenas entre delegados eleitos, o EOSIO também sofre de problemas similares às plataformas públicas como o Ethereum, pois qualquer participante pode inicializar um contrato inteligente ou emitir uma transação de forma anônima. Um estudo recente indica que mais de 30% dos usuários no EOSIO correspondem a redes de robôs (*botnets*) e mais de 300 ataques já foram detectados em aplicações descentralizadas [Huang et al. 2020]. No entanto, diferente do Ethereum, o EOSIO

Tabela 1. Análise dos protocolos de consenso em corrente de blocos

Protocolo	Tipo de Consenso	Vazão máxima	Nº de validadores	Vulnerabilidades	Contramedidas
PBFT	BFT tradicional	$\approx$ 300 tx/s	Dezenas	Atrasos na finalização das rodadas	Trocar o líder e monitorar vazão
NEO	BFT delegado	$\approx$ 10000 tx/s	7	Exploração da mudança de perspectiva [Wang et al. 2020] e bifurcação determinística.	Descartar mensagens após mudança de perspectiva e introduzir fase de confirmação.
EOSIO	Híbrido BFT-DPoS	$\approx$ 3000 tx/s	21	Exploração de contratos inteligentes [He et al. 2020, Zhao et al. 2020], manipulação de delegados [Lee et al. 2019] e subversão do consenso através de gangues de votação [Huang et al. 2020].	Restringir as funcionalidades de contratos inteligentes, aumentar a quantidade de delegados no consenso.
Ripple	FBA	$\approx$ 1500 tx/s	Dezenas	Ataques Sybil	Uso automático de UNL padrão atualizada pelo Ripple Labs
Stellar	FBA	$\approx$ 1000 tx/s	Dezenas	Exploração das interseções de quórum [Kim et al. 2019]	Inserção de validadores confiáveis para prover a base de um quórum

permite modificar contratos inteligentes já publicados na corrente de blocos, o que abre espaço para ataques de modificação de código em contratos que não possuem código aberto [He et al. 2020]. Os contratos inteligentes do EOSIO também possuem uma funcionalidade que permite ao contrato gerenciar automaticamente os *tokens* de um usuário, o que na prática causa abusos e perdas financeiras milionárias [Lee et al. 2019].

8. Conclusão

Os protocolos tolerantes a falhas bizantinas são os que melhor proveem segurança em correntes de blocos [Schwartz et al. 2014, Castro and Liskov 1999] com alta vazão. A Tabela 1 apresenta uma comparação entre os protocolos analisados neste artigo. O protocolo de consenso prático tolerante a falhas bizantinas (PBFT) apresenta robustez quanto a segurança (*safety*), entretanto, na presença de participantes maliciosos líderes de rodadas o seu desempenho pode ser extremamente degradado, prejudicando a vivacidade (*liveness*). Outra dificuldade da aplicação do PBFT é a sua baixa escalabilidade quanto ao número de participantes e não suporta ambientes dinâmicos, no qual o conjunto de participantes do consenso se altera com frequência. A variação do PBFT, denominada Spinning, modifica o líder a cada rodada, mitigando os problemas de vivacidade do PBFT. Entretanto, sofre dos mesmos problemas quanto à escalabilidade e ambientes dinâmicos.

Os protocolos Ripple e Stellar se baseiam no acordo bizantino federado para aumentar a vazão, diminuir os custos das transações e para conseguir lidar com conjuntos dinâmicos de participantes. A ideia-chave é fatiar o quórum e com isto flexibilizar a confiança. O protocolo é resistente aos ataques Sybil, pois um atacante deve ganhar a confiança de diversos participantes para ser adicionado às fatias de quórum. No entanto, a segurança e desempenho do FBA dependem diretamente da quantidade de participantes, do número de fatias de quórum e do tamanho da interseção entre as fatias. Tanto no protocolo Ripple quanto no Stellar, o sistema mitiga este problema oferecendo a possibilidade de inserir uma lista padrão de participantes confiáveis em todos os novos usuários da rede. A lista pode ser modificada posteriormente pelo usuário.

O protocolo dBFT da moeda NEO e o protocolo híbrido (BFT-dPoS) da moeda EOS possuem alto desempenho em relação ao PBFT e são adequados para ambientes dinâmicos ou com grande número de participantes. Entretanto, o modo de seleção dos participantes do consenso é vital para a segurança, uma vez que nem todos os nós são

responsáveis por gerar novos blocos. As falhas de segurança encontradas na implementação do dBFT e EOSIO são reconhecidas pela fundação NEO e pela empresa Block.One, respectivamente, e serão corrigidas nas próximas versões.

Em trabalhos futuros, pretende-se analisar protocolos híbridos e protocolos baseados em grafos acíclicos direcionados (DAG) adaptados para Internet das Coisas.

Referências

Amir, Y., Coan, B., Kirsch, J., and Lane, J. (2010). Prime: Byzantine replication under attack. *IEEE TDSC*, 8(4):564–577.

Amoussou-Guenou, Y., Pozzo, A. D., Potop-Butucaru, M., and Tucci-Piergiovanni, S. (2019). Dissecting Tendermint. In *ICNS*, pages 166–182. Springer.

Angelis, S. D., Aniello, L., Baldoni, R., Lombardi, F., Margheri, A., and Sassone, V. (2018). PBFT vs Proof-of-Authority: applying the CAP theorem to permissioned blockchain. In *Italian Conference on Cyber Security (06/02/18)*.

Aublin, P.-L., Mokhtar, S. B., and Quéma, V. (2013). RBFT: Redundant Byzantine Fault Tolerance. In *IEEE ICDCS'13*, pages 297–306. IEEE.

Bessani, A. N., Sousa, J., and Alchieri, E. A. P. (2014). State machine replication for the masses with bft-smart. In *IEEE/IFIP DSN*, pages 355–362.

BitcoinWiki (2019). Bitcoin Scalability. Acessado em 31/07/2020.

Block One (2018). EOSIO RAM resource exploit patch. Disponível em https://block.one/news/eosio-ram-resource-exploit-patch/. Acessado em 31/07/2020.

Castro, M. and Liskov, B. (1999). Practical Byzantine Fault Tolerance. In *OSDI*, volume 99, pages 173–186.

Christodoulou, K., Iosif, E., Inglezakis, A., and Themistocleous, M. (2020). Consensus crash testing: Exploring Ripple's decentralization degree in adversarial environments. *Future Internet*, 12(3):53.

Dedeoglu, V., Jurdak, R., Dorri, A., Lunardi, R. C., Michelin, R. A., Zorzo, A. F., and Kanhere, S. S. (2020). Blockchain technologies for IoT. In *Advanced Applications of Blockchain Technology*, pages 55–89. Springer.

Digiconomist (2019). Bitcoin Energy Consumption Index. Acessado em 31/07/2020.

Eletrobras (2017). Relatórios de sustentabilidade socioambiental. Technical report, Eletrobras S.A. Acessado em 31/07/2020.

EOSIO (2018). EOSIO development documentation. Disponível em https://developers.eos.io/welcome/latest/protocol/consensus_protocol. Acessado em 31/07/2020.

Fischer, M. J., Lynch, N. A., and Paterson, M. S. (1985). Impossibility of Distributed Consensus with One Faulty Process. *Journal of the ACM*, 32(2):374–382.

Hasanova, H., Baek, U., Shin, M., Cho, K., and Kim, M. (2019). A survey on blockchain cybersecurity vulnerabilities and possible countermeasures. *IJNM*, 29(2):e2060.

He, N., Zhang, R., Wu, L., Wang, H., Luo, X., Guo, Y., Yu, T., and Jiang, X. (2020). Security Analysis of EOSIO Smart Contracts. *arXiv e-prints*, page arXiv:2003.06568.

HongFei, D. and Zhang, E. (2018). NEO white paper. Disponível em https://docs.neo.org/docs/en-us/basic/whitepaper.html. Acessado em 31/07/2020.

Huang, Y., Wang, H., Wu, L., Tyson, G., Luo, X., Zhang, R., Liu, X., Huang, G., and Jiang, X. (2020). Understanding (mis) behavior on the EOSIO blockchain. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 4(2):1–28.

Joshi, A. P., Han, M., and Wang, Y. (2018). A survey on security and privacy issues of blockchain technology. *Mathematical foundations of computing*, 1(2):121.

Kim, M., Kwon, Y., and Kim, Y. (2019). Is Stellar as secure as you think? In *IEEE EuroS PW'19*, pages 377–385.

Lamport, L., Shostak, R., and Pease, M. (1982). The Byzantine Generals Problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401.

Larimer, D. (2017). EOS.IO White Paper. Disponível em https://developers.eos.io/-welcome/latest/protocol/consensus_protocol. Acessado em 31/07/2020.

Lee, S., Kim, D., Kim, D., Son, S., and Kim, Y. (2019). Who spent my EOS? On the (in)security of resource management of EOS.IO. In *USENIX WOOT'19*.

Miers, C., Koslovski, G., Pillon, M., Simplício, M., Carvalho, T., Rodrigues, B., and Battisti, J. (2019). *Análise de Mecanismos para Consenso Distribuído Aplicados a Blockchain*, chapter 3. SBC.

MME (2017). Anuário estatístico de energia elétrica 2017. Technical report, Ministério de Minas e Energia do Brasil. Acessado em 31/07/2020.

Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system.

Oliveira, M. T., Reis, L. H., Carrano, R. C., Seixas, F. L., Saade, D. C., Albuquerque, C. V., Fernandes, N. C., Olabarriaga, S. D., Medeiros, D. S., and Mattos, D. M. (2019). Towards a blockchain-based secure electronic medical record for healthcare applications. In *IEEE ICC*, pages 1–6.

Palma, L. M., Vigil, M. A., Pereira, F. L., and Martina, J. E. (2019). Blockchain and smart contracts for higher education registry in Brazil. *IJNM*, 29(3):e2061.

Pinno, O. J. A., Gregio, A. R. A., and De Bona, L. C. (2017). ControlChain: Blockchain as a central enabler for access control authorizations in the IoT. In *IEEE GLOBECOM*, pages 1–6.

Schwartz, D., Youngs, N., Britto, A., et al. (2014). The Ripple protocol consensus algorithm. *Ripple Labs Inc White Paper*, 5(8).

Veronese, G. S., Correia, M., Bessani, A. N., and Lung, L. C. (2009). Spin one's wheels? Byzantine Fault Tolerance with a spinning primary. In *2009 28th IEEE International Symposium on Reliable Distributed Systems*, pages 135–144. IEEE.

Vukolić, M. (2015). The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication. In *International workshop on open problems in network security*, pages 112–125. Springer.

Wang, Q., Yu, J., Peng, Z., Bui, V. C., Chen, S., Ding, Y., and Xiang, Y. (2020). Security analysis on dBFT protocol of NEO. In *International Conference on Financial Cryptography and Data Security*, pages 20–31. Springer.

Xiao, Y., Zhang, N., Lou, W., and Hou, Y. T. (2020). A survey of distributed consensus protocols for blockchain networks. *IEEE Communications Surveys & Tutorials*, 22(2):1432–1465.

Zhao, Y., Liu, J., Han, Q., Zheng, W., and Wu, J. (2020). Exploring EOSIO via Graph Characterization. *arXiv e-prints*, page arXiv:2004.10017.