Programa de Pós-Graduação em

Computação Aplicada

Mestrado Acadêmico

Fabricio Reis Furtado

L7SP: Serviços para otimizar o gerenciamento e o desempenho de Blockchain Privados

São Leopoldo, 2019

Fabricio Reis Furtado

L7SP:
Serviços para otimizar o gerenciamento e o desempenho de Blockchain Privados

Dissertação apresentada como requisito parcial para a obtenção do título de Mestre em Computação Aplicada pela Universidade do Vale do Rio dos Sinos — UNISINOS

Orientador:
Prof. PhD. Rodrigo da Rosa Righi

São Leopoldo
2019

F992l Furtado, Fabrício Reis.
L7SP: serviços para otimizar o gerenciamento e o desempenho de Blockchain Privados / Fabrício Reis Furtado. – 2019.
114 f. : il. color. ; 30 cm.

Dissertação (mestrado) – Universidade do Vale do Rio dos Sinos, Programa de Pós-Graduação em Computação Aplicada, São Leopoldo, 2019.
"Orientador: Prof. PhD. Rodrigo da Rosa Righi."

1. Blockchains (Base de dados). 2. Desempenho. 3. Bitcoin. 4. Transferência eletrônica de fundos. I. Título.

CDU 004.65

Dados Internacionais de Catalogação na Publicação (CIP)
(Bibliotecária: Bruna Sant'Anna – CRB 10/2360)

Fabricio Reis Furtado

L7SP:
Serviços para otimizar o gerenciamento e o desempenho de Blockchain Privados

Dissertação apresentada à Universidade do Vale do Rio dos Sinos – Unisinos, como requisito parcial para obtenção do título de Mestre em Computação Aplicada.

Aprovado em 02 de abril de 2019

BANCA EXAMINADORA

Prof. Dr. Rodrigo da Rosa Righi – UNISINOS

Prof. Dr. Cristiano André da Costa – UNISINOS

Prof. Dr. Avelino Francisco Zorzo – PUCRS

Prof. Dr. Rodrigo da Rosa Righi (Orientador)

Visto e permitida a impressão
São Leopoldo,

Prof. Dr. Rodrigo da Rosa Righi
Coordenador PPG em Computação Aplicada

AGRADECIMENTOS

Agradeço primeiramente a Deus pela minha vida e por me abençoar com esta oportunidade de aprendizado e crescimento. Aos meus pais Arione e Glades pela minha educação e incentivo a estudar desde sempre. Ao meu orientador e professor Rodrigo Righi, pelo constante incentivo e suporte nas atividades durante todo o curso. Ao Sicredi, representado pelo seus colaboradores e gestores, que investiu nesta jornada junto comigo e possibilitou que eu dedicasse o tempo necessário para atingir os objetivos. À minha esposa Marcielle pela compreensão e apoio em todos os momentos. E por último e mais importante, a minha filha Lara, por me dar forças de seguir em frente e ser fonte de inspiração e coragem nos momentos de dificuldade.

″O que sabemos não é muito. O que não sabemos é imenso”.
Pierre Simon Laplace

RESUMO

Desde o lançamento da moeda digital Bitcoin em 2009, o interesse pela tecnologia blockchain tem crescido ao longo dos anos e tem atraído muitos pesquisadores. A tecnologia surgiu inicialmente dentro da plataforma Bitcoin para suportar as operações com a moeda digital de forma descentralizada e segura, sem a necessidade de intermediários como bancos e organizações governamentais. Devido às suas características de descentralização e imutabilidade, logo foi percebido o seu potencial para outros tipos de aplicações, além dos pagamentos com moedas digitais. Apesar de mudar alguns paradigmas tecnológicos existentes e prover vários benefícios interessantes, a tecnologia blockchain também apresenta algumas limitações e desafios técnicos comparada aos sistemas centralizados, principalmente no que diz respeito a desempenho e escalabilidade. Por ser uma rede descentralizada, para validar as transações a blockchain necessita obter consenso entre os diferentes nós participantes. Esta característica é o principal ofensor, pois, os protocolos implementados para prover este processamento distribuído de forma segura afetam diretamente o desempenho do sistema. As limitações de escalabilidade e desempenho da blockchain são críticas para as plataformas públicas como o Bitcoin, mas também afetam as redes privadas, com o diferencial que estas possuem maior autonomia na configuração da solução. A blockchain privada oportuniza que empresas ou grupo de empresas utilizem a tecnologia para a validação de transações internas para diferentes tipos de aplicações. Existem diversas propostas para melhorar o desempenho do blockchain, porém estas propostas trazem uma série de mudanças de difícil implementação e que não são compatíveis com as aplicações hoje existentes. As limitações atuais impedem que a blockchain seja usado em larga escala para outros tipos de aplicações ou mesmo substitua as formas de pagamento eletrônico conhecidas atualmente, fazendo com que estudos sejam necessários para endereçar estas lacunas e evoluir a tecnologia. Considerando este cenário da necessidade em melhorar os aspectos abordados da blockchain e a oportunidade de explorar a tecnologia para outras aplicações no meio corporativo, este trabalho propõe o modelo L7SP que será acoplado à blockchain privada para prover ganho de desempenho e escalabilidade, e melhorar o gerenciamento do sistema. L7SP é uma camada intermediária que provê o empilhamento de serviços buscando maior rapidez nas validações das transações e geração dos blocos, monitoramento de desempenho e um gerenciamento de forma centralizada. O modelo proposto provê alta disponibilidade e desempenho, gerando economia de recursos, e mantendo as características de segurança e descentralização da blockchain. A viabilidade e eficácia de L7SP é demonstrada por meio de um protótipo que atua entre os nós validadores, sendo transparente para os clientes que interagem com o sistema e sem interferir na forma como a blockchain é implementada, sendo assim também compatível com as aplicações já existentes. L7SP roda na nuvem da AWS e se comunica com os demais componentes da solução utilizando os protocolos disponibilizados pelo *middleware* Multichain, cujos nós também rodam na nuvem. Os resultados da introdução de L7SP e seus serviços apresentaram ganhos de desempenho de 4% até 422%, quando comparado ao desempenho do Multichain em sua configuração padrão. Testes com diferentes parametrizações e cargas de trabalho demonstraram que o balanceamento de carga com estratégia *Round Roubin* e a utilização de compressão de dados foi a melhor combinação para se obter o ganho máximo de desempenho. A eficiente combinação desses fatores, com a introdução de novas métricas e um monitoramento do sistema mais eficiente configuram a contribuição científica deste estudo.

Palavras-chave: Blockchain. Escalabilidade. Desempenho.

ABSTRACT

Since the launch of the Bitcoin digital currency in 2009, the interest in blockchain technology has grown over the years and has attracted many researchers. The technology initially emerged within the Bitcoin platform to support operations with the digital currency in a decentralized and secure manner, without the need for intermediaries such as banks and government organizations. Due to its characteristics of decentralization and immutability, its potential was soon perceived for other application types, in addition to payments with criptocurrencies. Although changing some existing technology paradigms and providing several interesting benefits, blockchain technology also has some limitations and technical challenges compared to centralized systems, especially in terms of performance and scalability. Because it is a decentralized network, to validate transactions, blockchain needs to reach consensus among the different participating nodes. This feature is the main offender, because the protocols implemented to provide this securely distributed processing directly affect system performance. The limitations of scalability and performance of the blockchain are critical for public platforms such as Bitcoin, but also affect private networks, with the differential that they have greater autonomy in the solution setting. The private blockchain allows companies or groups of them to use the technology to validate internal transactions for different types of business applications. There are several proposals to improve the performance of the blockchain, however these proposals bring a series of changes difficult to implement and are not compatible with current applications. The current limitations prevent blockchain from being used on a large scale for other types of applications or even replace the currently known electronic payment forms, making studies necessary to address these gaps and evolve the technology. Considering this scenario of the need to improve the aspects covered by the blockchain and the opportunity to explore the technology for other applications in the corporate environment, this paper proposes the L7SP model that will be coupled to the private blockchain to provide performance and scalability gain and improve management of the system. L7SP is an intermediate layer that provides the stacking of services seeking faster transaction validation and block generation, performance monitoring and centralized management. The proposed model provides high availability and performance, generating resource savings, and maintaining the security and decentralization characteristics of the blockchain. The viability and effectiveness of L7SP is demonstrated through a prototype that acts among the validating nodes, being transparent to the clients that interact with the system and without interfering in the way the blockchain is implemented, being thus also compatible with the existing applications . L7SP runs on the AWS cloud and communicates with the other components of the solution using the protocols provided by the Multichain middleware, whose nodes also run in the cloud. The results of the introduction of L7SP and its services showed performance gains of 4 % to 422 % when compared to the performance of Multichain in its standard configuration. Tests with different parameterizations and workloads showed that load balancing with Round Roubin strategy and the use of data compression was the best combination to obtain the maximum performance gain. The efficient combination of these factors, with the introduction of new metrics and a more efficient system monitoring, configure the scientific contribution of this study.

Keywords: Blockchain. Scalability. Performance.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

API	Application Program Interface
ASIC	Application Specific Integrated Circuits
AWS	Amazon Web Services
BFT	Byzantine Fault Tolerant
BKPI	Blockchain Key Performance Indicator
CPU	Central Processing Unit
DHT	Distributed Hash Table
DLT	Distributed Ledger Technology
DPoS	Delegated Proof of Stake
ECDSA	Elliptic Curve Digital Signature Algorithm
FBA	Federated Byzantine Agreement
FBFT	Federated Byzantine Fault Tolerant
FPGA	Field Programmable Gate Array
GPU	Graphics Processing Unit
IBLT	Invertible Bloom Lookup Tables
IP	Internet Protocol
ISO	International Organization for Standardization
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
L7SP	Layer Seven Service Provider
LPOS	Leased Proof of Stake
P2P	Peer to Peer
PPM	Parts per Million
PBFT	Practical Byzantine Fault Tolerant
PoA	Proof of Activity
PoB	Proof of Burn
PoC	Proof of Capacity
PoET	Proof of Elapsed Time
PoI	Proof of Importance
PoR	Proof of Retrievability
PoS	Proof of Stake
PoSV	Proof of Stake Velocity
PoW	Proof of Work

RAM	Random Access Memory
RPC	Remote Procedure Call
RPCA	Ripple Protocol Consensus Algorithm
SCP	Stellar Consensus Protocol
SO	Sistema Operacional
TCP	Transmission Control Protocol
TPS	Transactions per Second
UNL	Unique Node List
UTXO	Unspent Transaction Output
VM	Virtual Machine

SUMÁRIO

1 INTRODUÇÃO

A tecnologia blockchain é constituída por um banco de dados distribuído que cresce continuamente. Ele contém uma lista de registros transacionais irrevogáveis, que são assinados criptograficamente e compartilhados por todos os participantes em uma rede ponto a ponto (P2P). Em uma visão de negócios, a blockchain é uma rede de troca de transações, valores ou ativos entre pares, sem a ajuda de intermediários (MOUGAYAR, 2016). Blockchain também é definida como um livro razão público, que registra todas as operações em uma sequência de blocos que são adicionados linearmente e em ordem cronológica (SWAN, 2015). A blockchain permite que indivíduos que não se conhecem ou não confiam uns nos outros, mantenham um consenso sobre o *status* e as alterações nesse banco de dados compartilhado.

A notoriedade da blockchain deve-se especialmente à popularização da moeda digital Bitcoin, que é estabelecida sem a necessidade de intermediários como bancos, operadoras de cartão ou instituições governamentais. Garantir a segurança nas transações sem intermediários e resolver o problema do gasto duplo de forma descentralizada foi a principal inovação do modelo proposto na criação do Bitcoin (NAKAMOTO, 2008).

Existem várias características chave que diferenciam a blockchain dos demais sistemas centralizados conhecidos atualmente, mas com certeza a descentralização é a principal delas. Em sistemas centralizados cada transação deve ser validada através da instituição central confiável, inevitavelmente resultando em gargalos de custo e desempenho nos servidores centrais. Na blockchain os algoritmos de consenso são usados para manter a consistência dos dados na rede distribuída, dispensando a necessidade do uso de intermediários. A blockchain também provê persistência, uma vez que as transações podem ser validadas rapidamente e é quase impossível excluir ou reverter transações depois que elas são adicionadas ao *ledger*. Blocos que contêm transações inválidas podem ser descobertos imediatamente. Outra importante característica é o anonimato. Dependendo do tipo de blockchain e sua configuração, cada usuário pode interagir com o sistema com um endereço gerado, que não revela a identidade real do usuário. A blockchain também provê rastreabilidade, o histórico de transações está registrado no *ledger* distribuído e seu modelo de dados permite que as transações possam ser facilmente verificadas e auditadas (ZHENG et al., 2017).

A blockchain inicialmente teve sua aplicação voltada apenas para criptomoedas, mas seu potencial para ser utilizado em outras aplicações logo se tornou aparente, devido a sua principal característica de prover um registro público e descentralizado de transações. A segunda aplicação a ser explorada na blockchain foi chamada de *smart contracts*, onde foi utilizada como validadora dos termos dos contratos digitais publicados na rede (SWAN, 2015).

1.1 Motivação

Sendo uma tecnologia recente que não possui um padrão e regulação, a implementação pode ser modificada de acordo com a necessidade da aplicação, surgindo assim diferentes plataformas e tipos de arquiteturas de blockchain (YLI-HUUMO et al., 2016). Além de diferentes arquiteturas, a blockchain tem várias limitações de desempenho em comparação com outras soluções de pagamento conhecidas, como os pagamentos com cartão por exemplo, que são processados em poucos segundos. A blockchain enfrenta em menor ou maior grau, de acordo com a aplicação e implementação, limitações de escalabilidade, latência, taxa de transferência, controle de versão, usabilidade, desperdício de recursos, entre outros (SWAN, 2015; YLI-HUUMO et al., 2016). A figura 1 compila as principais limitações da tecnologia, estas são bem conhecidas pela comunidade e são abordadas constantemente na literatura. Estes pontos são os principais ofensores que impedem uma expansão mais acelerada do uso da tecnologia.

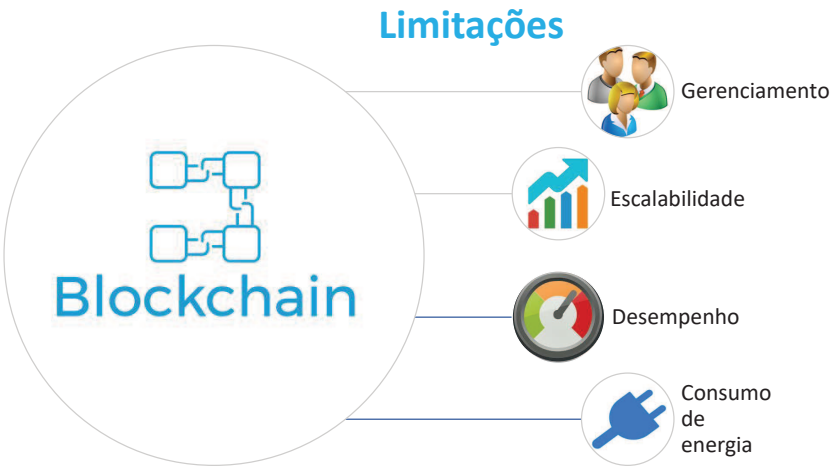

Figura 1 – Principais limitações da tecnologia Blockchain
Fonte: elaborado pelo autor

Para garantir a segurança e o consenso em sua rede distribuída, a blockchain utiliza mecanismos que degradam o desempenho do sistema e dependendo do protocolo de consenso utilizado ainda resultam em um alto consumo de energia. Além de causar um problema de sustentabilidade, se compararmos os aspectos de escalabilidade e capacidade de processamento com outros sistemas centralizados bem conhecidos, percebemos que a blockchain ainda performa muito abaixo e tem muito o que melhorar. Na figura 2 é possível ter uma dimensão deste cenário, onde verificamos a capacidade de processamento de transações simultâneas de duas das principais plataformas blockchain que são Bitcoin e Ethereum e as comparamos com os sistemas da VISA, PayPal e Twitter. Podemos perceber que enquanto o Bitcoin chega a 7 TPS (CROMAN et al., 2016; BITCOIN AND ETHEREUM VS VISA AND PAYPAL – TRANSACTIONS PER SECOND, 2018) e o Ethereum a 15 TPS (SCHERER; ERIKSSON, 2017;

BITCOIN AND ETHEREUM VS VISA AND PAYPAL – TRANSACTIONS PER SECOND, 2018), os sistemas centralizados do Paypal (BITCOIN AND ETHEREUM VS VISA AND PAYPAL – TRANSACTIONS PER SECOND, 2018), VISA (VISA FACT SHEET, 2018) e Twitter (TWEETS PER SECOND RECORD, 2018) conseguem processar picos de 450, 65.000 e mais de 140.000 TPS respectivamente.

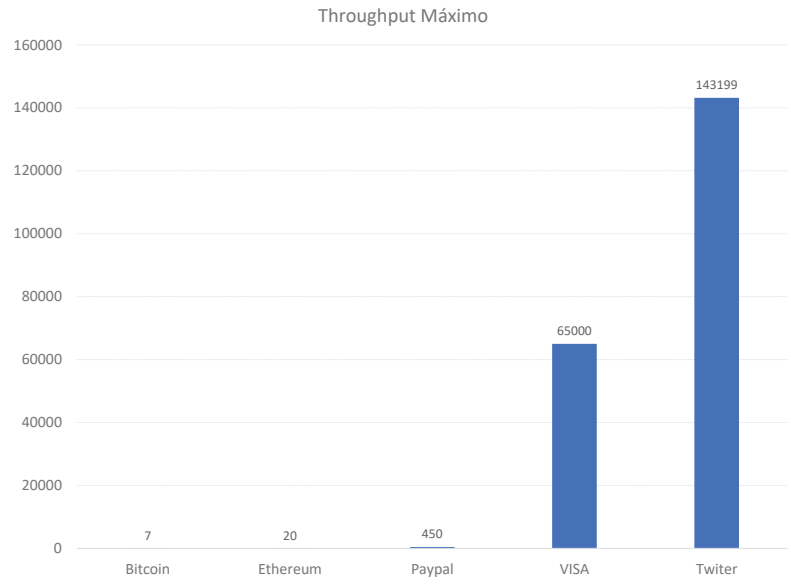

Figura 2 – Comparação da capacidade de plataformas Blockchain com outros sistemas centralizados
Fonte: elaborado pelo autor

Um ambiente descentralizado também pode ser difícil para administrar, uma vez que a configuração do sistema deve ser replicada em diversos nós, ficando passível de erros operacionais. Falando de uma rede privada, estabelecida para realizar operações entre instituições, o acesso à infraestrutura e o conhecimento dos administradores pode não estar equalizado, resultando em problemas no gerenciamento da rede. Por esse motivo, desde que haja um consenso da rede, um gerenciamento centralizado pode ser vantajoso, trazendo maior agilidade e minimizando o risco de erros.

Apesar destas limitações, a blockchain tem um grande potencial para ser a tecnologia que irá revolucionar o mercado de diversas áreas, trazendo novas soluções. Não só existe a possibilidade de a tecnologia blockchain poder reinventar todas as categorias de mercados monetários, pagamentos, serviços financeiros e economia, mas também oferecer possibilidades de reconfiguração semelhantes para todas as indústrias, e ainda mais em geral, para quase todas as áreas do empreendimento humano. Entre algumas áreas que já exploram a utilização da blokchain como plataforma, pode-se citar serviço DNS, registros de saúde, registros de governo, entre outros. A blockchain é fundamentalmente um novo paradigma para organizar atividades com menos fricção e mais eficiência, e escala muito maior do que os paradigmas atuais (SWAN, 2015).

Os resultados já obtidos com a utilização da tecnologia e o seu grande potencial para novas

aplicações faz com que a comunidade e pesquisadores ao redor do mundo busquem melhorar o desempenho da blockchain. Neste contexto, principalmente no ambiente privado existe uma ótima oportunidade para a criação de soluções que gerem ganho de desempenho e tornem a blockchain mais adequada para suportar negócios de diferentes naturezas, melhorando também o nível de alguns dos seus atributos de qualidade como compatibilidade, manutenibilidade, disponibilidade e usabilidade.

1.2 Questão de pesquisa

Tendo em vista as lacunas e desafios expostos a respeito da tecnologia blockchain, o modelo proposto nesta dissertação busca responder a seguinte questão de pesquisa:

Como seria um modelo flexível de arquitetura para a blockchain privada capaz de melhorar o desempenho e gerenciamento do sistema sem comprometer a segurança e descentralização, sendo compatível com as soluções existentes?

Melhorar o desempenho do sistema significa realizar transações de forma mais rápida e garantir que um maior volume delas possa ser processada de forma simultânea, estabelecendo limites de latência e *throughput* além dos praticados atualmente. Obter um gerenciamento mais eficiente consiste em monitorar todos os recursos e funcionalidades da rede blockchain de forma centralizada, possibilitando realizar modificações no ambiente de forma mais ágil e com menor esforço. A flexibilidade do modelo se dá por meio de um fácil acoplamento com o *middleware* blockchain e disponibilização de variados tipos de serviços que podem ser combinados conforme desejado, e ainda tenham seu acionamento de forma estática conforme configuração pré-definidas ou dinâmica a partir de regras que atuam com base em dados coletados em tempo de execução da aplicação. Ofertar compatibilidade significa possibilitar um acoplamento e a ativação da solução de forma transparente para os usuários finais e demais componentes do sistema, não necessitando de alteração no código fonte ou configuração das plataformas atuais para funcionar. Tudo isto sem comprometer a descentralização, utilizando o *ledger* e protocolo de consenso distribuído da blockchain, usufruindo dos seus benefícios, sem haver a necessidade de uma terceira parte confiável para realizar as transações.

1.3 Objetivos

Buscando proporcionar melhor desempenho, mantendo as características de segurança e descentralização da blockchain, esta dissertação propõe um modelo de arquitetura para a blockchain privada, que pode servir como plataforma para diversos tipos de aplicações. As limitações de desempenho atuais identificadas neste estudo impedem que a tecnologia seja utilizada mais amplamente, por isto existe a oportunidade da criação de mecanismos para melhorar estas lacu-

nas e fazer com que a sua utilização ganhe ainda mais força. O modelo proposto se chama L7SP e possui uma camada intermediária que se acopla ao *middleware* blockchain, ela tem a responsabilidade de disponibilizar serviços para a rede blockchain de forma transparente. Esta topologia viabiliza o monitoramento das variáveis de desempenho da rede em tempo real, contribuindo para que os requisitos estabelecidos neste estudo sejam atendidos. A camada de gerenciamento é centralizada e autônoma, e atua também como um balanceador de carga, que distribui a carga de trabalho uniformemente entre os nós validadores, a fim de otimizar a utilização de recursos, maximizar o desempenho, minimizar o tempo de resposta e evitar a sobrecarga do sistema. De acordo com as características descritas acima, foi estabelecido o seguinte objetivo para o trabalho:

Desenvolver um modelo de arquitetura blockchain aplicada ao ambiente privado, que seja acoplado de forma transparente às soluções existentes e ofereça a capacidade de adicionar serviços para prover ganho de desempenho e um gerenciamento mais eficiente. O modelo deve operar com o empilhamento de serviços de forma estática ou dinâmica, e sua implementação deve adotar estratégias que resultem em um menor tempo de resposta das transações e melhor aproveitamento dos recursos do sistema.

O modelo desenvolvido visa atender este objetivo com a premissa de que será realizado apenas um *setup* inicial por parte do usuário, portanto, não haverá a necessidade de intervenção em tempo de execução da aplicação. Para alcançar este objetivo, foram elencados os seguintes objetivos específicos:

- Realizar uma revisão literária dos conceitos necessários para compreensão da tecnologia blockchain e suas principais características;
- Por meio da análise do estado da arte, identificar lacunas com relação às estratégias de melhoria de desempenho em blockchain;
- Desenvolver um modelo que enderece soluções para as lacunas e oportunidades identificadas;
- Desenvolver um protótipo do modelo para a realização de testes;
- Realizar testes para mensurar o desempenho do sistema em diferentes condições, identificando limitações e ganhos obtidos com a aplicação do modelo;
- Avaliar os resultados obtidos.

1.4 Organização do texto

Esta dissertação está organizada em sete capítulos. Neste capítulo de introdução foi feita uma abordagem inicial do tema e foram apresentados os objetivos de pesquisa. O capítulo 2 traz

a revisão de literatura abordando os principais conceitos sobre blockchain e assuntos específicos contemplados no modelo L7SP. O capítulo 3 apresenta a análise do estado da arte por meio dos trabalhos relacionados, listando os critérios de seleção dos mesmos e destacando as lacunas identificadas. No capítulo 4 é apresentado o Modelo L7SP contemplando as decisões de projeto, sua arquitetura de sistema e técnicas utilizadas. No capítulo 5 é apresentada a metodologia de avaliação, que demonstra os cenários de testes e as métricas utilizadas. O capítulo 6 apresenta os resultados obtidos a partir da aplicação dos métodos definidos no capítulo anterior. Por fim, o capítulo 7 traz as considerações finais, apresentando as contribuições e sugestões de trabalhos futuros.

2 CONCEITOS SOBRE BLOCKCHAIN

Este capítulo apresenta os principais conceitos relacionados a tecnologia blockchain, baseados no conhecimento disponível atualmente. A seção 2.1 apresenta as principais características da tecnologia, a estrutura de dados do sistema e como é o seu funcionamento. Na seção 2.2 são apresentados os recursos de segurança existentes na blockchain. Na seção 2.3 e 2.4 são apresentadas as categorias de blockchain e os protocolos de consenso respectivamente. Este capítulo além de proporcionar um entendimento sobre os elementos mais importantes da blockchain, mostra como eles estão fortemente relacionados e podem interferir diretamente na segurança e desempenho do sistema.

2.1 Estrutura de dados e Transações

A blockchain é formada por uma sequência de blocos, que mantém uma lista completa dos registros de transações como um livro razão distribuído que chamamos de *ledger* (CHUEN, 2015). A figura 3 ilustra a estrutura de dados da blockchain e como os blocos são encadeados. O cabeçalho de cada bloco contém um hash do bloco anterior, de modo que cada bloco tem apenas um bloco pai. O primeiro bloco de uma blockchain não possui um bloco pai e é conhecido como bloco Gênesis (ZHENG et al., 2017). Vamos então ver o interior de uma blockchain em detalhes.

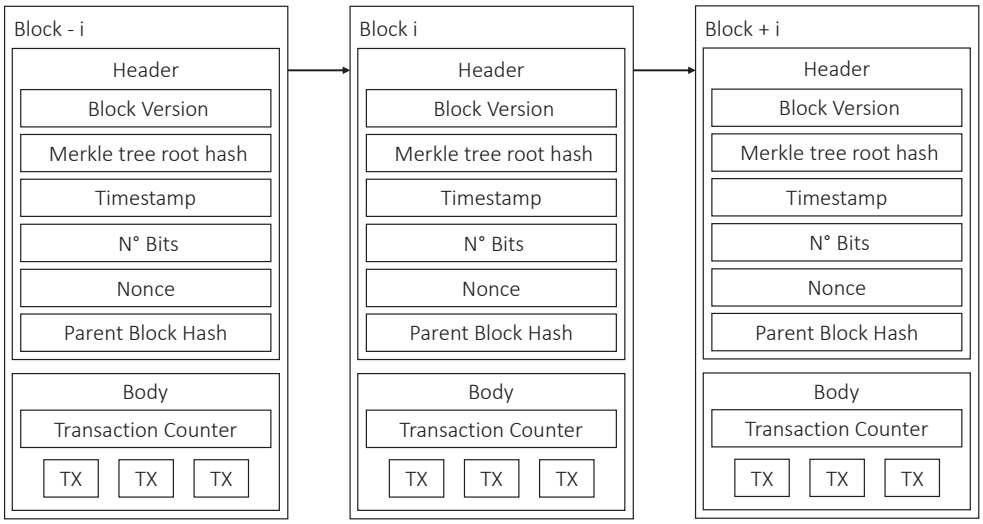

Figura 3 – Blockchain - Estrutura de dados
Fonte: adaptado de NAKAMOTO (2008)

Na blockchain, cada transação executada é armazenada dentro de um bloco, os blocos são

armazenados ordenadamente e com registro de data e hora. Um bloco é formado pelo seu cabeçalho (*header*) e seu corpo (*body*), como também é mostrado na figura 3 (BITCOIN.ORG, 2009). O cabeçalho do bloco inclui as seguintes informações (ZHENG et al., 2017):

1. *Block version*: versão do bloco, indica qual conjunto de regras de validação de bloco deve ser seguido.
2. *Merkle tree root hash*: valor de criptografia de hash de todas as transações do bloco.
3. *Timestamp*: identifica quando o evento ocorreu, informando a data e hora atual (considerando segundos).
4. *nBits*: refere-se ao alvo, é o valor limite para do hash para que o bloco seja válido.
5. *Nonce*: um campo de 4 bytes que corresponde a um valor aleatório, geralmente começa em 0 e aumenta para cada cálculo de hash.
6. *Block Main hash*: um valor de hash de 256 bits que aponta para o bloco anterior.

Um contador de transações e as respectivas transações compõem o corpo do bloco. O número máximo de transações que um bloco pode conter depende do tamanho do bloco e do tamanho de cada transação. A blockchain usa um mecanismo de criptografia assimétrica para validar a autenticação das transações (INSTITUTE, 2016). A assinatura digital baseada em criptografia assimétrica é usada em um ambiente não confiável. O processo de assinatura digital é apresentado a seguir.

2.2 Segurança

Para fornecer segurança no Blockchian é necessário garantir a autenticidade e rastreabilidade dos dados, a fim de evitar transações não autorizadas. Todas as transações são gerenciadas entre os nós da rede, portanto, não há necessidade de terceira parte confiável ou autoridade central. Para atender esses requisitos, foi projetada uma arquitetura de sistema onde os registros podem ser armazenados, facilmente verificados e bloqueados de forma segura, mantendo de forma estável o ecossistema contra ataques de usuários mal-intencionados. A Blockchain usa tecnologias de segurança já conhecidas, como criptografia de chave pública, assinatura digital e hash (BOZIC; PUJOLLE; SECCI, 2016).

2.2.1 Criptografia de chave pública

Criptografia de chave pública ou criptografia assimétrica (SIMMONS, 1979) é um método criptográfico que usa um par de chaves diferentes para criptografia e descriptografia. O problema de entregar chaves existentes na criptografia de chave simétrica foi resolvido dividindo-se a chave em uma para uso privado (chave privada) e uma disponível para qualquer pessoa

(chave pública) (PRPIC PHD, 2017). A implementação da criptografia de chave pública de maneiras específicas pode afetar dois resultados diferentes: privacidade e autenticação (DIFFIE; HELLMAN, 1979)

- Privacidade: um remetente criptografa uma mensagem com a chave pública do destinatário (receptor) e envia a mensagem criptografada para ele. O receptor então descriptografa a mensagem com a chave privada.

- Autenticação: um remetente criptografa uma mensagem com sua chave privada e em seguida torna a mensagem pública ou a envia para um destinatário (receptor). O receptor ou qualquer outra pessoa pode usar a chave pública do remetente para verificar se os dados criptografados foram de fato criados pelo remetente. Usar uma chave privada para criptografar uma mensagem e em seguida tornar pública a mensagem criptografada para a verificação de outras pessoas pela chave pública é chamado de assinatura digital (DIFFIE; HELLMAN, 1979)

2.2.2 Assinatura Digital

Uma assinatura digital refere-se a um mecanismo para provar a autenticidade dos dados usando criptografia de chave pública. Geralmente, uma assinatura digital é feita criptografando o valor de hash de um arquivo com a chave privada do remetente. Ele é enviado ao destinatário juntamente com o arquivo original. O receptor usa a mesma função hash do remetente para criar o valor hash do arquivo e o compara com o valor de hash obtido por meio da descriptografia da assinatura digital do remetente com a chave pública do remetente, confirmando que a assinatura digital do remetente é autêntica (INSTITUTE, 2016). Na blockchain, as assinaturas digitais resolvem o problema de confiar em terceiros, fornecendo autenticidade e integridade através da criptografia de chave pública (SANKAR; SINDHU; SETHUMADHAVAN, 2017). Como mencionado anteriormente, o processo de autenticação na blockchain inclui duas fases: fase de assinatura e fase de verificação. A chave privada usada na fase de assinatura deve ser mantida em segredo, ela é usada para assinar as transações, que são transmitidas por toda a rede e são validadas por nós completos usando a chave pública (ZHENG et al., 2017). O algoritmo típico de assinatura digital usado em blockchains é o algoritmo de assinatura digital de curva elíptica (ECDSA) (D. JOHNSON; VANSTONE, 2001). No sistema Bitcoin, a criptografia de chave pública e a assinatura digital são usadas para identificar um criador de dados de transação e também como um endereço de uma carteira Bitcoin (INSTITUTE, 2016).

2.2.3 Hashing

As funções hash aplicam uma função matemática para converter dados de tamanho arbitrário em uma nova *string* de comprimento predefinido e fixo que é chamada de hash. A principal

característica e a grande vantagem da criptografia hash é que o cálculo inverso para obter os dados originais deve ser o mais difícil possível, enquanto um hash (dados criptografados) deve ser fácil de ser computado (PRPIC PHD, 2017). O processo de hashing é considerado o principal recurso de segurança da blockchain (BOZIC; PUJOLLE; SECCI, 2016).

Mesmo possuindo todos esses recursos de segurança, a blockchain também apresenta algumas vulnerabilidades. *Forks* acidentais podem acontecer na blockchain, eles são comuns e podem ser resolvidos rapidamente. Os invasores podem aproveitar esse comportamento para realizar o ataque de gasto duplo (*double spending*) e a mineração egoísta. Gasto duplo (KARAME, 2012) significa gastar algum dinheiro mais de uma vez com sucesso. Nesse cenário, o invasor envia uma transação para um bloco na bifurcação, espera que ela seja aceita pelos usuários e em seguida, envia uma transação conflitante para outro bloco na ramificação principal (DINH et al., 2017). Na mineração egoísta, os mineradores atacam para aumentar sua participação relativa na mineração na blockchain, retendo blocos já processados e apenas publicando-os gradualmente (GERVAIS et al., 2016). O invasor interrompe os incentivos para mineração e força outros participantes a se unirem à ação mal-intencionada. É possível controlar toda a geração de blocos da rede comprometendo 25% dos nós (DINH et al., 2017). Para evitar gastos duplicados, diferentes estratégias foram usadas para confirmar que uma transação é adicionada com segurança à blockchain. A primeira é esperar que um número pré-determinado de blocos (6 para Bitcoin e 12 para Ethereum) tenha sido gerado após a transação ser incluída na blockchain. A segunda é adotar um mecanismo de *checkpoint*, onde todos os participantes aceitarão as transações até o *checkpoint* como válidas e irreversíveis. Se uma bifurcação começa de um bloco antes que o ponto de verificação ocorra, ela não é aceita por nenhum dos participantes (XIWEI XU INGO WEBER, 2017).

Os blocos nos *forks* (também chamados de blocos órfãos) representam a janela de vulnerabilidade na qual o invasor pode realizar o gasto duplo ou a mineração egoísta (DINH et al., 2017). Como podemos ver, apesar de ter uma arquitetura robusta que usa diferentes mecanismos para fornecer segurança em sua rede distribuída, ainda existem lacunas a serem resolvidas na blockchain. Há também *trade-offs* entre as propriedades fundamentais da blockchain, no que diz respeito principalmente à segurança e desempenho. Algumas decisões de design, como tamanho e frequência de blocos, protocolo de consenso, tipo de blockchain e estrutura de dados, afetam diretamente características importantes do sistema, como escalabilidade, segurança, eficiência de custos e desempenho (XIWEI XU INGO WEBER, 2017). Existem requisitos, como a necessidade de tempos de resposta baixos e armazenamento de dados confidenciais, que se revelam um desafio para a segurança e a escalabilidade. Na maioria dos casos, ao usar uma blockchain para aumentar a segurança, a escalabilidade e o desempenho da aplicação são significativamente reduzidos. Isso se deve principalmente à sobrecarga dos diferentes algoritmos de consenso que protegem a blockchain (SPASOVSKI; EKLUND, 2017). Nas próximas seções, podemos verificar outros conceitos e recursos da blockchain e como eles estão relacionados à segurança do sistema e são diretamente influenciados por ela. Melhorar o desempenho na

blockchain mantendo a segurança é um grande desafio na evolução de sua arquitetura.

2.3 Categorias dos sistemas Blockchain

De forma resumida e em alto nível, os atuais sistemas blockchain podem ser categorizados em três tipos: Blockchain Pública, Blockchain Privada e Blockchain Consórcio (BUTERIN, 2015). Abaixo podemos explorar essas categorias em mais detalhes.

2.3.1 Blockchain Pública

Na blockchain pública, todos os registros são visíveis e todos podem participar do processo de consenso. Os nós podem entrar e sair da rede a qualquer momento que quiserem. Por esse motivo, esse tipo de sistema blockchain também é conhecido por blockchain *permissionless* (sem permissão) (BUTERIN, 2015). Como os registros são armazenados em um grande número de participantes, é quase impossível adulterar transações em uma blockchain pública, porém, leva muito tempo para propagar as transações, tendo latência mais alta (ZHENG et al., 2017). Este modelo foi proposto no Bitcoin e é usado pelas principais criptomoedas. Para garantir a confiabilidade e incentivar os usuários a disponibilizar seu poder computacional para a validação de transações, mecanismos de consenso são usados para alcançar a concordância da rede e recompensar os usuários que validaram as transações. Uma vez que a blockchain pública é totalmente descentralizada, uma das principais vantagens é a tolerância a falhas, pois como a rede consiste em vários nós e todos têm a mesma capacidade, se um nó falhar ou sair da rede, sua operação não será afetada. Outro ponto relevante é que a característica de anonimato é mais percebida nesse tipo de blockchain. A principal desvantagem da blockchain pública é o desempenho, porque consome um poder computacional muito grande para gerar consenso na rede e recompensar os mineradores. Devido às regras de consenso, as transações têm um tempo de validação alto (BUTERIN, 2015). É importante notar que um design que funciona bem para blockchains públicas sem permissão construído em torno de uma criptomoeda não é necessariamente adequado para aplicações de negócios que desejam se beneficiar da tecnologia de *ledger* distribuído (DLT), sem depender de uma criptomoeda específica (VUKOLIĆ, 2017).

2.3.2 Blockchain Privada

Na Blockchain Privada, apenas os nós pertencentes a uma organização específica podem participar do processo de consenso. Esse tipo de blockchain geralmente é usado por instituições onde o controle de informações é necessário, mas a estrutura da tecnologia blockchain de validação criptográfica é vantajosa (BUTERIN, 2015). Diferente da blockchain do Bitcoin, a blockchain privada não precisa implementar um sistema de recompensa e somente os nós que possuem permissão podem participar da rede, então ele tem um desempenho melhor que

a blockchain pública, processando as transações em muito menos tempo (PONGNUMKUL; SIRIPANPORNCHANA; THAJCHAYAPONG, 2017). A blockchain privada tem uma confiabilidade maior do que a blockchain pública porque os dados são centralizados em um único local com segurança e são totalmente controlados por uma organização. Por causa dessas características, não é bom em fornecer anonimato e pode ser considerado como uma rede centralizada (BUTERIN, 2015). Como existem menos participantes, a blockchain privada pode ser adulterada mais facilmente, porém é mais eficiente em propagar as transações, possuindo latência mais baixa (ZHENG et al., 2017). As blockchains privadas com permissão (*permissioned*) evoluíram como uma alternativa às blockchains sem permissão (dos quais qualquer um pode participar), para atender à necessidade de executar a tecnologia blockchain entre um conjunto de participantes conhecidos e identificáveis que precisam ser explicitamente admitidos na rede blockchain (VUKOLIĆ, 2017).

2.3.3 Blockchain Consórcio

A Blockchain Consórcio formada por várias organizações também é conhecida como parcialmente descentralizada, uma vez que apenas um grupo de nós pré-selecionados determinam o consenso. Apenas um pequeno número de nós selecionados pode armazenar na blockchain e executar as validações das transações. As outras operações, como leitura e solicitações, variam de acordo com a implementação, mas geralmente podem ser executadas pelos outros nós da rede (BUTERIN, 2015). Assim como na blockchain privada, as transações na blockchain consórcio são mais fáceis de ser adulteradas, pois há apenas um número limitado de participantes. Por possuir menos validadores, também se torna mais eficiente que a blockchain pública (ZHENG et al., 2017). As blockchains consórcio valem como uma alternativa para evitar o custo de desempenho que as blockchains descentralizadas geram ao distribuir confiança (CROMAN et al., 2016). Atualmente, Hyperledger (HYPERLEDGER FABRIC, 2018) e Ethereum (ETHEREUM - BLOCKCHAIN APP PLATFORM, 2018) fornecem ferramentas para a construção de blockchains consórcio para aplicações de negócio.

Da mesma maneira que foi detalhado acima, Zheng avaliou os principais aspectos que diferenciam e ajudam a classificar os tipos de Blockchain. De forma mais resumida e objetiva, na tabela 1 podemos ver uma comparação entre as 3 categorias baseados nestes aspectos (ZHENG et al., 2017).

2.4 Algoritmos de consenso

Consenso significa concordância, e em termos de blockchain não é diferente. Os algoritmos de consenso ajudam uma rede descentralizada a tomar uma decisão por unanimidade sempre que necessário (SANKAR; SINDHU; SETHUMADHAVAN, 2017). Chegar a um consenso em ambiente distribuído é considerado um desafio. Protocolos de consenso são essenciais

Tabela 1 – Comparação entre os tipos de Blockchain

Propriedade	Blockchain Pública	Blockchain Privada	Blockchain Consórcio
Determinação de consenso	Todos	Uma oraganização	Nós selecionados
Permissão de Leitura	Pública	Pública ou restrita	Pública ou restrita
Imutabilidade	Quase impossível adulterar	Possível ser adulterado	Possível ser adulterado
Eficiência	Baixa	Alta	Alta
Centralização	Não	Sim	Parcial
Processo de consenso	Permissionless	Permissioned	Permissioned

Fonte: adaptado de ZHENG et al. (2017)

na Blockchain para garantir que *ledgers* em diferentes nós sejam consistentes (ZHENG et al., 2017). Depois que as transações são criadas, elas precisam ser verificadas pela rede. No entanto, pode existir uma divergência de ramificações em uma blockchain, uma vez que cada nó pode ter uma visão diferente de todo o estado da rede. Os mecanismos distribuídos fornecidos pelos protocolos de consenso ajudam a mitigar esse problema (ZHENG et al., 2016). Os recursos existentes incluem a garantia de governança descentralizada, estrutura de quorum, autenticação, integridade, não-repúdio, tolerância a falhas bizantinas e desempenho (SANKAR; SINDHU; SETHUMADHAVAN, 2017). O término do consenso indica que um bloco foi definido para o próximo índice de bloco disponível. Dizemos que todas as transações de um bloco definido estão concluídas (NATOLI; GRAMOLI, 2016). Como chegar a um consenso sobre uma transação entre nós não confiáveis é comparado ao problema dos generais bizantinos, onde um grupo de generais que comandam uma parte do exército bizantino cerca a cidade. Alguns generais preferem recuar, enquanto outros preferem atacar. Eles precisam chegar a um acordo para atacar ou recuar, porque os ataques irão falhar se apenas parte dos generais decidirem atacar a cidade (L. LAMPORT; PEASE, 1982). Abaixo são apresentadas algumas das abordagens mais comuns e conhecidas para estabelecer consenso na blockchain.

2.4.1 PoW - Proof of Work

PoW é o mecanismo de consenso mais utilizado nas blockchains existentes. Foi introduzido pelo Bitcoin (NAKAMOTO, 2008) e assume que cada nó fornece seu poder computacional para resolver um problema criptográfico (um enigma) e construir os blocos (GERVAIS et al., 2016). Esse enigma é difícil de resolver, mas é fácil de ser verificado e leva um tempo aleatório. Os nós da rede competem para resolver esse enigma para cada bloco, usando grandes quantidades de energia de computador (consequentemente eletricidade) para aumentar suas chances de ganhar

a competição para submeter um bloco (XIWEI XU INGO WEBER, 2017).

Muito trabalho também precisa ser feito para provar que o nó não tem intenção de atacar a rede. Os nós que publicam um bloco de transações são chamados de mineradores e o procedimento PoW é chamado de mineração em Bitcoin (ZHENG et al., 2017). Neste problema criptográfico, os mineradores precisam encontrar um número aleatório específico chamado "nonce", que é dado como entrada para uma função hash juntamente com os dados da transação. Cada hash representa um número entre 0 e o valor máximo de um número de 256 bits, o *nonce* esperado é encontrado quando a função hash retorna um valor (hash) mais baixo do que o destino. O alvo do Bitcoin, por exemplo, é um número de 256 bits, e quanto menor o alvo, mais difícil é gerar um bloco. Se o hash estiver abaixo do alvo, a mineração está feita (BOZIC; PUJOLLE; SECCI, 2016).

Quando um nó atinge o valor alvo, ele transmite o bloco para os outros nós da rede e eles devem confirmar a correção do valor do hash. Se o bloco for validado, os nós incluirão esse novo bloco em suas próprias blockchains (ZHENG et al., 2017). Se o problema foi resolvido com sucesso, o minerador recebe uma recompensa por realizar isso, junto com uma pequena taxa de transação coletada de cada transação no bloco (GÖBEL et al., 2016). Esse incentivo deu origem a um número de *pools* de mineração concorrentes e desencadeou uma corrida para ter alta capacidade de *hash rate* (DEV, 2014). Resolver o problema criptográfico requer alto poder computacional, portanto, o uso convencional da CPU não é comum para esse propósito. Os mineradores começaram a usar os recursos de processamento paralelo de Unidades de Processamento Gráfico (GPUs). A dificuldade do problema tem aumentado com o tempo, então os mineradores evoluíram para o uso dos FPGAs, e depois o uso de hardware dedicado customizado construído em torno de ASICs (GÖBEL et al., 2016).

Ainda sobre procedimentos de mineração, para resolver com sucesso um bloco, é necessário combinar todos os dados de entrada com o *nonce* correto de tal maneira que o hash resultante comece com um certo número de zeros. O trabalho médio requerido é exponencial ao número de bits zero requeridos. O número de zeros iniciais e o limite superior de valor especificado para o cálculo de um novo bloco (destino) é determinado pelo fator de dificuldade. O fator de dificuldade é ajustado de acordo com a produção de um novo bloco, ou seja, a descoberta de um hash menor que o valor específico, que ocorre em média a cada 10 minutos. A velocidade de descoberta do próximo hash necessário relaciona-se diretamente com a taxa de hash total de todos os mineradores combinados, o que, por sua vez, resulta diretamente na variância do fator de dificuldade do próximo trabalho (DEV, 2014). Ajustar a dificuldade de POW significa mudar o intervalo do bloco, que define a latência na qual o conteúdo é gravado no blockchain. Quanto menor o intervalo de blocos, mais rápida é a confirmação de uma transação e aumenta a probabilidade de existência de blocos obsoletos. Uma dificuldade maior resulta em menos blocos gerados na rede, enquanto uma menor dificuldade resulta em mais blocos, considerando o mesmo período de tempo. Portanto, é essencial analisar se a alteração da dificuldade afeta a capacidade de atacar a cadeia mais longa, que é o principal pilar de segurança da maioria das

blockchains baseadas em PoW (GERVAIS et al., 2016).

As características do PoW ajudam a prevenir ataques em uma rede descentralizada, exigindo que quem deseja realizar alguma ação na rede tenha que fazer algum trabalho antes. É possível que um nó mal-intencionado tente adicionar exatamente a mesma transação a um novo bloco no *ledger*, isso resultaria na bifurcação da cadeia. Portanto, é necessário provar qual das transações duplicadas é autêntica, ou seja, qual é a versão correta da cadeia. A transação correta pode ser considerada como aquela que foi enviada na rede primeiro, o que é difícil de determinar, então a maneira de determinar qual cadeia é autêntica, é verificando seu comprimento em número de blocos. No protocolo PoW, uma cadeia que se torna mais longa é considerada autêntica e será aceita, e a outra cadeia será declarada como não confiável, por isso será descartada (BOZIC; PUJOLLE; SECCI, 2016).

Forks acidentais (ramificações) também podem ocorrer porque vários nós podem encontrar o *nonce* correto quase ao mesmo tempo, gerando blocos válidos simultaneamente. Da mesma forma, a regra da cadeia mais longa ajuda a resolver essa divergência (ZHENG et al., 2017). Se os *forks* ocorrerem, naturalmente, todos os mineradores que trabalham em favor da transação autêntica continuarão a adicionar novos blocos, tornando a cadeia com a transação correta no *fork* por mais tempo. O nó que quer executar uma fraude terá que fazer sua cadeia ser aceita como a mais longa por meio de muitas tarefas computacionais, o que fica muito difícil. Para cometer uma fraude com sucesso, os nós maliciosos exigem pelo menos 51% do poder de computação de todos os participantes da rede blockchain (BOZIC; PUJOLLE; SECCI, 2016).

Embora o PoW seja um mecanismo muito eficiente contra ataques, ele não pode garantir segurança completa, com riscos de comprometer a rede se houver um grupo de nós maliciosos dominando o poder computacional da rede. Outro ponto negativo é que consome muita energia e poder de computação, já que os nós gastam seus ciclos de CPU resolvendo desafios (enigmas), em vez de fazer trabalhos úteis (DINH et al., 2017). De fato, 90% dos sistemas públicos de blockchain empregam variantes do protocolo de prova de trabalho. O PoW é não-determinístico e computacionalmente caro, o que não o torna adequado para aplicações como bancos e finanças, que devem lidar com muitas transações de maneira determinística e com um baixo tempo de resposta (DINH et al., 2017).

As garantias de segurança das blockchains de PoW variantes não receberam muita atenção na literatura. Estudos recentes sugerem que o desempenho de blockchains baseados em PoW não pode ser melhorado sem afetar sua segurança. No entanto, a relação entre o desempenho e as provisões de segurança de blockchains PoW até agora não foi estudada com muito detalhe (GERVAIS et al., 2016).

2.4.2 PoS - Proof of Stake

Em comparação com o PoW, o Proof-of-stake (PoS) é um mecanismo de consenso alternativo e mais eficiente, onde, para obter consenso, os nós responsáveis por verificar e publicar os

blocos são selecionados proporcionalmente à sua participação na blockchain. Em criptomoedas significa a posse de moeda (número de moedas) dentro da rede blockchain, enquanto em blockchains com outro tipo de aplicação, abordagens alternativas como votação são necessárias (SPASOVSKI; EKLUND, 2017). O procedimento para publicar blocos consiste primeiramente em um participante "bloquear"um número de moedas para poder publicar blocos na rede. Depois, o participante precisa gerar um bloco com um hash válido, então o bloco é publicado e validado por outros participantes (SIKORSKI; HAUGHTON; KRAFT, 2017). A operação correta é o interesse principalmente dos nós com maior participação na rede, pois uma ação mal-intencionada pode resultar na redução da confiabilidade e desvalorização da moeda, funcionando como um incentivo para que qualquer participante evite atos desonestos. Por outro lado, a seleção baseada no saldo da conta é considerada injusta porque a única pessoa mais rica é forçada a ser dominante na rede (ZHENG et al., 2017).

Na figura 4 é apresentada uma comparação entre PoW e PoS, dando destaque para as principais diferenças entre eles. Diz-se que a prova de participação é mais rentável, pois elimina o desperdício de recursos e a despesa computacional do PoW. Além de ser mais sustentável em relação à mineração, devido à economia de energia elétrica que ela fornece (XIWEI XU INGO WEBER, 2017). Outros benefícios são a imunidade à centralização de hardware e a redução do risco de qualquer membro adquirir o controle da participação, pois seu custo pode ser maior do que o custo de aquisição de quantidade significativa de energia de mineração (SIKORSKI; HAUGHTON; KRAFT, 2017).

Algumas abordagens diferentes de PoS tem sido usadas. Uma delas é a seleção de blocos randomizados que propõe uma fórmula que busca o menor valor de hash em combinação com o tamanho da participação, estratégia usada pela BlackCoin e NXT. Outra é a seleção que combina a randomização com a idade da moeda, como é usado em Peercoin (TASCA; THANABALASINGHAM; TESSONE, 2017). Outras plataformas que usam PoS são o Tendermint usado em Eris e o Casper usado no Ethereum. Eles têm objetivos de design diferentes, favorecendo uma propriedade não funcional em relação a outra (XIWEI XU INGO WEBER, 2017). Muitas Blockchains adotam PoW no início e gradualmente se transformam em PoS, este é o caso de Ethereum por exemplo (de Ethash para Casper) (ZHENG et al., 2017). Embora o PoS resolva vários problemas encontrados no PoW, ele sofre do problema chamado de "nothing at stake". Ao contrário do POW, há pouco custo em trabalhar em várias cadeias, portanto, os geradores de bloco não têm nada a perder, votando em várias histórias de blockchain, o que leva a um consenso que nunca se resolve. O problema do "*nothing at stake*"foi abordado pelo Delegated Proof of Stake (DPoS), que é uma evolução do protocolo PoS (TASCA; THANABALASINGHAM; TESSONE, 2017).

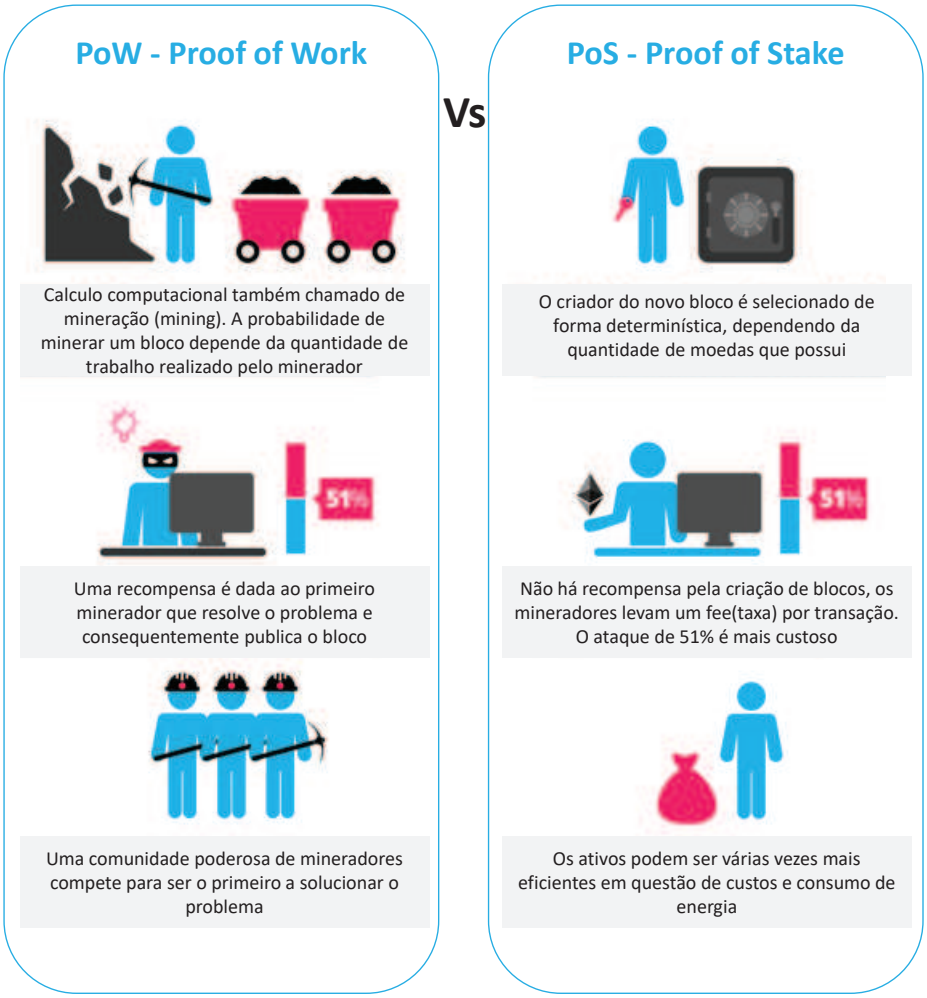

Figura 4 – Comparação entre PoW e PoS
Fonte: elaborado pelo autor

2.4.3 DPOS - Delegated proof of stake

DPoS é um protocolo muito semelhante ao PoS. A principal diferença entre os dois é que no DPoS os nós delegados por gerar e validar os blocos são eleitos pelas partes interessadas, e é necessário menos nós para conseguir isso, contribuindo para que o blocos possam ser confirmados rapidamente (XIANG et al., 2017). No DPoS, os parâmetros de rede, como tamanho de bloco e intervalos de bloco, podem ser ajustados pelos delegados. Além disso, não é necessário se preocupar com delegados desonestos porque eles podem ser facilmente removidos (ZHENG et al., 2017). O DPOS implementa uma camada tecnológica de democracia para compensar os efeitos negativos da centralização, de modo que o DPOS é chamado de democrático representativo, enquanto o PoS é democrático direto (XIANG et al., 2017). O DPOS é a espinha dorsal do Bitshares (BITSHARES, 2018).

2.4.4 PBFT - Practical byzantine fault tolerance

PBFT é uma versão do algoritmo BFT (Bizantin Fault Tolerant), usado para validação de transação em blockchains privadas. Esse tipo de protocolo de consenso tem a função de alcançar concordância entre nós distribuídos na rede, sendo capaz de tolerar falhas bizantinas. O protocolo é capaz de tolerar até f nós defeituosos, onde f é uma fração arbitrária e conhecida do número total de nós. No PBFT, conforme demonstrado na figura 5, um novo bloco é gerado em uma rodada, onde um cliente envia uma solicitação para a máquina principal (primário) que é responsável por fazer o pedido da transação e replica a solicitação para os outros servidores, que também são chamados de backups. Os backups executam a solicitação e enviam uma resposta de volta. O resultado será considerado correto se o cliente receber f + 1 respostas iguais com assinatura válida de diferentes réplicas (BOZIC; PUJOLLE; SECCI, 2016). Essa característica contribui para um nível de disponibilidade mais alto (ANIELLO et al., 2017). Se o primário estiver com defeito e um cliente não receber as respostas, o cliente enviará a solicitação para todas as réplicas, em vez de apenas enviá-la para o primário (ZHENG et al., 2017).

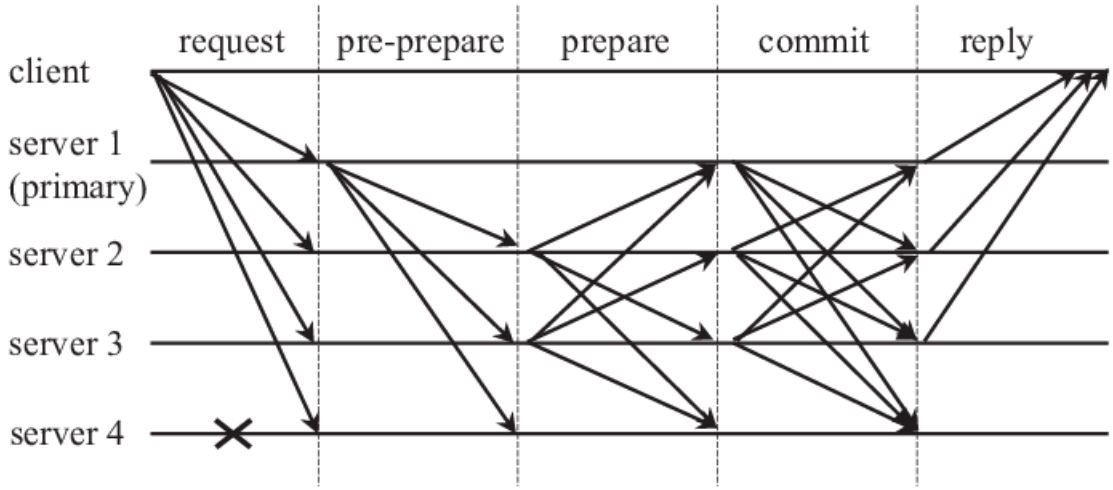

Figura 5 – Fluxo de mensagens do algoritmo PBFT
Fonte: QICHAO ZHANG ZHUYUN QI (2018)

É difícil aplicar este algoritmo em sistemas públicos porque o número total de nós deve ser conhecido e o número máximo de nós ilegais deve ser definido (INSTITUTE, 2016). O sistema precisa ser configurado por uma autoridade central e todas as partes precisam concordar com a lista exata de participantes, o que o torna um método apropriado para blockchains privadas e de consórcio. O protocolo BFT é adequado para plataformas baseadas em ativos digitais e sistemas de armazenamento de baixa latência que precisam de muitas transações, mas não exigem uma grande taxa de transferência de dados (CASTRO M, 1999). A blockchain baseada em BFT

oferece um bom desempenho para um pequeno número de nós, não exige qualquer *hashing* e fornece convergência de consenso rápida e eficiente, enquanto a blockchain baseada em PoW oferece boa escalabilidade de nós com baixo desempenho (SCHERER; ERIKSSON, 2017). O PBFT supera em muito os sistemas PoW Blockchain tanto na latência de transações quanto no *throughput*, no entanto, ele sofre de limitações de escalabilidade (CROMAN et al., 2016). Configurações envolvendo protocolos BFT receberam pouca atenção na literatura acadêmica, mas são de considerável interesse na prática, e seu uso tem sido explorado ativamente por instituições financeiras convencionais (CROMAN et al., 2016). PBFT é usado como algoritmo de consenso no Hyperledger Fabric, também é utilizado em Ripple, Stellar e está programado para ser adotado em Orb (INSTITUTE, 2016).

Além das abordagens anteriores apresentadas em maiores detalhes, existem inúmeros outros algoritmos de consenso que possuem um uso menos amplo, possuem menor notoriedade na literatura e não serão alvos desta pesquisa. Também baseados em BFT podemos citar ainda a existência dos algoritmos Ripple (DAVID SCHWARTZ NOAH YOUNGS, 2014), Tendermint (KWON, 2014) e Stellar (MAZIèRES, 2014), que possuem algumas variações no mecanismo de consenso e regras de seleção dos nós validadores. Existem também outras variações do algoritmo PoS e versões que combinam mais de um mecanismo de consenso, como por exemplo o PoA que é uma abordagem híbrida que combina prova de trabalho e prova de participação.

3 TRABALHOS RELACIONADOS

Este capítulo relaciona os principais estudos sobre as limitações de desempenho da blockchain e propostas para solucioná-las. A Seção 3.1 apresenta a metodologia utilizada para a seleção dos trabalhos, a Seção 3.2 apresenta os trabalhos selecionados, e a Seção 3.3 apresenta o comparativo entre cada um dos estudos.

3.1 Metodologia para Seleção dos Trabalhos

A metodologia para seleção dos trabalhos segue algumas das diretrizes propostas por Kitchenham e Charters (B; S, 2007). O protocolo de estudo desenvolvido permite identificar, avaliar e interpretar todos os dados disponíveis e relevantes para responder a questão de pesquisa. Seguindo os passos do método adotado é possível: resumir as evidências existentes sobre blockchain com foco em arquitetura e desempenho, identificar lacunas na pesquisa atual, e obter uma base para posicionar adequadamente novas atividades de pesquisa. O primeiro passo é encontrar um conjunto completo de estudos relacionados a questão de pesquisa. Esta etapa consiste em definir o escopo da pesquisa e as palavras-chave de busca. As palavras-chave de busca são elaboradas para obter os resultados mais adequados para ajudar a responder a questão de pesquisa. Para isto, foram elencadas as principais bases de dados eletrônicas para pesquisa em engenharia de software, conforme apresentado na tabela 2.

Tabela 2 – Bases de dados de Literatura

Database	**URL**
ACM Digital Library	http://dl.acm.org
CiteSeerX Library	http://citeseerx.ist.psu.edu
EBSCOhost	http://eds.a.ebscohost.com/ehost
Google Scholar	https://scholar.google.com
IEEE Xplore	http://ieeexplore.ieee.org
Ledger Journal	https://ledgerjournal.org/
PLOS One	http://journals.plos.org/plosone/
PubMed	https://www.ncbi.nlm.nih.gov/pmc/
ScienceDirect	http://www.sciencedirect.com
Springer Link	http://link.springer.com
SSRN	https://www.ssrn.com/
Wiley Online Library	http://onlinelibrary.wiley.com

Fonte: elaborado pelo autor

Com base na questão de pesquisa, foram definidas as palavras-chave de busca, delimitando a pesquisa em dois grandes tópicos: arquitetura e desempenho. Este foi o ponto de partida para iniciar a pesquisa, porém, conforme sugerido por Petticrew e Roberts (PETTICREW; ROBERTS, 2006), foram identificados novos termos para serem incluídos e combinados para obter o resultado esperado. Petticrew e Roberts (PETTICREW; ROBERTS, 2006) recomendam a

utilização de sinônimos, siglas, abreviaturas e ortografias alternativas para serem combinados usando operadores booleanos, consequentemente ajudando a enriquecer os resultados da pesquisa. A evolução da configuração das palavras-chave de pesquisa é apresentada na tabela 3. As palavras-chave base são derivadas da questão de pesquisa, elas representam o escopo geral da pesquisa. Depois disso, conforme citado anteriormente e recomendado pelo método, foram identificados alguns sinônimos e termos relacionados, assim foi definido um conjunto de palavras-chave expandido e mais completo. O conjunto de palavras-chave expandido foi usado na primeira rodada de consultas nos bancos de dados de literatura. O principal objetivo desta rodada é obter os artigos sobre blockchain que tenham uma abordagem mais técnica sobre a tecnologia, com foco em questões sobre arquitetura e desempenho. Dos estudos obtidos na primeira rodada foram extraídos outros termos específicos, que são considerados como um novo conjunto de palavras-chave que é usado para ampliar o escopo da pesquisa. Assim, foram adicionados estes novos termos ao conjunto de palavras-chave expandido para gerar um terceiro conjunto, identificado como palavras-chave final. As *strings* de busca utilizadas em cada passo são apresentadas na tabela 4. Este conjunto de palavras-chave representa o escopo completo da consulta nos bancos de dados de literatura. A pesquisa realizada com base neste conjunto de palavras-chave resultou em um total de 267 artigos. O resultado dessa consulta foi considerado para análise posterior neste estudo, conforme descrito a seguir.

Tabela 3 – Palvaras-Chave - Estrutura

Passo	**Palavras-Chave**
1	Palavras chave base
2	Palavras chave base + Palavras chave expandidas
3	Palavras chave base + Palavras chave expandidas + Palavras chaves específicas

Fonte: elaborado pelo autor

Tabela 4 – Palavras-Chave - String de Busca

Passo	**Palavras-Chave**
1	$Blockchain \wedge (Architecture \vee Performance)$
2	$(Blockchain \vee DistributedDatabase \vee Cryptocurrencies \vee SmartContracts)$ $\wedge((Architecture \vee ArchitecturalAssessment \vee ArchitecturalAnalysis) \vee$ $(Performance \vee Latency \vee Scalability))$
3	$(Blockchain \vee DistributedDatabase \vee Cryptocurrencies \vee SmartContracts$ $\vee Bitcoin \vee Ethereum \vee HyperLedger) \wedge ((Architecture \vee Architectural$ $Assessment \vee ArchitecturalAnalysis) \vee (Performance \vee Latency \vee Scalability)$ $\vee(ConsensusAlgorithm \vee Mining \vee ProofofWork \vee ProofofStake))$

Fonte: elaborado pelo autor

Uma vez que os artigos foram obtidos a partir dos mecanismos da Web, aqueles que não eram relevantes para este estudo foram removidos. Um grande número de artigos não responde à pergunta de pesquisa, o principal motivo para isso acontecer é que uma palavra-chave pode ter significados diferentes ou ser usada em estudos que não estão conectados com o objetivo

principal do trabalho. Os critérios de inclusão e exclusão dos artigos devem basear-se na questão de pesquisa, eles podem ser encontrados na literatura, como em Kitchenham (B; S, 2007) e Biolchini (BIOLCHINI J MIAN P; G, 2005), ou ser definidos pelos próprios pesquisadores. Abaixo seguem os critérios de inclusão e exclusão definidos para este trabalho:

1. Merge e Remoção de artigos duplicados: os estudos de bancos de dados individuais foram agrupados e os arquivos duplicados foram eliminados;

2. Revisão de Título e Resumo: O título e o resumo de todos os estudos selecionados na etapa 1 foram revisados, aqueles que não abordam aspectos técnicos relacionados a arquitetura e desempenho em Blockchain foram removidos;

3. Análise de texto completo: Foi lido o texto completo dos artigos selecionados na etapa 2. Todos eles foram analisados e removidos apenas aqueles que não abordam as limitações de arquitetura e desempenho da Blockchain e não apresentam propostas para solucioná-las;

3.2 Trabalhos Selecionados

Esta seção apresenta os trabalhos relacionados seguindo os critérios de seleção apresentados na seção anterior. Os trabalhos aqui listados têm o objetivo de apresentar quais são as limitações de desempenho da Blockchain e apresentar propostas para melhorar estas lacunas.

3.2.1 Bitcoin-NG: A Scalable Blockchain Protocol

O Bitcoin-NG, é um novo protocolo blockchain projetado para escalar. Com base no protocolo blockchain do Bitcoin, o Bitcoin-NG é tolerante a falhas bizantinas e compartilha o mesmo modelo de confiança, evitando mudanças qualitativas no ecossistema. Além de introduzir o Bitcoin-NG, Eyal (EYAL et al., 2015) apresentou várias novas métricas de interesse na quantificação da segurança e eficiência dos protocolos blockchain do tipo Bitcoin. O Bitcoin-NG serializa as transações, muito parecido com o Bitcoin, mas possibilita melhorar a latência e largura de banda sem sacrificar outras propriedades. O protocolo divide o tempo em épocas. Em cada época, um único líder é responsável pela serialização das transições de máquinas de estado. Para facilitar a propagação do estado, os líderes geram blocos. O protocolo introduz dois tipos de blocos: blocos chave para eleição de líder e micro-blocos que contêm as entradas do ledger. Cada bloco possui um cabeçalho que contém, entre outros campos, a referência exclusiva de seu predecessor, ou seja, um hash criptográfico do cabeçalho predecessor. A segurança do protocolo deriva de sua compatibilidade de incentivo, motivando os participantes a seguir as regras.

Blocos chave: Blocos chave são usados para escolher um líder. Como um bloco Bitcoin, um bloco de chaves contém a referência ao bloco anterior, a hora GMT atual, uma transação

com base em moeda para pagar a recompensa, um valor de destino e um campo que contém bits arbitrários. Para que um bloco de chaves seja válido, o hash criptográfico de seu cabeçalho deve ser menor que o valor de destino. Ao contrário do Bitcoin, um bloco de chaves contém uma chave pública que será usada nos micro-blocos subsequentes. Como no Bitcoin, para um minerador gerar um bloco de chaves, ele deve iterar através de valores de nonce até que a condição de criptografia seja satisfeita. Como resultado, o intervalo entre os blocos chave é distribuído exponencialmente. Para manter uma taxa média ajustada, a dificuldade é ajustada alterando deterministicamente o valor alvo com base na hora GMT nos cabeçalhos dos blocos chave.

Micro-blocos: Uma vez que um nó gera um bloco chave, ele se torna o líder. Como líder, o nó tem permissão para gerar micro-blocos a uma taxa definida menor do que um máximo predeterminado. A taxa máxima é determinística e pode ser muito mais alta que o intervalo médio entre os blocos chave. O tamanho dos micro-blocos é limitado por um máximo predeterminado. Especificamente, se o *timestamp* de um micro-bloco estiver no futuro, ou se a sua diferença com o *timestamp* de seu predecessor for menor que o mínimo, então o micro-bloco é inválido. Este limite proíbe que um líder malicioso sobrecarregue o sistema com micro-blocos. Um micro-bloco contém entradas do *ledger* e um cabeçalho. O cabeçalho contém a referência ao bloco anterior, a hora GMT atual, um hash criptográfico de suas entradas do *ledger* e uma assinatura criptográfica do cabeçalho. A assinatura usa a chave privada que corresponde à chave pública no último bloco de chaves da cadeia. Para um micro-bloco ser válido, todas as suas entradas devem ser válidas de acordo com a especificação da máquina de estados, e a assinatura deve ser válida.

Tempo de confirmação: Quando um minerador gera um bloco chave, ele pode não ter conhecimento de todos os micro-blocos gerados pelo líder anterior. Se a geração de micro-bloco for frequente, esse pode ser o caso comum na troca de líderes. O resultado é um *fork* de micro-blocos curto. Este *fork* é resolvido assim que o bloco chave se propaga para esse nó. Portanto, um usuário que vê um micro-bloco deve aguardar o tempo de propagação da rede antes de considerá-lo na cadeia, para garantir que ele não seja removido por um novo bloco chave.

Mineração: Da mesma forma que no Bitcoin, para motivar a mineração, um líder é recompensado por seus esforços pelo protocolo. A remuneração é composta de duas partes. Primeiro, cada bloco chaves dá ao seu gerador um valor definido. Em segundo lugar, cada registro no *ledger* possui uma taxa. Essa taxa é dividida pelo líder que coloca essa entrada em um micro-bloco e o líder subsequente que gera o próximo bloco chave. Especificamente, o líder atual ganha 40% da taxa, e o líder subsequente ganha 60% da taxa. Por não ser necessário minerar micro-blocos, para evitar ataques de gastos duplos, são utilizados "transações envenenadas"que são entradas no *ledger* que invalidam a transação. Estas transações envenenadas contêm o cabeçalho do primeiro bloco excluído que foi provado a fraude, sendo propagado para o próximo bloco chave antes do ativo ser gasto pelo fraudador.

O Bitcoin-NG foi avaliado e comparado com o Bitcoin em dois conjuntos de experimentos, variando a frequência do bloco e o tamanho do bloco. Observou-se que é possível melhorar o atraso de consenso e a largura de banda do Bitcoin ajustando seus parâmetros, mas seu desempenho se deteriora perigosamente em todas as métricas relacionadas à segurança. A latência do Bitcoin-NG é limitada pelo atraso de propagação da rede, e sua largura de banda pela capacidade de processamento dos nós individuais. O Bitcoin-NG alcança uma melhoria de desempenho ao desacoplar a operação de blockchain do Bitcoin em dois planos: eleição de líder e serialização de transação. Pode-se salientar como contribuições deste trabalho os seguintes pontos:

- Quantifica o desempenho e a escalabilidade do Bitcoin-NG através de experimentos em grande escala, demonstrando que alcança um rendimento significativamente maior e menor latência que o Bitcoin;

- Introduz métricas quantitativas para avaliar os protocolos de consenso do Bitcoin.

3.2.2 Secure High-Rate Transaction Processing in Bitcoin

Este estudo sugere o protocolo GHOST de Sompolinsky (SOMPOLINSKY; ZOHAR, 2015), que melhora a escalabilidade do Bitcoin mudando sua regra de seleção de cadeia. Enquanto em Bitcoin a cadeia com mais trabalho é a cadeia principal, em uma bifurcação no GHOST, um nó escolhe o lado cuja sub-árvore contém mais trabalho, conforme apresentado na figura 6. O benefício desta técnica é que a escolha da sub-árvore mais pesada leva em conta a prova de trabalho que não termina na cadeia principal. Assim, o GHOST melhora a imparcialidade e a utilização de energia de mineração sob alta contenção, além de que a taxa de crescimento é menor, tornando o algoritmo mais performático, não exercendo influência sobre a segurança. No entanto, no GHOST, os blocos nas sub-árvores ignoradas afetam somente a regra de seleção no ponto de ramificação. Além disso, existem alguns desafios na utilização do GHOST. No Bitcoin, a qualquer momento, pelo menos um nó sabe qual é a cadeia principal, pois conhece todos os seus blocos. No GHOST, esse não é o caso, e é possível que nenhum nó tenha informações suficientes para determinar qual é a cadeia principal. Uma solução para encontrar a verdadeira cadeia principal no GHOST é propagar todos os blocos. No entanto, isso expõe o sistema a ataques de negação de serviço, já que um nó mal-intencionado pode sobrecarregar a rede com blocos de baixa dificuldade. Pode haver heurísticas para evitar o perigo de segurança; Avaliando o comportamento do sistema quanto à propagação dos blocos, o GHOST apresenta um desempenho pior que o do Bitcoin, já que a sobrecarga de propagação de todos os blocos superava os benefícios da regra de seleção da cadeia. Trabalhos futuros podem encontrar uma solução para os desafios práticos do GHOST, como por exemplo, propagar apenas um cabeçalho de bloco, otimizar o esforço dos mineradores para altas taxas e grandes blocos, e prevenir ataques de negação de serviço ampliado criando checkpoints ao longo da cadeia.

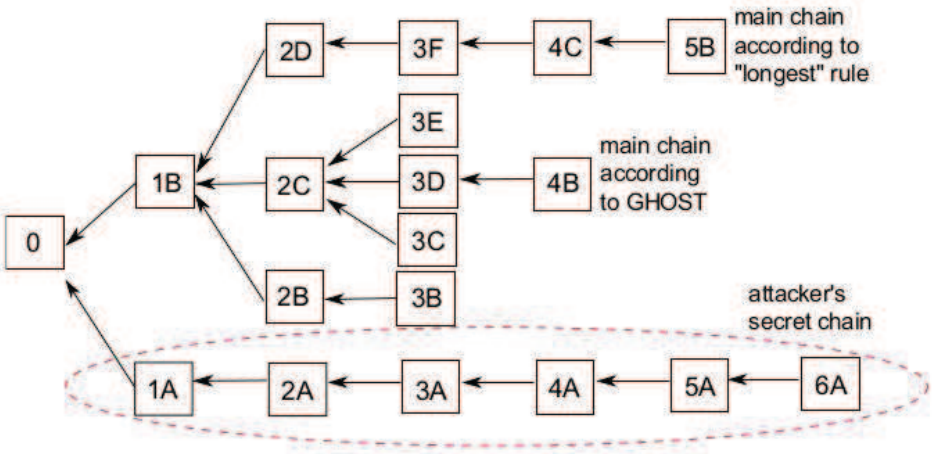

Figura 6 – Árvore de blocos com a seleção da cadeia principal pelo GHOST e a seleção da cadeia principal pelo Bitcoin.

Fonte: SOMPOLINSKY; ZOHAR (2015)

3.2.3 On Scaling Decentralized Blockchains

Neste estudo (CROMAN et al., 2016), são analisados os principais gargalos existentes na blockchain do Bitcoin que limitam sua capacidade de suportar *throughputs* mais altos e latências mais baixas. Os resultados obtidos sugerem que a reparametrização do tamanho e intervalo dos blocos devem ser vistos apenas como um primeiro incremento para obter um protocolo blockchain de alto desempenho e próxima geração. Para alcançar avanços mais significativos, adicionalmente um repensar básico das abordagens técnicas será exigido. O estudo realizado apresenta três contribuições referentes ao problema de escalabilidade da blockchain com foco maior no Bitcoin:

1. Estudo de medição e exploração de reparametrização da blockchain;

2. Compilação e revisão das abordagens técnicas disponíveis que podem ajudar na escalabilidade da blockchain;

3. Posicionamento de desafios abertos.

Os resultados obtidos estão relacionados diretamente com a métrica de taxa de transferência efetiva (*effective throughput*) na rede. Se a taxa de transação exceder a taxa de transferência efetiva de 90%, 10% dos nós da rede não conseguirão acompanhar, o que pode resultar na negação de serviços aos usuários e na redução do poder de mineração efetivo da rede. Para garantir que pelo menos 90% dos nós na rede atual tenham taxa de transferência suficiente, foram estabelecidas duas diretrizes:

- Limite de transferência (*Troughput*): O tamanho do bloco não deve exceder 4MB, dado

hoje a média de 10 minutos de intervalo de bloco . O tamanho do bloco de 4MB corresponde a um *throughput* máximo de no máximo 27 transações/seg.

- Limite de latência: O intervalo de blocos não deve ser menor que 12s, se não a utilização máxima da largura de banda da rede deve ser alcançada.

Assim, foi possível quantificar os atuais limites de escalabilidade do Bitcoin dentro de seus componentes descentralizados. Croman (CROMAN et al., 2016) afirma que é difícil medir com precisão as métricas da blockchain, por isto foi considerado apenas um subconjunto de métricas dentre as possíveis. Isto faz com que os resultados sobre a reparametrização podem ser vistos como limites superiores, possivelmente a inclusão de novas métricas podem revelar limites ainda mais restritos. Uma vez estabelecidos os limites, o estudo de medição evolui com a análise de algumas das principais métricas do sistema Bitcoin como existe hoje:

- Taxa de transferência máxima (*Maximum throughput*): O *throughput* máximo é a taxa máxima na qual a blockchain pode confirmar transações. Hoje, a taxa de transferência máxima do Bitcoin é de 3,3 a 7 transações por segundo. Este número é limitado pelo máximo tamanho do bloco e o tempo entre blocos.

- Latência: Tempo para uma transação ser confirmada. Uma transação é considerada confirmada quando é incluída em um bloco, o que leva aproximadamente 10 minutos.

- Tempo de *bootstrap*: O tempo que leva um novo nó para baixar e processar o histórico necessário para validar o estado atual do sistema. O tempo é linear ao tamanho do histórico da blockchain, e é de aproximadamente quatro dias atualmente para o Bitcoin.

A reparametrização é um fator que gera muitas discussões, a comunidade Bitcoin tem várias proposições em discussão para aumentar o tamanho máximo do bloco. Porém estas estratégias não são compatíveis com as estratégias autuais adotadas pela rede. Embora seja possível dimensionar o protocolo blockchain por meio do ajuste de parâmetros, descobrimos que a taxa de transferência melhor alcançada é significativamente menor que o limite imposto pela infraestrutura subjacente. Croman (CROMAN et al., 2016) elaborou uma discussão estruturada dos desafios de pesquisa e de novas estratégias para solucionar as lacunas existentes nos sistemas blockchain, particionada da seguinte forma:

Plano de Rede: Duas ineficiências no plano de rede do Bitcoin se destacam. Primeiro, para evitar negação de serviço pela propagação de transações inválidas, um nó deve receber e validar completamente uma transação antes de propagá-la ainda mais. Essa validação local de transações contribui significativamente para o tempo de propagação geral. Segundo, o protocolo da camada de rede do Bitcoin primeiro propaga todas as transações e, em seguida, propaga um bloco novamente quando é extraído. Isso efetivamente exige que cada transação seja transmitida duas vezes. Uma possibilidade, para evitar a transferência de cada transação duas vezes, é confiar em um protocolo de reconciliação no qual os nós buscam apenas transações que eles

não possuem em um bloco recém-extraído. Outra opção, usada pelos mineradores hoje, é usar uma rede centralizada e dedicada de alta velocidade para comunicação entre os mineradores.

Plano de consenso: Tentativas de melhoria no desempenho do protocolo de prova de trabalho da blockchain do Bitcoin comprometem a segurança, mas podem ser uma alternativa. Muitas plataformas favorecem a velocidade de consenso em relação à segurança, empregando uma blockchain padrão de Bitcoin com uma alta frequência de geração de blocos. Outra opção é aderir a modelos de consenso privado ou de consórcio da Blockchain, utilizando protocolos alternativos de consenso que eliminem a despesa computacional, como PoS ou BFT. Outra técnica possível para melhorar a escalabilidade do consenso é conhecida como sharding, e consiste em dividir a tarefa de consenso entre conjuntos de nós que funcionam simultaneamente, com o objetivo de melhorar o rendimento e reduzir os requisitos de processamento e armazenamento por nó. Outra técnica de escalonamento é criar uma hierarquia de "instâncias de consenso"de nível inferior, comumente chamadas de "cadeias laterais"ou Sidechains. As cadeias laterais podem potencialmente ter um grau menor de descentralização do que as cadeias de blocos de nível superior. Cadeias laterais também podem executar protocolos de consenso que não utilizam prova de trabalho, como o BFT. A introdução de sidechains levantam alguns desafios técnicos. É importante que sejam protegidas independentemente da blockchain principal e exista coordenação dos mineradores para evitar que se tornem vulneráveis. Outro ponto é que transações entre cadeias podem sobrecarregar a blockchain principal e, portanto, ter um impacto adverso na escalabilidade. Além disso, transações envolvendo mais de uma cadeia descentralizada podem incorrer em alta latência.

Plano de armazenamento: A implementação de referência do Bitcoin hoje, por padrão, armazena todo o *ledger*, como resultado, o sistema armazena muitas duplicatas do *ledger* inteiro. O Plano de Armazenamento em Bitcoin aceita apenas gravações que acrescentam dados, ou seja, blocos recém-extraídos, e não suporta operações de exclusão. A única operação de leitura geralmente suportada para o Bitcoin é fazer o download do conteúdo de todo o ledger. Assim, o Plano de Armazenamento do Bitcoin tem notáveis ineficiências. A comunidade propôs ideias interessantes que podem essencialmente dividir o armazenamento de uma estrutura de dados UTXO, porém ainda apresentam desafios de pesquisa. Tabelas distribuídas de hash (DHT) são um possível começo, juntamente com técnicas de autenticação de dados adequadas.

Plano de visualização: *View* é uma estrutura de dados derivada do ledger completo. Há várias opções para implementar uma visualização, incluindo as seguintes: Visualizações via replicação: Bitcoin, Ethereum e outras criptomoedas descentralizadas populares exigem que todos os nós de consenso verifiquem todas as transações e, com base no resultado da computação, atualizem suas respectivas visualizações. Neste caso, a visualização é uma saída implícita do Plano de Consenso e pode ser considerada como residente na saída de Armazenamento. Terceirização de visualizações via criptografia: É possível terceirizar a computação de uma exibição para um provedor de serviços terceirizado. Este provedor pode liberar um resumo criptográfico (por exemplo, raiz da árvore Merkle) dessa visualização, juntamente com uma prova de sua cor-

reção. A visualização pode então ser inserida no plano de armazenamento. Se a disponibilidade não for essencial, a visualização pode, alternativamente, ser armazenada dentro de alguma outra parte do sistema, por exemplo, pelo próprio provedor, e não no Plano de Armazenamento. Uma vantagem dessa abordagem é que os nós de consenso agora não precisam armazenar o *ledger* inteiro. Em vez disso, eles podem operar sobre visualizações adequadamente escolhidas.

Plano lateral: Crome (CROMAN et al., 2016) afirma que podemos considerar a utilização de funcionalidades fora da cadeia (*off-chain*), utilizando canais "colaterais"pré-estabelecidos, como demonstrado em redes de pagamento. Embora as redes de pagamento tenham sido anunciadas como uma solução para as limitações inerentes do Bitcoin, grande parte de sua operação e as garantias que podem oferecer dependem da natureza dos links formados entre as partes. Mesmo quando as redes de pagamento usam o mesmo formato de transação subjacente do Bitcoin, eles formam essencialmente um Plano de Rede separado e um Plano de Consenso P2P independente, apoiado pelo Bitcoin. Como resultado, sua capacidade, capacidade de encontrar rotas, garantias de *throughput*, latência e privacidade dependem fundamentalmente das propriedades emergentes do esquema da rede de pagamento.

3.2.4 The Mini-Blockchain Scheme

Em Mini-Blockchain (BRUCE, 2014) é proposto um novo esquema para criptomoedas onde transações antigas podem ser esquecidas pela rede. Como os nós exigem apenas a parte mais nova da blockchain para sincronizar com a rede, essa parte da cadeia é chamada de “mini-blockchain”. A introdução de uma pequena “cadeia de prova” resolve a perda de segurança que esse processo de corte gera, e a perda de dados de propriedade de moedas é resolvida com um banco de dados que contém o saldo de todos os endereços não vazios, chamado de "árvore de contas". A cadeia de prova protege a miniblockchain e a mini-blockchain protege a árvore de contas. Esses três mecanismos trabalham juntos para formar um sistema que ofereça um alto nível de integridade e segurança. Ele também oferece outros benefícios potenciais, como transações mais rápidas e taxas mais baixas, sincronização de rede mais rápida, suporte a altos níveis de tráfego, mais espaço de bloqueio para mensagens personalizadas e, potencialmente, aumento do anonimato. Como o esquema mini-blockchain funciona armazenando o saldo de todos os endereços não vazios na "árvore da conta", não é preciso nenhuma das transações para calcular o saldo de qualquer endereço. Foi removido o sistema de scripts e a ideia de transações interligadas , sendo substituído por um conceito muito mais simples, no qual as transações executam operações básicas na árvore da conta, como subtrair moedas do saldo do endereço A e adicioná-las ao saldo de endereço B. As entradas e saídas nas transações não apontam para outras transações, elas simplesmente apontam para endereços na árvore da conta, então as transações não são vinculadas juntas da mesma maneira que no Bitcoin, e pode-se descartar todas as transações depois de decorrido uma quantia segura de tempo, o suficiente para inviabilizar o ataque secreto. Chama-se ataque secreto quando um invasor se baseia na cadeia de

provas legítima em segredo usando uma árvore de contas inválida e, em seguida, libera a cadeia secreta para a rede quando acharem que ela é tão longa que ninguém terá histórico tão antigo. Nessa situação, novos nós não seriam capazes de detectar qual cadeia é a real. Cada conta possui um *hash* correspondente na árvore de contas. se o *hash* de um nó na árvore for alterado, o *hash* da raiz muda. O *hash* da raiz é armazenado nos cabeçalhos dos blocos assim provendo a integridade dos dados pois este é assegurado pela blockchain. Cada proprietário de um endereço não vazio demonstra sua propriedade com uma chave privada e as transações são criadas como um conjunto de dados assinado e propagadas pela rede, da mesma forma que no Bitcoin. Os mineradores também atuam de forma semelhante, resolvendo cálculos complexos para aceitar as transações e adicionar à Mini-Blockchain. A Mini-Blockchain é o mecanismo responsável por coordenar a forma de processamento das transações na rede e possui o diferencial de não ser necessário manter a cópia dos blocos históricos, tornando o plano de armazenamento mais leve e o sistema mais escalável. Apesar desse estudo fazer referência ao Bitcoin e ser mais voltado para criptomoedas, pode-se fazer uma reflexão sobre as ideias aplicadas para que possam ser adaptadas e estendidas para plataformas de blockchain privadas.

3.2.5 Subchains: A Technique to Scale Bitcoin and Improve the User Experience

Subchains (RIZUN, 2016) é uma técnica para usar blocos de dificuldade fracionária (blocos fracos) para construir sub-cadeias que unam pares adjacentes de blocos reais. As sub-redes reduzem o risco de blocos órfãos ao propagar blocos camada por camada durante todo o intervalo de blocos, em vez de todos de uma só vez quando a prova de trabalho é resolvida. Cada nova camada de transações ajuda a proteger as transações incluídas nas camadas inferiores, mesmo que nenhuma das transações tenha sido confirmada em um bloco real. Os mineradores são incentivados a cooperar na construção de sub-cadeias em torno da cadeia principal, conforme a figura 7, assim é possível processar mais transações por segundo sem incorrer em risco adicional de blocos órfãos. O uso de sub-cadeias também desvia a receita das taxas para o hash da rede, em vez de retirá-lo do sistema para pagar por blocos órfãos. Como as sub-cadeias são construídas sobre o protocolo Bitcoin existente, sua implementação não requer nenhuma mudança nas regras de consenso do Bitcoin.

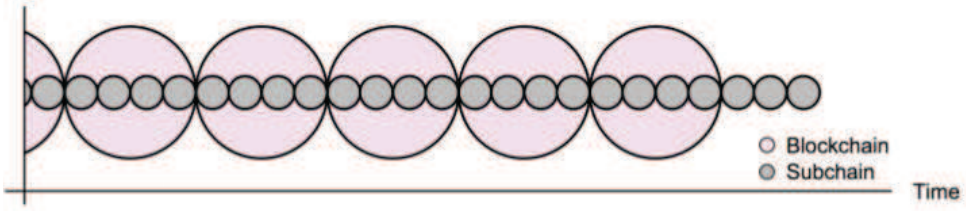

Figura 7 – Sub-cadeias de blocos fracos em torno da cadeia principal
Fonte: RIZUN (2016)

Sub-cadeias são formadas como uma série de blocos fracos, com o próximo bloco fraco

construindo uma nova camada de transações sobre o bloco fraco anterior. Os mineradores transmitem os blocos (fracos e fortes) enviando apenas a última camada de transações junto com um hash que faz referência às camadas anteriores. No caso de haver mais de uma sub-cadeia o sistema elege a cadeia mais longa para serem adicionados os novos blocos fracos e no caso de mesmo tamanho é escolhida a cadeia mais antiga. Subchains apresenta quatro contribuições importantes que surgiriam com a adoção da técnica, desde que ela seja amplamente implantada pelo poder de hash da rede. São elas:

1. Risco de blocos órfãos reduzido para um determinado tamanho de bloco;
2. Continuação da existência de um mercado de taxas de transação;
3. Maior segurança de prova de trabalho da receita de taxas;
4. Melhor segurança de gastos duplos para transações não confirmadas.

A probabilidade de existirem blocos órfãos é reduzida, porque somente as transações mais novas necessitam ser propagadas na rede, reduzindo também o tempo de propagação de bloco e tornando a blockchain mais escalável. Diferente de outras propostas para aumentar a escalabilidade da blockchain, por ser implementado como um recurso extra apenas nos mineradores que queiram contribuir com o subchain e não alterar nenhuma regra ou funcionalidade da blockchain do Bitcoin, o subchain se torna compatível com o Bitcoin. A técnica é outra opção para estimular a adoção do Bitcoin, contribuindo para minimizar o problema do limite de capacidade transacional.

3.2.6 Graphene: A New Protocol for Block Propagation Using Set Reconciliation

Graphene (OZISIK et al., 2017) combina filtros Bloom e IBLT para introduzir um método de reconciliação de conjuntos interativos para distribuição eficiente dos blocos na rede P2P.

Os protocolos blockchain não são eficientes no anúncio de novos blocos pois propagam a mesma transação pela rede por mais de uma vez. Ao invés de enviar todas as transações, no Graphene, os filtros são utilizados para verificar quais transações estão na memória do receptor e comparar com a listagem recebida do emissor, se as informações coincidirem ele produz a listagem de ID's nos blocos.

Ghrapene é comparado a outros dois estudos e se mostra mais eficiente. Enquanto um Xtreme Thinblock de 17,5 KB pode ser codificado em 10 KB com Compact Blocks, a mesma informação pode ser codificada em 2,6 KB com o Graphene. O tamanho do bloco padrão do Bitcoin por exemplo é 1 MB.

O desempenho do Graphene foi avaliado de forma analítica e empírica através de uma simulação de rede detalhada. Devido a redução do tamanho da estrutura de dados, ele reduz a sobrecarga de tráfego para cerca de 60% em comparação com o uso de Compact Blocks se os blocos são enviados a cada 2,5 minutos.

O protocolo é aplicável a uma variedade de protocolos de rede baseados em blockchain, como Bitcoin, Ethereum, Litecoin e Zerocash, porém o seu ganho de performance acaba sendo muito maior no Ethereum, pois a frequência de envio de blocos é maior.

3.2.7 Poster: Low-latency blockchain consensus

Este estudo introduz o Blinkchain, um protocolo de consenso bizantino que depende de técnicas de fragmentação e preservação de localidade de sistemas distribuídos para fornecer um limite na latência de consenso, proporcional ao atraso de rede entre os nós. O Blinkchain seleciona um conjunto aleatório de validadores, algumas das quais são legítimas com alta probabilidade, mesmo quando um atacante concentra suas forças para expulsar validadores legítimos em uma pequena vizinhança. Basescu (BASESCU; KOKORIS-KOGIAS; FORD, 2017) cita que existem duas categorias de soluções que são praticadas para tentar resolver os problemas de latência da blockchain:

- Alterar o algoritmo de consenso para PBFT, no entanto, o PBFT não é dimensionado para escalar grandes grupos de consenso. Assim, a latência do sistema e o uso da largura de banda diminuem com um número crescente de nós.

- Escalar o consenso do Bitcoin, aumentando a frequência de bloco ou o tamanho do bloco, no entanto, os sistemas resultantes sofrem as mesmas falhas de segurança que o Bitcoin.

O Blinkchain é uma blockchain que fornece limites na latência de consenso, utilizando *sharding* como uma técnica para escalar horizontalmente. Esta proposta difere na forma como os validadores são selecionados para a blockchain em cada fragmento. Para obter consenso de baixa latência entre os validadores, são utilizadas técnicas de Crux, que aprimora protocolos distribuídos escalonáveis com baixa latência. Ao mesmo tempo, é mantida a garantia de segurança de que um adversário não pode assumir um determinado fragmento, concentrando seus esforços nas proximidades do fragmento. O Blinkchain divide a blockchain em fragmentos que são associados a áreas geográficas. São associados a cada fragmento vários validadores próximos, ou seja, réplicas de fragmentos, que precisam chegar a um consenso sobre o estado do fragmento. Como de costume nos sistemas tolerantes a faltas bizantinos, é assumido que no máximo 1/3 do número total de validadores está com defeito. O objetivo é garantir que as transações envolvendo dois fragmentos diferentes incorram em uma latência proporcional ao diâmetro de uma área geográfica que cobre ambos, dentro de um fator polilogarítmico. Conforme demonstrado na figura 8, usando as técnicas de Crux, são criadas áreas geográficas sobrepostas centradas em torno de nós no sistema, os chamados anéis, cujo raio de latência aumenta exponencialmente. Os anéis incorporam como membros todos os nós localizados nessa área geográfica.

Para garantir que um invasor não possa influenciar a seleção do validador em um fragmento de gasto de destino, é variado o limite da área que abrange o fragmento de onde se seleciona os validadores. Em uma primeira fase, os validadores são selecionados de um fragmento usando

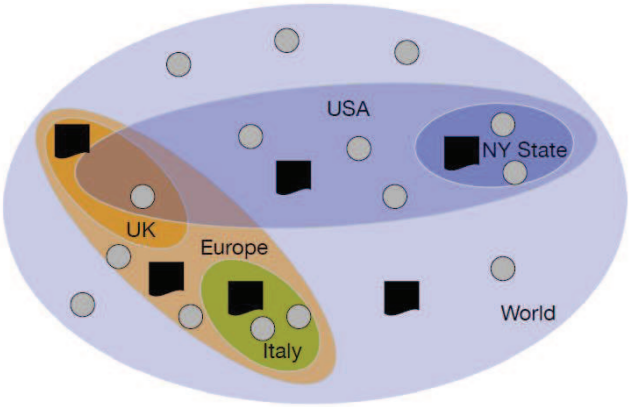

Figura 8 – Exemplo de criação de anel. Os quadrados pretos representam um fragmento em cada anel, enquanto os círculos cinzentos representam os nós.
Fonte: BASESCU; KOKORIS-KOGIAS; FORD (2017)

o *RandHound*. Em uma segunda fase, cada *shard* solicita apenas um subconjunto desses validadores para validar suas transações. O *shard* seleciona no momento da verificação, com base em um indicador aleatório, a porcentagem de validadores escolhidos de alguns de seus anéis. Como o invasor não conhece essas porcentagens, não pode saber o quão próximo do *shard* de destino colocar suas identidades. A segurança decorre do fato de que a atribuição do validador permanece aleatória, o que significa que no máximo 13 dos validadores por fragmento são indignos de confiança em uma escala global. Mesmo que o adversário tente colocar todas as suas identidades perto de um fragmento de vítima, a aleatoriedade garante que, enquanto existirem alguns validadores honestos em dentro de cada anel, pelo menos um deles é selecionado e sinaliza qualquer despesa dupla na vítima.

Para evitar gastos duplicados, o sistema precisa rastrear UTXOs de seu fragmentos de criação para seu fragmento de gastos e validar um único rastreio. No entanto, o Crux só oferece consistência fraca entre fragmentos. Foi adotada uma abordagem baseada em *tokens*, onde associou-se um token a cada UTXOs criado por um *shard*, *token* que é assinado por esse *shard*, e ao qual o *shard* acrescenta sua identidade. Cada UTXO usado como entrada para uma transação possui seu próprio token. Os validadores do fragmento onde os UTXOs são gastos primeiro verificam a assinatura do token de cada UTXO e depois se referem ao fragmento de origem, para ver se esses UTXOs aparecem como saídas. Após as verificações bem-sucedidas, os validadores inserem uma transação no fragmento de origem para finalizar a passagem de *token* entre o fragmento de origem e o fragmento de destino. Como os validadores sempre verificam se o *shard* de origem contém uma transação de "*token-spent*"inserida por outros validadores, eles rejeitam transações de gasto duplo.

3.2.8 A Prototype Evaluation of a Tamper-resistant High Performance Blockchain-based Transaction Log for a Distributed Database

Aniello (ANIELLO et al., 2017) apresenta a arquitetura 2LBC e uma avaliação experimental do seu protótipo. A 2LBC é uma arquitetura de duas camadas para um banco de dados distribuído baseado em blockchain, capaz de fornecer garantias de alta performance e integridade de dados em um ambiente totalmente descentralizado. Trata-se portanto de uma blockchain rápida de primeira camada para garantir baixa latência, ancorado a uma blockchain segura de segunda camada, baseado em prova de trabalho para obter uma integridade forte dos dados.

Como estudo de caso, o trabalho abordou o banco de dados distribuído subjacente ao FaaS, uma solução recente da Federação de Nuvem. A FaaS oferece uma federação democrática de nuvens, que depende crucialmente de um banco de dados distribuído para garantir que nenhum membro da federação possa adulterar dados federados. Claramente, os requisitos da aplicação exigem não apenas integridade de dados, mas também desempenho adequado, por isto, a implementação de um sistema baseado em blockchain rápida e segura é fundamental para alcançar estes requisitos. A figura 9 mostra e a arquitetura 2LBC no contexto do FaaS.

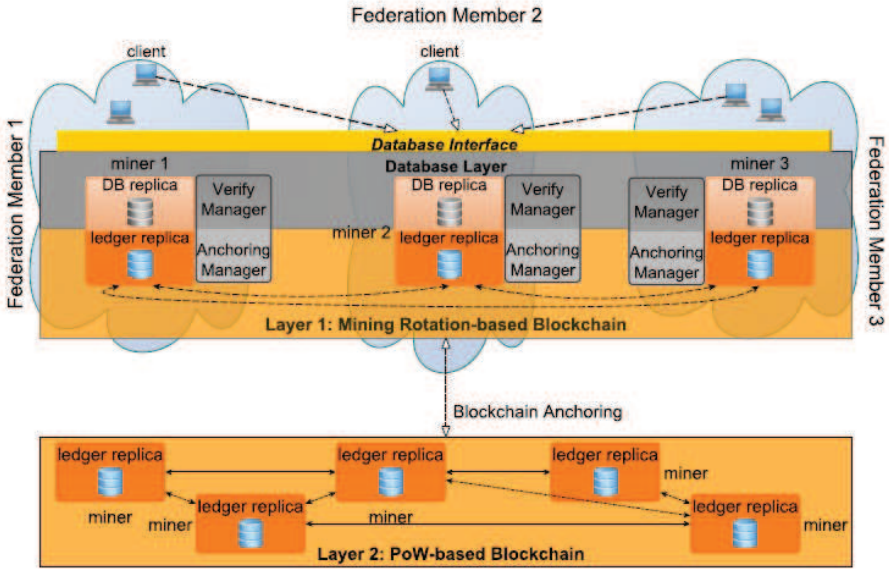

Figura 9 – Arquitetura 2LBC para uma federação FaaS
Fonte: ANIELLO et al. (2017)

Seja M o número de membros da federação, cada um fornecendo N mineradores. O anel DHT inclui M. N nós, o fator de replicação é definido como M, e os mineradores podem ser colocados sobre o anel para que os mineradores de cada membro da federação gerenciem coletivamente todas as chaves. A figura 10 mostra um exemplo de uma federação composta por M = 3 membros, cada um expondo N = 2 mineradores, onde cada membro (M) possui uma cor e

cada minerador (N) possui uma letra específica. Os mineradores são dispostos no anel, de modo que cada membro tem todas as chaves do banco de dados divididas entre seus mineradores. Os únicos mineradores mantêm apenas um subconjunto proporcional ao fator de replicação. No exemplo, o fator de replicação é M = 3 e cada minerador mantém apenas 1/N = 1/2 das chaves do banco de dados.

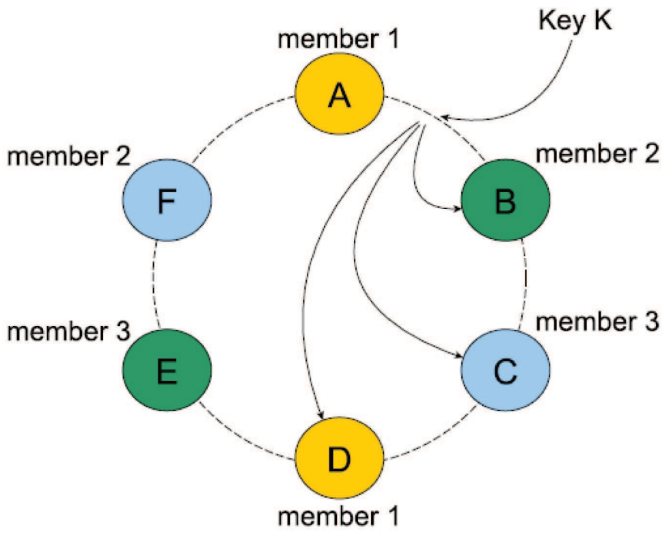

Figura 10 – Exemplo de federação composta por M=3 e N=2
Fonte: ANIELLO et al. (2017)

As principais limitações que foram identificadas inicialmente no protótipo de 2LBC estão relacionadas a problemas de disponibilidade e escalabilidade. O desempenho geral do sistema não aumentava com a adição de novos nós, pois o algoritmo de consenso total usado tem desempenho mais baixo com nós adicionais. As seguintes soluções foram propostas para lidar com estas limitações:

- Disponibilidade: Para lidar com falhas e ataques de DoS foi proposta uma solução tolerante a falhas bizantina baseada em PBFT. Para tolerar até os b mineradores bizantinos, é necessário que 3 mineradores f + 1 forneçam segurança e vivacidade. Isso leva a um nível de disponibilidade mais alto, já que os mineradores honestos precisam esperar apenas f + 1 assinaturas válidas para confirmar uma transação. É possível tolerar até os mineradores silenciosos ao custo de garantias de integridade mais fracas. De fato, a corrupção de dados é possível com apenas f + 1 mineradores comprometidos, ao invés de todos os N quando o consenso total é usado. De qualquer forma, a integridade dos riscos já presenciados na segunda camada ainda é garantida graças ao PoW.

- Escalabilidade: Para melhorar a escalabilidade, foi proposta uma solução de *sharding* de dados. Especificamente, foi introduzido um *ledger* baseado em tabela de hash distribuído

(DHT) no qual cada minerador, na base de um particionamento de espaço de chaves, manipula apenas transações para subconjuntos específicos de chaves. Essa abordagem permite ajustar cargas de transações em mineradores e, consequentemente, torna o sistema mais escalável. Além disso, cada intervalo de chave tem um fator de replicação configurável para habilitar a tolerância a falhas. Ao contrário das implementações comuns de DHT, os mineradores envolvidos em uma operação definida devem obter consenso antes de gravar a operação na réplica local. Isso permite alcançar uma consistência forte, evitando problemas de consistência que são conhecidos como bancos de dados NoSQL baseados em DHT.

3.2.9 Performance and Scalability of Blockchain Networks and Smart Contracts

Neste artigo (SCHERER; ERIKSSON, 2017) é realizada uma comparação da blockchain pública com a blockchain privada, abordando os aspectos de descentralização, escalabilidade e segurança em três das principais redes blockchain: Bitcoin, Ethereum e HyperLedger.

Vejamos as características de cada uma delas e técnicas para melhorar o desempenho e a escalabilidade:

- **Bitcoin:** é a blockchain pública com o mais complicado problema de escalabilidade e também o protocolo mais restrito para mudar. A capacidade de escalonamento do Bitcoin é restrita, uma vez que a taxa de transferência mais alta da transação atingiu o pico máximo no tamanho máximo do bloco dividido pelo intervalo do bloco. Nos últimos anos, o tamanho do bloco foi rapidamente preenchido para tamanhos de 1 MB, em 2017 cada bloco gerado atingiu o máximo de 1 MB, limitando efetivamente o throughput da transação para 2-4 transações/seg. Sem uma solução para bloquear o problema de congestionamento, a transação na blockchain continuará atrasada, e as taxas de transação continuarão a aumentar, portanto, não poderá cobrir o comércio mundial a qualquer momento no futuro próximo. O limite de tamanho de bloco foi definido como 1 MB por Nakamoto em 2008 quando o *white paper* foi publicado, para alterá-lo deve haver um acordo entre todos os mineradores da rede, o que tem sido um longo debate. Um *hard fork* é uma mudança no protocolo Bitcoin que o torna menos restritivo, os mineiros não atualizados não validarão blocos criados por mineradores atualizados e permanecerão. Os softforks implementam novas regras para o protocolo Bitcoin que o tornam mais restritivo, todos os blocos considerados válidos pela versão mais nova também são válidos na versão antiga. A diferença entre o *hard fork* e o *soft fork* é que os *hard forks* não são compatíveis para frente e para trás, enquanto os *soft forks* são. Para implementar mudanças “duras” no protocolo, cada minerador, comerciante e usuário deve atualizar ou ser deixado para trás. O *soft fork* requer apenas que a maioria dos mineradores faça upgrade para aplicar as novas regras. Não há um consenso na comunidade sobre as alterações no Bitcon, de um lado, há uma equipe de desenvolvedores que propôs um recurso através

de um *soft fork* visando otimizar o tamanho dos blocos. Este recurso visa remover dados relacionados a assinaturas da transação Bitcoin, tornando-os menores em tamanho. Isso, por sua vez, torna o bloco ainda menor, significando que mais transações podem ser incluídas no bloco. Do outro lado estão aqueles que queriam um aumento no tamanho do bloco, com mais debates sobre qual seria o tamanho ideal.

A maneira de prolongar a blockchain de maneira *Proof of Work* impacta negativamente a escalabilidade do sistema e o rendimento geral. Em termos de velocidade de transação, um protocolo com tempo de bloqueio mais curto precisa de mais confirmações para o mesmo nível de segurança que um protocolo com maior tempo de bloqueio. Uma confirmação é indiretamente determinada pela dificuldade, que aproximadamente determina quantos *hashes* são necessários para resolver um bloco. Geralmente, o tempo de bloqueio significa uma taxa mais alta que, por sua vez, exige um número maior de confirmação para corresponder à segurança do Bitcoin. Uma taxa mais alta também significa que mais trabalho é desperdiçado. O tamanho do bloco também é um dos principais parâmetros relacionados ao desempenho de uma blockchain PoW, como o Bitcoin. Aumentar o tamanho do bloco com o objetivo de aumentar o rendimento traz o custo de aumentar a latência, devido a atrasos mais longos de propagação de blocos maiores em toda a rede, implicando também na segurança. Isto porque aumenta o risco de bifurcação da cadeia, e consequentemente o risco de ataques de gasto duplo.

- **Ethereum:** pode ser implementado como uma blockchain pública, bem como uma blockchain autorizada. Provavelmente, os maiores problemas de escalabilidade com a Ethereum são que cada nó precisa processar todas as transações e armazenar todo o estado de cada saldo de conta, código de contrato e armazenamento. Embora isso forneça uma grande quantidade de segurança, limita muito a escalabilidade a ponto de uma blockchain não poder processar mais transações do que um único nó. Logicamente uma rede com milhares de nós deveria ser capaz de ter mais throughput do que um único nó, mas este não é o caso no Ethereum ou em redes blockchain públicas em geral. Uma possível solução para esse problema é criar um novo mecanismo em que apenas um pequeno subconjunto de nós precisa verificar um subconjunto de transações. Contanto que haja muitos nós suficientes verificando cada transação, o sistema ainda estará seguro, mas também permitirá que o sistema processe as transações em paralelo. Essa técnica é chamada de fragmentação (*sharding*). A ideia básica por trás do *sharding* é dividir o estado global das contas em partes menores conhecidas como *shard*. Cada *shard* obtém seu próprio conjunto de validadores e esses validadores não precisarão validar todos os *shards*.

  O Ethereum mitigou a maior parte da perda de segurança com o tempo de bloqueio mais rápido usando uma versão modificada do protocolo GHOST, mas os blocos ainda precisam se propagar pela rede, o que é um processo relativamente lento. O tamanho do bloco precisa ser menor para se propagar mais rapidamente e é por isso que o Ethereum só pode

processar cerca de 15 tps, embora o tempo de bloqueio seja de apenas 15 segundos. O Ethereum, por outro lado, está em estado muito pior do que o Bitcoin, já que tudo que a rede Bitcoin precisa fazer é lidar com transações simplistas. A rede do Ethereum precisa lidar com tarefas de processamento arbitrárias e armazenar uma quantidade potencialmente imensa de dados. Para poder dimensionar e fornecer mais *throughput*, é necessário uma solução para dois gargalos: processamento de transação e armazenamento de estado. A solução mais promissora é a escalabilidade *on-chain* via *sharding* e escalonamento *off-chain* complementar via canais, mas levará algum tempo até que essas soluções sejam implementadas.

- **Hyperledger Fabric:** o desempenho e a escalabilidade do Fabric são completamente diferentes do Bitcoin e do Ethereum, devido a ser uma blockchain privada de permissão. Isso permite que o Hyperledger tenha tipos diferentes de nós com suas próprias responsabilidades e permite que eles configurem uma rede de nós para escalar independentemente uns dos outros. Ao comparar Bitcoin e Ethereum com uma blockchain de permissões como o Hyperledger Fabric, fica nítido que sacrificar um pouco da descentralização melhora muito a escalabilidade e o desempenho. Isso significa que precisamos de mais confiança nos nós de validação, e a autoridade e confiança têm que vir de fora da rede. Por ter uma autoridade governamental que fornece um nível inerente de confiança entre os participantes, permite que as decisões de design, como fragmentação e canais, sejam implementadas sem muita complicação. E como a confiança já é obtida de fora da rede, não é necessário muito poder computacional para suportar essa confiança. Isso permite usar mecanismos de consenso diferentes, como os protocolos BFT, mitigando os problemas existentes nas blockchains públicas. Além disso, é possível garantir o acesso aos dados aos participantes do canal, e apenas permitir que eles visualizem dados confidenciais de transações.

A avaliação realizada com o HyperLedger Fabric comprova que a blockchain baseada em BFT oferece um bom desempenho para um pequeno número de nós, enquanto a blockchain baseada em PoW oferece boa escalabilidade de nós com baixo desempenho. O Hyperledger Fabric se demonstrou viável para ser usado no setor financeiro em comparação com soluções centralizadas tradicionais, ou pode atender aos requisitos de negócios impossíveis de atender a uma blockchain pública. Os resultados mostraram que sacrificar a descentralização criando autoridade e confiança fora da rede melhoram desempenho e a escalabilidade. Uma blockchain de permissão não requer poder computacional para suportar essa confiança em comparação com blockchains públicas, o que, por sua vez, elimina os problemas de taxa de transferência. O Hyperledger Fabric é certamente muito mais rápido e escalável que o Bitcoin e o Ethereum, podendo suportar um *throughput* muito mais alto. Embora os bons resultados, ainda há dúvida se o Hyperledger Fabric é rápido e escalável o suficiente para substituir os sistemas centralizados usados atualmente. Para resultados mais precisos, é preciso levar em consideração em avalia-

ções futuras a latência de rede e a largura de banda. A largura de banda certamente pode ter um impacto no tamanho do bloco, blocos maiores levariam mais tempo para serem enviados em comparação com blocos menores. Empresas ou um consórcio de instituições financeiras devem colocar seus computadores que funcionam como pares em estreita proximidade para minimizar a latência tanto quanto possível.

3.3 Análise e oportunidades de pesquisa

A tabela 5 traz uma visão consolidada dos trabalhos que foram discutidos e os relaciona com as principais características e limitações da blockchain, possibilitando a análise e comparação dos mesmos de forma mais objetiva. De acordo com a tabela, podemos ver que a maioria dos trabalhos tem foco na blockchain pública, mais especificamente em propor melhorias para a plataforma da criptomoeda Bitcoin. Este é o caso dos trabalhos Bitcoin-NG (EYAL et al., 2015), GHOST (SOMPOLINSKY; ZOHAR, 2015), Mini-Blockchain (BRUCE, 2014), Subchains (RIZUN, 2016) e Graphene (OZISIK et al., 2017), que são implementações baseadas no protocolo Bitcoin. Com exceção de Subchains, nenhuma destas soluções é compatível com as plataformas existentes, isto quer dizer que são aplicáveis somente à novas implementações do protocolo ou à *hard forks* das plataformas atuais. A maioria deles também aborda a alternativa de realizar reparametrizações no protocolo e sua estrutura de dados, mais especificamente significa alterar o tamanho e frequência de bloco. Estas reparametrizações resultam em complicações, refletindo diretamente na segurança e desempenho de rede. Além disso, a alternativa de reparametrização diverge opiniões entre os desenvolvedores e comunidade no geral, o que a torna difícil de ser aplicada. É possível comprovar que existem diferentes estratégias para melhorar a escalabilidade e desempenho do Bitcoin, mas não há um consenso sobre elas.

O trabalho Bitcoin-NG (EYAL et al., 2015) e GHOST (SOMPOLINSKY; ZOHAR, 2015) propõem alterações na regra de consenso. O Bitcoin-NG elege um nó líder e trata os blocos em épocas ao invés de tempo, ele divide o bloco em mini-blocos aproveitando melhor o poder computacional. Em GHOST a seleção da cadeia foi modificada nos casos de bifurcação, passando a escolher a árvore de maior peso considerando as sub-cadeias ao invés de escolher a árvore de maior profundidade. A implementação do GHOST em conjunto com o Bitcoin-NG pode ser interessante para complementar o Bitcoin-NG e permitir uma frequência maior de blocos de chaves.

O trabalho Graphene (OZISIK et al., 2017) altera a política de propagação do bloco, contribuindo para melhoras no plano de rede. o Graphene otimiza o tráfego de dados, ele aproveita que as transações são propagadas no momento da validação e propaga somente as informações suficientes para que os nós receptores consigam remontar o bloco com as transações que possuem na memória. Já o trabalho Subchains (RIZUN, 2016) propõe adicionar em cada bloco mini-cadeias de blocos fracos que são descartados para processar um volume maior de transações no período de frequência de bloco, aproveitando melhor o poder computacional e me-

lhorando a escalabilidade. A técnica de subchains pode ser adotada pelos mineradores, sendo transparente para o protocolo blockchain.

O trabalho Mini-Blockchain (BRUCE, 2014) é focado em criptomoedas e provê todas as características base da blockchain do Bitcoin. Oferece um grande benefício para o plano de armazenamento, pois passa a armazenar somente as transações mais recentes e diminui o tamanho do *ledger*. No plano de visualização provê mecanismos que permitem que as transações mais antigas sejam consultadas. Possui uma árvore de contas para armazenar os saldos, e mantém a prova de trabalho como algoritmo de consenso, mas altera o cálculo de *hash* e o tempo de inserção das transações nos blocos, sendo as transações inseridas juntas dentro de um determinado tempo.

Buscando avaliar soluções que não sejam aplicadas especificamente ao protocolo Bitcoin temos o Blinkchain (BASESCU; KOKORIS-KOGIAS; FORD, 2017), que se trata de uma implementação do protocolo de consenso bizantino que depende de técnicas de fragmentação e preservação de localidade de sistemas distribuídos para prover uma latência mais baixa para a rede. Já a arquitetura 2LBC (ANIELLO et al., 2017) apresenta uma estratégia híbrida combinando 2 camadas de blockchain para alcançar disponibilidade, escalabilidade e integridade dos dados. A primeira camada utiliza PBFT para garantir baixa latência e a segunda camada utiliza PoW para prover mais segurança. Este trabalho possui uma aplicação específica para log de registros de banco de dados distribuídos, mas traz uma reflexão interação com relação a combinação de estratégias e o uso de diferentes tipos blockchain para atender os requisitos de negócio de uma aplicação.

"On Scaling Decentralized Blockchains" (CROMAN et al., 2016) e *"Performance and Scalability..."* (SCHERER; ERIKSSON, 2017) são dois estudos bem completos que abordam a maioria das limitações que impactam a blockchain e suas principais características. Eles consolidam as abordagens existentes e sugerem futuras direções para muitas delas, mas não apresentam nenhuma implementação de fato. Ambos abordam as questões referentes a reparemetrização na blockchain e são unanimes em sugerir o protocolo de consenso BFT como alternativa de consenso com maior desempenho. Enquanto em *"Performance and Scalability..."* é feita uma comparação entre as características de blockchain pública e privada e são avaliadas as principais plataformas blockchain existentes (Bitcoin, Ethereum e Hyperledger), Em *On Scaling Decentralized Blockchains* é realizado um estudo focado na blockchain do Bitcoin afim de identificar estratégias para conseguir atingir throughputs mais altos e latências mais baixas. Ele apresenta 3 contribuições: medição e exploração de reparametrização da blockchain; compilação das abordagens técnicas disponíveis e posicionamento de desafios abertos. Ele trata dois aspectos de limitações da blockchain, um de parametrização que se refere aos parâmetros do protocolo e outra é de infraestrutura, que são limitações referentes ao desenvolvimento e implementação da blockchain. Ele divide as limitações de infraestrutura nos seguintes planos: rede, regra de consenso, *view*, armazenamento e *side*. Os estudos selecionados abordam a maioria destes gargalos e apresentam propostas de melhorias, com destaque nas regras de consenso e no plano de rede.

Tabela 5 – Comparação das propostas dos trabalhos relacionados em relação às principais características e limitações da blockchain

Trabalho	Ano	Fonte	Tópicos										
			Bitcoin	**Consenso**	**Publico**	**Privado**	**Latencia**	**Escalabilidade**	**Armazenamento**	**Rede**	**Reparametrização**	**Implementação**	**Compatibilidade**
Bitcoin-NG	2016	ACM	✓	✓	✓		✓	✓		✓	✓	✓	
Secure High-Rate.. (GHOST)	2015	Springer Link	✓	✓	✓			✓		✓	✓	✓	
On Scaling Decentralized Blockchains	2016	Springer Link	✓	✓	✓	✓	✓	✓	✓	✓	✓		
Mini-Blockchain Scheme	2017	cryptonite.info	✓				✓	✓	✓	✓	✓	✓	
Subchains	2016	Ledger Journal	✓		✓			✓				✓	✓
Graphene	2017	Springer Link	✓		✓					✓		✓	
Poster: Low-latency blockchain consensus (Blinkchain)	2017	IEEE		✓			✓	✓		✓		✓	
A Prototype Evaluation.. (2LBC)	2017	IEEE		✓			✓	✓				✓	
Performance and Scalability..	2017	Springer Link	✓	✓	✓	✓	✓	✓	✓	✓	✓		

Fonte: elaborado pelo autor

A partir da análise dos trabalhos relacionados foram identificados alguns pontos ainda não explorados que se traduzem em oportunidades de pesquisa futuras em Blockchain:

- Balanceamento de carga: Nenhum dos trabalhos abordou técnicas de balanceamento de carga utilizadas em cluster para tentar distribuir melhor a carga de trabalho do sistema e torná-lo mais escalável, contribuindo também para a melhora de desempenho.

- Acoplamento de serviços externos: Nenhum dos trabalhos abordou estratégias de acoplamento da blockchain com outros serviços e plataformas externas para explorar recursos que combinados com a blockchain ofertem maior desempenho, mantendo a compatibilidade com as soluções existentes.

- Alocação de Recursos: Nenhum dos trabalhos abordou a alocação dinâmica de recursos ou serviços para prover maior capacidade e velocidade no processamento das transações.

- Monitoramento online: Muitos dos trabalhos realizam experimentos e simulações para avaliar o desempenho da blockchain, mas isto é feito de forma planejada, demandando um esforço considerável na execução e medição dos resultados. Medir o desempenho da blockchain é uma tarefa complexa, e atualmente não existem ferramentas que ofertem uma visão da saúde do sistema de forma online.

- Plano de armazenamento: Algumas estratégias foram discutidas para melhorar o plano de armazenamento, como é o caso da Mini-blockchain. Este tópico merece mais atenção

para sejam disponibilizadas novas alternativas. O banco de dados da blockchain cresce linearmente. Com o passar dos anos e o aumento do volume transacional, problemas de desempenho devem surgir em virtude do seu tamanho.

- Blockchain Privada: A maioria das soluções é focada no protocolo Bitcoin, existem poucas implementações para melhorar o cenário da blockchain privada e contribuir para uma maior adoção da tecnologia para outras aplicações no mundo corporativo.

4 MODELO L7SP

Este capítulo apresenta as definições técnicas para implementação do modelo L7SP, uma solução que se acopla à camada de aplicação da blockchain privada e oferece recursos que visam melhorar o desempenho e o gerenciamento do sistema comparado às plataformas de blockchain existentes em sua configuração padrão. O nome L7SP é um acrônimo do termo em inglês *Layer Seven Service Provider*, que remete a uma solução que é capaz de disponibilizar serviços para a blockchain privada na camada de aplicação. No modelo L7SP foi desenvolvida uma camada centralizada de gerenciamento para ser acoplada de forma transparente à plataforma blockchain. Três abordagens de serviço foram implementadas nesta camada para que combinadas contribuam para o ganho de desempenho do sistema e um gerenciamento mais eficiente: (i) balanceamento de carga, (ii) compressão de dados, (iii) monitoramento. Na seção 4.1 são apresentadas as escolhas realizadas levando em consideração o contexto onde a proposta é aplicada e as características que são mantidas ou modificadas na solução blockchain. Na seção 4.2 é apresentado o desenho da arquitetura de sistema, e quais são os componentes, ferramentas e recursos utilizados. Por fim, na seção 4.3 é apresentada a composição do modelo, detalhando as 3 abordagens aplicadas que são disponibilizadas como serviço no módulo gerenciador do L7SP.

4.1 Decisões de Projeto

Dentre as categorias de blockchain que foram apresentadas no capítulo de fundamentação teórica, optou-se por desenvolver uma arquitetura diferenciada com foco na blockchain privada e consórcio, pois estes modelos ofertam maior autonomia para realizar modificações e ainda são pouco explorados pelas organizações como plataforma para outros tipos de negócio além das criptomoedas e contratos inteligentes. Com suas características de acesso mais restrito e um processo de consenso mais leve, a utilização da blockchain privada se torna atrativa para diferentes tipos de aplicação em instituições. A blockchain privada pela sua natureza já oferta um melhor desempenho, pois, esta categoria não sofre com as limitações de parametrização e desempenho da blockchain pública, que requer um esforço computacional considerável para garantir segurança gerando diversos gargalos, como ocorre no Bitcoin. O desafio neste contexto é por meio da implementação de L7SP prover ainda uma melhor escalabilidade e desempenho na blockchain privada, para que ela se torne uma opção ainda mais atrativa para ser adotada pelas corporações frente às arquiteturas de sistema centralizadas. A camada de gerenciamento proposta no modelo L7SP compõe uma arquitetura de alta disponibilidade, alto desempenho e com otimização de uso dos recursos. Para atingir estes objetivos foi fundamental estabelecer os requisitos iniciais durante o *design* da solução, delimitando assim o escopo e direcionando o desenvolvimento. Desta forma, o modelo foi elaborado de modo a atender as seguintes decisões de projeto:

1. Utilizar um *middleware open source* como plataforma de blockchain privada;

2. Utilizar a interface de comunicação disponibilizada pelo *middleware* para interagir com a blockchain e atuar em seu funcionamento;

3. A aplicação proposta deve atuar de forma transparente sem a necessidade de alterações no *middleware* (desenvolvedores não precisam reescrever o código);

4. Garantir que os recursos do sistema sejam homogêneos, ou seja, que as configurações de hardware e software dos nós sejam as mesmas;

5. Para ganho de desempenho e eficiência aplicar técnicas consolidadas em sistemas distribuídos que sejam aderentes a qualquer outra plataforma de blockchain privada ou consórcio;

6. O monitoramento da blockchain e seus nós poderá ser feito de maneira centralizada por um administrador, utilizando os recursos providos pelo modelo e o ambiente adotado;

7. O ambiente para validação consistirá em uma aplicação gerenciadora acoplada à blockchain sendo executada em uma rede P2P em uma nuvem privada.

O modelo proposto se acopla de forma centralizada com a blockchain, os trabalhos relacionados mostram que esta é uma estratégia que não interfere nas principais características providas pela tecnologia e já vem sendo utilizada em outros estudos e aplicações. Uma blockchain privada também pode ser considerada como uma rede centralizada, uma vez que é totalmente controlada por uma organização, e a blockchain consórcio parcialmente descentralizada, constituída por várias organizações e apenas uma pequena porção de nós é selecionada para determinar o consenso (ZHENG et al., 2017). Conforme citado em Pilkington (PILKINGTON, 2015), salienta-se que algumas características conceituais da blockchain podem não ser previstas em uma solução de *ledger* distribuído, mas não se deve abrir mão da imutabilidade, que permanece crucial. A blockchain pode ser até completamente centralizada, desde que todos os seus dados sejam imutáveis e verificáveis externamente. Na tentativa de melhorar a escalabilidade da blockchain, a comunidade já vem propondo abordagens diferentes, combinando estratégias centralizadas com a tecnologia de *ledger* distribuído. Um exemplo disto foi a introdução da rede de retransmissão de Corallo, um mecanismo centralizado de propagação de blocos (CROMAN et al., 2016). Apesar de nesse estudo ser avaliado como um componente centralizado em uma rede mais reduzida, L7SP pode ser escalado para atuar como uma solução parcialmente descentralizada. Isto pode ser viabilizado por meio da utilização de vários servidores para rodar a camada de gerenciamento, funcionando como super nós em uma rede P2P. Com relação a administração do ambiente, Kakavand (KAKAVAND; Kost De Sevres; CHILTON, 2017) aborda sobre a existência de riscos operacionais sistêmicos em uma rede descentralizada. O fato de que apenas uma parcela muito limitada da população entende realmente como a blockchain opera, exige que seja colocada uma quantidade extrema de confiança na habilidade e integridade dos responsáveis por tomar decisões sobre o código e a rede blockchain. Quanto maior o sistema

se torna, mais pressão é colocada sobre esse pequeno grupo de especialistas para fazer escolhas que sejam implementadas com precisão e segurança no código. Além disso, não há um responsável direto por manter o software da blockchain operacional. Isto pode levar a demora na tomada de decisões e na realização de reparos e ações necessárias para resolver uma crise operacional, fatores que podem ser muito críticos em um ambiente privado.

4.2 Arquitetura

Esta seção tem o objetivo de apresentar a arquitetura proposta para L7SP. A arquitetura do sistema consiste na definição dos componentes de software, suas propriedades, e seus relacionamentos com elementos internos e externos. A elaboração do desenho de arquitetura visa facilitar a comunicação entre os *stakeholders*, documentar em alto-nível as decisões iniciais acerca do projeto, e permitir o reuso das técnicas e padrões adotados em projetos futuros.

A blockchain em sua essência é constituída por uma arquitetura P2P, que contém uma série de computadores interconectados entre si, onde cada um atua como um nó na rede. Em geral, todos os nós são considerados hierarquicamente iguais, possuindo os mesmos privilégios e a mesma influência na rede. Logo, podem atuar tanto como clientes quanto servidores, compartilhando exatamente os mesmos dados e recursos. Para realizar as operações, os nós utilizam mecanismos de consenso e confiança na comunicação direta entre eles, sem o intermédio de terceiros. Os nós na rede trocam mensagens entre si contendo transações, blocos e endereços de outros nós. A base de dados é compartilhada por todos os nós na rede P2P, assim quando uma nova transação ocorre, a informação nesta mensagem é propagada entre todos eles.

A diferença na proposta de L7SP com relação à arquitetura convencional da blockchain é a inclusão de um componente centralizado chamado de gerenciador, que tem o objetivo de prover o ganho de desempenho desejado e melhorar o gerenciamento do sistema. Esta mudança não fere os princípios da blockchain privada já abordados, mantendo as características de segurança e descentralização na realização das operações de negócio. A comunicação ocorre em uma rede TCP/IP utilizando os protocolos ofertados pelo *middleware* Multichain, mantendo assim a compatibilidade e facilitando o acoplamento do novo componente. A figura 11 apresenta a topologia da solução, onde temos uma visão da rede privada do Multichain representada pelos nós ligados pela linha tracejada, e L7SP ao centro, interligado a todos os componentes. Podemos identificar na topologia que não há alteração na configuração da rede blockchain privada, mantendo a conexão entre todos os nós participantes, sem haver interferência no protocolo de mensagens. A diferença é que os clientes externos passam a se conectar com o Gerenciador L7SP e este faz o intermédio da comunicação das aplicações externas com os nós, agregando serviços na camada de aplicação.

Funcionalmente a arquitetura proposta é transparente para o usuário, não interferindo na forma como ele se relaciona com o sistema. Para o administrador do sistema a nova arquitetura permite que o monitoramento seja realizado de forma centralizada, ofertando maior facilidade

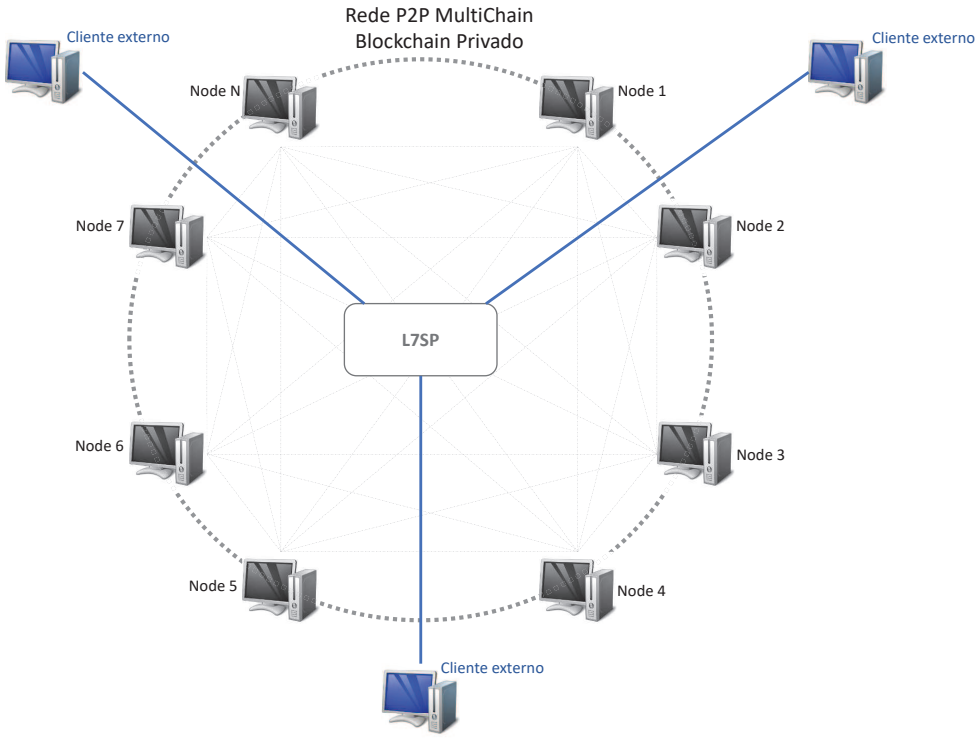

Figura 11 – Topologia da solução
Fonte: elaborado pelo autor

na manutenção e gerenciamento. Para viabilizar a implementação das abordagens citadas no início do capítulo, foi implementado também um agente local que trabalha em sincronia com o gerenciador. O agente local é executado em cada nó blockchain e possibilita o monitoramento do desempenho de cada nó de forma automática e centralizada. L7SP foi acoplado ao *middleware* blockchain sem a necessidade de configurações adicionais ou de alterações no código do *middleware* por parte do desenvolvedor. Na figura 12 é apresentada a arquitetura detalhada da solução, contemplando a estrutura interna de cada componente e a conexão entre eles. Na figura, os novos componentes de software introduzidos pelo modelo L7SP estão destacados em azul, e os componentes já existentes na rede blockchain privada padrão são aprensentados em cinza. As flechas e setas indicam o sentido de comunicação entre os componentes. As requisições JSON-RPC originadas das aplicações externas passam a ser recepcionadas pelo Gerenciador L7SP, são submetidas às regras internas de acordo com os serviços habilitados e logo após são encaminhadas para o *middleware* Multichain. O Multichain se comunica com os demais nós e realiza as operações utilizando seu protocolo proprietário. É possível observar de forma macro no desenho de arquitetura a estrutura interna do Multichain. O core do Multichain é responsável por processar as operações e gerenciar sua base de dados não relacional LevelDB. As interfaces de comunicação são seu protocolo de mensagens proprietário que é encapsulado ao TCP/IP para comunicação entre os nós, e sua API de comunicação JSON-RPC com as aplicações externas e usuários, disponibilizando diversos serviços.

O modelo L7SP foi elaborado para que o sistema continue se apresentando para o usuário

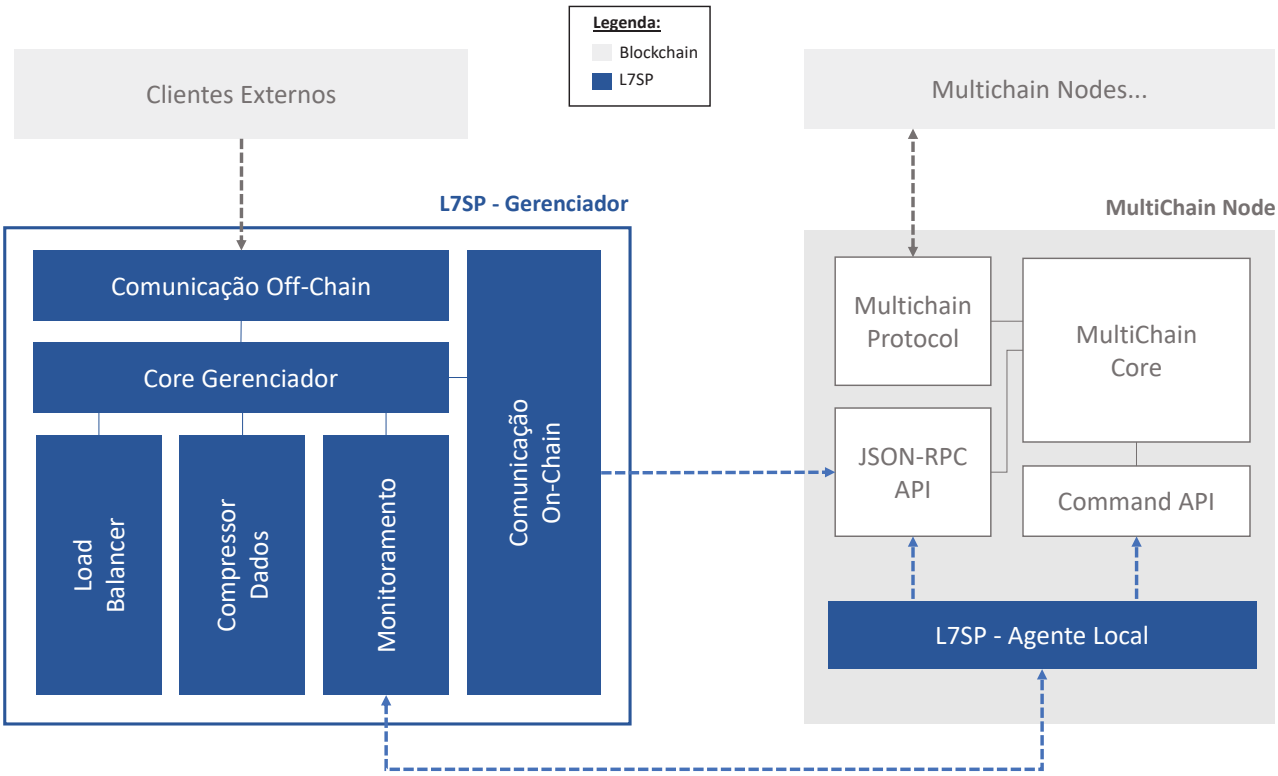

Figura 12 – Arquitetura detalhada da solução
Fonte: elaborado pelo autor

externo como um sistema único e atue internamente na rede blockchain de forma compatível com a topologia e os protocolos disponíveis. L7SP também contribui para a interoperabilidade, não só por acoplar novos componentes de forma transparente, mas também por ser facilmente adaptado a novas topologias, e facilmente customizado para a interligação de diferentes soluções blockchain. Nas seções a seguir são apresentados em detalhe cada componente da solução.

4.2.1 L7SP: Camada de gerenciamento

Conforme já antecipado, L7SP é formado por duas aplicações distintas. A principal delas é a camada intermediária de gerenciamento que é acoplada à plataforma blockchain. Esta aplicação é composta de três módulos que possuem objetivos específicos, possibilitando que diferentes configurações sejam adotadas por meio da ativação destes módulos para ofertar ganho de desempenho no processamento de transações e um gerenciamento mais eficiente do sistema. Projetos futuros podem acrescentar novos módulos nesta camada, afim de interagir com a rede e gerar diferentes tipos de benefícios.

- **Load Balancer:** Este módulo atua de forma autônoma (sem interferência do usuário) recebendo conexões dos clientes externos e dos nós internos da rede blockchain para distribuir a carga de trabalho de maneira uniforme entre os recursos, fazendo com que seja

tirado o melhor proveito do poder computacional disponível na rede. Do ponto de vista das transações realizadas pelos clientes externos, este recurso também provê a capacidade de o sistema se tornar mais tolerante à falhas no caso de indisponibilidade de algum dos nós. Pode ser ativado ou desativado por meio de parametrização.

- **Compressão de dados:** Módulo responsável por realizar a compactação dos dados para otimizar o tráfego de rede e o armazenamento na blockchain. Pode ser ativado ou desativado por meio de parametrização;

- **Monitoramento:** Este módulo verifica regularmente o status de variáveis de ambiente dos nós como CPU e memória, e variáveis de desempenho da rede blockchain. Se comunica com os nós por meio do agente local instalado.

O Gerenciador L7SP escuta uma porta local e repassa as mensagens para o Multichain na porta de destino configurada. Quando uma requisição é recebida, a função principal da aplicação é invocar os métodos respectivos a cada operação da API Multichain, submeter às regras dos serviços ativos que estão implementados nas funções, e por último enviar a requisição para o Multichain. Este fluxo é o core do gerenciador, que é apresentado no pseudocódigo do algoritmo 1. Dentro de cada função específica os dados são tratados de acordo com a configuração do serviço. A aplicação também registra a requisição e resposta no arquivo de log, e retorna para a aplicação cliente. Para cada método da API Multichain que se queira utilizar é necessário replicar este mesmo método dentro do gerenciador. As regras dos serviços disponibilizados pelo gerenciador são aplicadas também conforme o método. No caso do protótipo de L7SP, todos os métodos estão aptos a utilizar o balanceamento de carga, mas somente o método Publish_Stream que publica os dados no *ledger* da blockchain foi implementado para usufruir da compressão, pois, tem a característica de transportar e armazenar um volume significativo de dados.

Algoritmo 1: Gerenciador

Entrada: local_port, multichain_port

Saída: resp

```
1  início
2      listen(local_port);
3      enquanto (true) faça
4          receive_data_req();
5          log(data_req);
6          payload = set_payload(data_req);
7          id_node, url = lb_define_route();
8          ipaddr = get_ipaddr(id_node);
9          log(id_node, payload);
10         resp = request.post(ipaddr, multichain_port, payload);
11         log(resp);
12     fim
13 fim
```

4.2.2 L7SP: Agente local

Algumas ações só podem ser realizadas localmente pelo administrador do sistema via API de comando, por este motivo, um mecanismo eficaz proposto neste trabalho é a implementação de um agente local para compor o modelo L7SP. O agente funciona de forma contínua e autônoma em cada nó da rede realizando a interface com o gerenciador, sendo um componente fundamental no escopo de monitorar o desempenho de cada nó de forma centralizada. O agente local exerce um papel de sensor, monitorando regularmente os recursos da máquina (CPU e memória) e variáveis de desempenho de rede como *throughput* e tempo de resposta. As informações são enviadas para o gerenciador regularmente, respeitando um intervalo de tempo pré-configurado.

4.2.3 Clientes externos

Chamamos de clientes externos as aplicações que não fazem parte da rede interna da blockchain, mas interagem com o sistema no dia a dia para realizar operações. São chamados de clientes porque iniciam a comunicação com a blockchain e realizam solicitações de serviço por meio da API disponibilizada pelo *middleware* blockchain, sejam elas consultivas ou transacionais. Em criptomoedas como o Bitcoin, um exemplo clássico destas aplicações são os sistemas das *exchanges*, que são utilizados para compra, venda e transferência de ativos entre os usuários. Além do contexto das criptomoedas, estas aplicações podem ser Web ou Desktop, ou até dispositivos físicos que se conectam à blockchain para realizar operações de forma segura e

descentralizada na plataforma para os mais diversos tipos de negócio, como por exemplo, suprimentos e logísticas, *health care*, telemedicina, entre outros. Independente da natureza da aplicação, a interface com a plataforma blockchain se dá da mesma forma pelas API's disponíveis. Para simular de forma genérica o comportamento dos clientes externos, foi construída uma aplicação que se conecta ao gerenciador e aos nós para realizar a injeção de transações e requisições de serviços da API Multichain. Esta aplicação é chamada de injetor e viabiliza a execução dos cenários de avaliação de modelo, gerando uma alta carga de processamento para a rede. O protótipo do injetor possui parametrização para o número de *threads* que serão utilizadas e também para o tempo máximo de execução da aplicação, estas estão disponíveis no arquivo de configuração, além do apontamento para o servidor Multichain. Quando a aplicação inicia é cronometrado o tempo de execução e são iniciadas as *threads* conforme quantidade parametrizada pelo usuário, conforme podemos ver no pseudo código apresentado no algoritmo 2. A função principal aguarda o término de todas as *threads* para prosseguir com as próximas instruções.

Algoritmo 2: Injetor - Função principal

```
Entrada: num_threads
Saída: average_resp_time, execution_time
1  início
2      start_time = time();
3      para (i = 0, i < num_threads, i++) faça
4          th_id[i] = create_thread(multichain_injector, i, num_threads);
5      fim
6      para (i = 0, i < num_threads, i++) faça
7          join(th_id[i], num_threads);
8      fim
9      end_time = time();
10     execution_time = end_time - start_time;
11     total_resp_time = 0;
12     para (i = 0, i < len(resp_time), i++) faça
13         total_resp_time += resp_time[i];
14     fim
15     average_resp_time = total_resp_time  count_trn;
16     return average_resp_time, execution_time;
17 fim
```

Dentro de cada *thread* é definido o *payload* da requisição realizada à blockchain, é enviada a solicitação, registrado em arquivo de log as mensagens trocadas e é verificado se o tempo atual de execução não excedeu o limite configurado. O código da *thread* de injeção é apresentado no pseudocódigo do algoritmo 3, ele se refere a função multichain_injector invocada na

função principal. Quando o tempo atual excede o tempo limite configurado, o laço é terminado e consequentemente a *thread* também. Após o término das *threads*, a função principal calcula e informa o tempo total de execução da aplicação, a quantidade total de transações realizadas e o tempo médio de resposta.

Algoritmo 3: Injetor - Thread injetora

```
   Entrada:
   Saída: resp_time, count_trn
1  início
2      req = set_payload();
3      current_time = time();
4      i = 0;
5      enquanto ((current_time − start_time) < max_time) faça
6          registra_log(req);
7          t_start = time();
8          resp = request.post(payload);
9          t_end = time();
10         registra_log(resp);
11         count_trn += 1;
12         resp_time.append([t_end - t_start]);
13         i+=1;
14         current_time = time();
15     fim
16     return resp_time, count_trn;
17 fim
```

4.3 Introduzindo uma camada de empilhamento de serviços para melhorar o desempenho e gerenciamento da Blockchain Privada

Uma vez constituída a arquitetura e definido o papel de cada um de seus componentes, foram especificadas as regras a serem implementadas em cada um dos módulos da camada gerenciadora que é o *core* da solução. Estes serviços são os fatores principais que contribuem para que o objetivo estabelecido na proposta seja alcançado. É importante avaliar que a introdução de uma camada adicional em qualquer solução pode resultar em problemas de desempenho devido ao *overhead* de processamento das operações e maior latência de rede dependendo de como se dá a comunicação entre os componentes. Além disso ainda pode se criar mais pontos de falha que podem levar a indisponibilidade. Para minimizar estes aspectos, alguns requisitos não funcionais foram definidos para a solução:

- Transparência no acoplamento, apenas mudando o apontamento na aplicação cliente;

- Cuidados na implementação do protótipo para que a tecnologia usada e código gerado tenham bom desempenho e não tenham bugs que impactem na disponibilidade;
- Definição de serviços que agreguem desempenho e compensem o *overhead* adicionado pela camada intermediária;
- Evitar a distância geográfica entre os componentes da solução para diminuir a latência de rede.

Há 2 objetivos principais na proposta: o primeiro é monitorar o desempenho do sistema e ter condições de tomar ações rápidas em qualquer parte do ambiente; o segundo é aumentar o desempenho da blockchain privada. Mais especificamente, para aumentar o desempenho é preciso diminuir a latência (quantidade de tempo até a transação ser confirmada) e aumentar o *throughput* (vazão, quantidade de transações por unidade de tempo). Para isto foi criada a camada intermediária de gerenciamento que provê os seguintes mecanismos:

4.3.1 Balanceamento de carga

Com o decorrer do tempo, fatores como o aumento do poder de processamento, avanços na tecnologia de comunicação de dados e a necessidade de compartilhamento de recursos contribuíram para o surgimento dos sistemas computacionais distribuídos. Os objetivos de escalonar processos em sistemas desta natureza são minimizar o tempo de execução e atrasos na comunicação e maximizar o *throughput* do sistema e a utilização de recursos (BRANCO, 2004). Considerando estes objetivos, uma proposta de escalonamento pode ser avaliada quanto aos aspectos de desempenho e eficiência. Se o foco da avaliação é o tempo gasto na execução das políticas de escalonamento, quanto menor o seu valor, melhor o desempenho do escalonador. Se a análise contempla as políticas adotadas, o foco é na eficiência do escalonador (TSENG L.-Y.; CHIN, 2009). Vários autores já sugeriram taxonomias afim de classificar os algoritmos de escalonamento existentes por meio de suas características. Dentre elas, devido a sua aceitação e abrangência, destaca-se a de Casavant (CASAVANT T. L; KUHL, 1998), que á apresentada na figura 13 . Primeiramente esta taxonomia divide o escalonamento em local e global. O escalonamento local determina como um processo em uma única CPU é alocado e executado. Por outro lado, o escalonamento global usa informações do sistema para alocar processos em vários processadores, com o objetivo de otimizar seu desempenho como um todo. Este é o tipo de escalonamento normalmente utilizado em aplicações distribuídas e que será contemplado neste estudo. O escalonamento global é dividido em duas abordagens que dizem respeito ao momento no qual as decisões sobre o escalonamento são tomadas, são elas: estática e dinâmica. No escalonamento estático, as decisões são tomadas antes da aplicação ser iniciada e permanecem constantes durante a execução do programa. Esta estratégia é de simples implementação e é vantajosa quando são conhecidas previamente as informações sobre o ambiente e as características das tarefas a serem processadas. Por outro lado, em muitas aplicações não é possível

prever o comportamento do sistema e a distribuição da carga de trabalho antes da execução do programa. Esta estratégia pode não ser eficiente em casos que há uma maior variação das características dos recursos computacionais e da rede de comunicação em tempo de execução do programa (LU K.; SUBRATA, 2004a). Já o escalonamento dinâmico trabalha com o conceito que pouco ou nada se conhece previamente sobre o comportamento da aplicação e seu ambiente de execução (LU K.; SUBRATA, 2004a; WANG X.; ZHU, 2007; LU K.; SUBRATA, 2004b). Devido à falta destas informações prévias, as decisões sobre o escalonamento são tomadas pelo programa em tempo de execução.

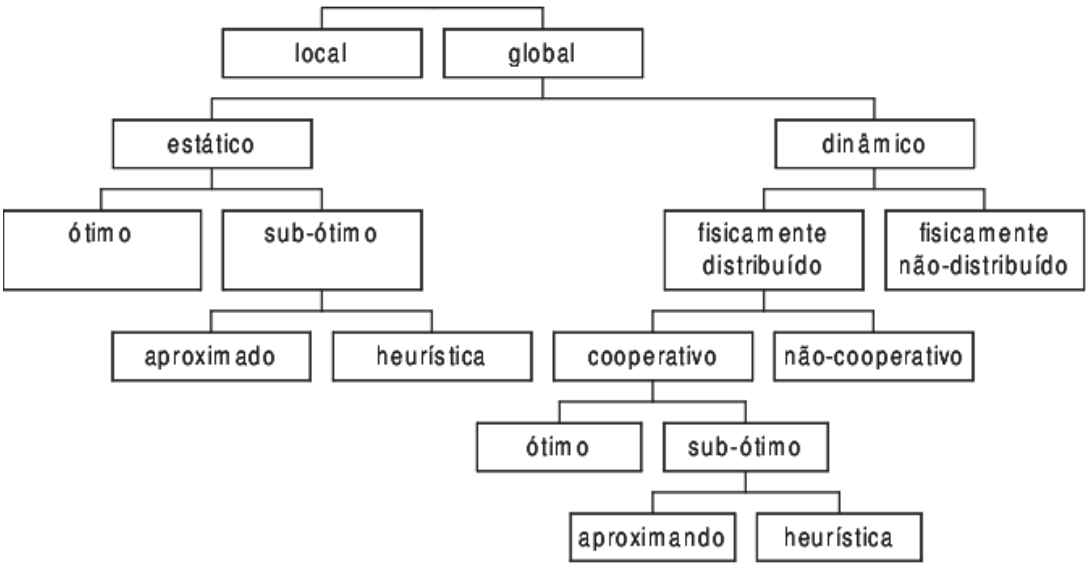

Figura 13 – Taxonomia de Casavant - Algoritmos de escalonamento em sistemas distribuídos
Fonte: CASAVANT T. L; KUHL (1998)

A estratégia de balanceamento proposta em L7SP consiste em formar um *cluster* com grupos de nós que irão processar os comandos de API que interagem com a blockchain, onde todos os nós estão conectados à camada de gerenciamento proposta. Os comandos podem ser direcionados para qualquer um dos nós no cluster, retornando o mesmo resultado para os mesmos parâmetros de entrada. O objetivo é distribuir a carga de trabalho uniformemente entre os nós da blockchain, otimizando a utilização dos recursos e maximizando o desempenho (menor tempo de resposta). Esta estratégia também aumenta a disponibilidade do sistema, uma vez que clientes externos conectados a um nó específico poderão ser direcionados para outro nó qualquer em caso de falha. Para a adoção deste mecanismo é preciso que algumas premissas sejam satisfeitas, os nós devem estar em sincronia com a blockchain, possuir o mesmo conjunto de endereços de carteira, ser elegíveis para o mesmo conjunto de recursos e fluxos e ter o mesmo valor para os parâmetros globais de configuração. Com base nas características dos algoritmos de escalonamento globais estáticos e dinâmicos, foram aplicados dois diferentes mecanismos

de balanceamento para avaliação desta estratégia: o *Round Robin* e *Least Connections*. A definição de rota para o encaminhamento das requisições dos nós Multichain é demonstrada no algoritmo 4. Primeiro é verificado se o balanceamento de carga está ativo, e uma vez que está habilitado, é verificado qual das 2 estratégias de balanceamento devem ser aplicadas.

Algoritmo 4: Load Balancer - Definição de Rota

```
   Entrada: LB, LB_Type
   Saída: node_id, ip_addr
1  início
2      se (LB == ON) então
3          se (LB_Type == ROUND_ROBIN) então
4              se ( (rr_queue + 1) == max_lb_nodes) então
5                  rr_queue = 0;
6              fim
7              senão
8                  rr_queue += 1;
9                  node_id = node_id_list[rr_queue];
10             fim
11             ip_addr = node_ipaddr_list[rr_queue];
12         fim
13         se (LB_Type == LEAST_CONNECTION) então
14             node_id = node_connections.index(min(node_connections));
15             ip_addr = node_ipaddr_list[node_id];
16             node_connections[node_id] += 1;
17         fim
18         senão
19             rota_default(node_id, url);
20         fim
21     fim
22     senão
23         rota_default(node_id, url);
24     fim
25     return node_id, ip_addr;
26 fim
```

- **Round Robin**: o *Round Roubin* é um algoritmo de escalonamento amplamente utilizado em soluções de balanceamento de carga e distribuição de tarefas em um sistema operacional. Sua implementação é simples, distribuindo as tarefas em partes iguais em uma fila circular, sem haver qualquer prioridade entre os nós, apenas obedecendo a ordem estabelecida previamente. Este comportamento é apresentado na figura 14. Devido as

estas características, as literaturas consideram *Round Robin* como uma técnica de balanceamento estático, pois distribui as requisições baseado somente em qual é o próximo recurso de determinada sequência em que a requisição deve ser encaminhada.

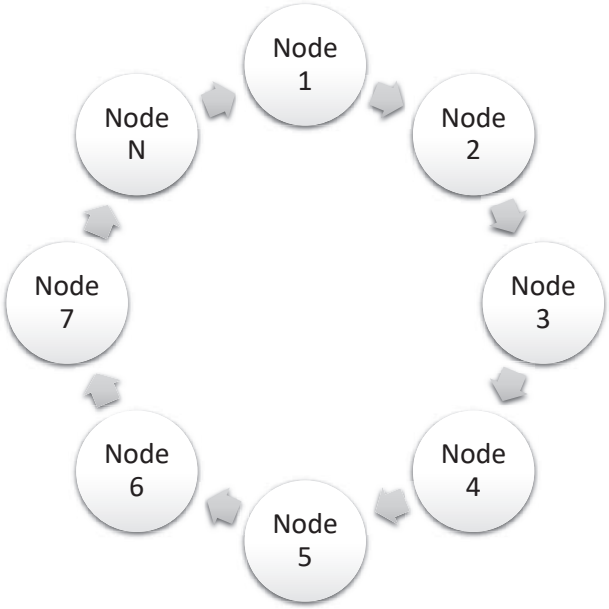

Figura 14 – Balanceamento de Carga - Round Robin
Fonte: elaborado pelo autor

- **Least Connections**: Esta é uma estratégia dinâmica de balanceamento, pois a decisão de para qual nó será roteada a requisição é tomada em tempo de execução do programa, baseada em informações coletadas da aplicação ou do ambiente. Neste caso o objetivo é verificar em cada requisição qual é o nó que possui menos canais de comunicação estabelecidos com o gerenciador, e encaminhar o pacote para o mesmo processar. O conceito neste caso é que o nó com menor número de conexões consequentemente possui menor carga de trabalho.

Tabela 6 – Balanceamanto de Carga - Least Connections

ID Node	Qtde Conexões	Prioridade
1	5	4
2	4	2
3	3	1
4	4	3

Fonte: elaborado pelo autor

4.3.2 Compressão de dados

O objetivo da funcionalidade de compressão de dados é reduzir o espaço ocupado por dados no dispositivo de armazenamento. Essa operação é realizada através do algoritmo de compressão com o intuito de reduzir a quantidade de Bytes para representar um dado de *stream* armazenado na blockchain. Além de economizar espaço no dispositivo de armazenamento, a compressão visa ganhar desempenho nas transmissões, diminuindo o tempo em que elas ocorrem na rede. Um método de compressão é classificado como sem perdas (em inglês, *lossless*) se os dados obtidos após a compressão são idênticos aos dados originais. Quando os dados obtidos após a compressão não são idênticos aos originais, pois "perderam"as informações irrelevantes, dizemos então que é um método de compressão com perdas (em inglês, *lossy*). No caso dos dados armazenados na blockchain, para fins de avaliação foi utilizada uma compressão sem perdas, considerando dados de texto que são obtidos diretamente por meios digitais, onde uma pequena perda de dados acarreta em um não funcionamento do sistema ou torna os dados incompreensíveis. Algumas imagens e sons também precisam ser reproduzidos de forma exata, como imagens e gravações para perícias, impressões digitais, etc. Os métodos de compressão podem ser classificados ainda quanto a simetria e a adaptabilidade. Quando a compressão e a descompressão são feitas executando-se métodos ou algoritmos idênticos ou bem semelhantes, dizemos que o método de compressão é simétrico. Esta estratégia é mais comum e útil quando vamos comprimir e descomprimir várias vezes, os métodos possuem a mesma complexidade. O algoritmo 5 apresenta a lógica para a compressão de dados em L7SP. Primeiro é verificado pelo gerenciador se a compressão de dados está habilitada, logo os dados são recuperados e submetidos à compressão, substituindo a *stream* original pelos dados compactados.

Algoritmo 5: Função que trata a compressão e retorna o payload

```
   Entrada: req
   Saída: new_req
1  início
2      se ((compression == ON)&&(method == COMPRESSED)) então
3          data = recover_data_req() compressed_stream = bz2.compress(data.stream);
4          new_req = data;
5          new_req["params"][2] = compressed_stream;
6      fim
7      senão
8          new_req = data;
9      fim
10     return new_req;
11 fim
```

4.3.3 Monitoramento:

O monitoramento provido por L7SP tem como primeiro objetivo comprovar a eficiência da implementação do modelo, fornecendo informações sobre o desempenho do sistema e os recursos dos nós da rede Multichain. O monitoramento *online* ofertado por esta camada é uma forma efetiva de garantir que os resultados esperados estão sendo atingidos, pois é possível saber o quanto o sistema está sendo usado e como está o seu desempenho. Desta forma, um objetivo mais amplo deste módulo é gerar uma visão do estado global do sistema em tempo real, considerando o monitoramento de métricas relevantes em períodos de tempo pré-estabelecidos. Isto se aplica tanto para os cenários de testes de avaliação do modelo quanto para o acompanhamento da operação do dia a dia. Neste contexto, L7SP introduz o conceito de BKPI's, que são indicadores chave de desempenho para blockchain. BKPI's são sugeridos neste estudo como métricas quantitativas elegidas pelo usuário como essenciais para avaliar a blockchain, conectando estes indicadores diretamente aos efeitos que podem gerar no negócio que é suportado pela solução. São indicadores que o gestor do sistema define para acompanhar a evolução da operação, mantendo assim o foco nas metas estabelecidas. Os BKPI's são úteis para embasar decisões como programar manutenções preventivas, realizar ações de melhoria na aplicação e investimentos de infraestrutura, buscando sempre prover a melhor experiência para o usuário final. Para fins de avaliação, no protótipo L7SP foram elegidos os seguintes BKPI's:

- Tempo médio de resposta
- Throughput
- Taxa de Erro

Estes indicadores contribuem para que a gestão do ambiente deixe de ser reativa e passe a ser proativa. Além disso, ainda por meio da análise dos dados históricos dos BKPI's, o gestor do sistema pode criar modelos de previsão, tendências de capacidade e correlação de alertas mais precisos. Nesta mesma linha, o monitoramento dos recursos de *hardware* dos nós blockchain é realizado pela ação do Agente Local do protótipo L7SP junto ao Gerenciador. O alto consumo de CPU e memória de forma desequilibrada nestes servidores pode impactar no desempenho da solução blockchain, levando também a impactar indiretamente no resultado dos BKPI's. No capítulo 5 são abordados os detalhes deste monitoramento e de como os dados serão coletados e tratados para realizar a análise dos resultados.

5 METODOLOGIA DE AVALIAÇÃO

Este capítulo apresenta a metodologia utilizada para avaliar o modelo L7SP, contemplando os atributos do protótipo desenvolvido e a infraestrutura utilizada. Na seção 5.1 são abordadas as etapas do desenvolvimento e na seção 5.2 são apresentados os aspectos de implementação e a infraestrutura que suporta a solução. A seção 5.3 descreve os cenários de testes estabelecidos para avaliação do modelo. Na seção 5.4 são abordadas as ferramentas e as métricas de desempenho utilizadas para validar os resultados coletados nos testes. Na seção 5.5 são apresentados detalhes sobre o monitoramento de recursos e coleta de dados nos nós. Por fim, a seção 5.6 aborda os conceitos de *Streams*, que serão alvo das simulações realizadas na blockchain neste estudo.

5.1 Etapas de Desenvolvimento

As etapas de desenvolvimento são apresentadas na figura 15. Inicialmente foi realizado a instalação e configuração do *middleware* Multichain em uma VM Linux na nuvem da Amazon. Feita a instalação da plataforma blockchain, foram realizados os primeiros testes por meio da interface de comando e API JSON-RPC para controlar a execução do nó blockchain, criação de *streams*, consultas de status, transações, entre outros. Uma vez tendo uma imagem de um nó Multichain, foi realizado um *snapshot* para criar novas instâncias correspondendo a novos nós na rede blockchain. Com a rede blockchain em operação foi realizada a implementação do injetor de mensagens JSON-RPC, que representa os clientes/aplicações externas que se conectam à plataforma blockchain para realizar operações. O injetor pode se conectar diretamente a um nó da rede blockchain ou ao gerenciador do L7SP, apenas alterando o apontamento da conexão. Logo após foi realizada a implementação do protótipo do gerenciador, que recebe as solicitações do injetor, aplica as regras configuradas e comunica com a plataforma blockchain. Com o gerenciador em funcionamento, foi realizada a implementação do agente local para rodar em cada um dos nós blockchain para coletar informações dos recursos, gerando as informações para o monitoramento. A implementação foi sucedida com a realização de vários cenários de testes de desempenho, onde foram coletados os resultados por meio das funcionalidades implementadas no Agente Local. Após a análise dos resultados iniciais, foram realizados ajustes no protótipo e na configuração do ambiente. Por último, foram realizados novos testes e coletados os resultados finais.

5.2 Implementação e Infraestrutura

O protótipo do modelo L7SP contemplando a camada de gerenciamento, o injetor e o agente local foram desenvolvidos utilizando a linguagem Python, que oferece algumas vantagens, além de ser multiplataforma, é uma linguagem de fácil manutenção. Seu código é interpretado, por-

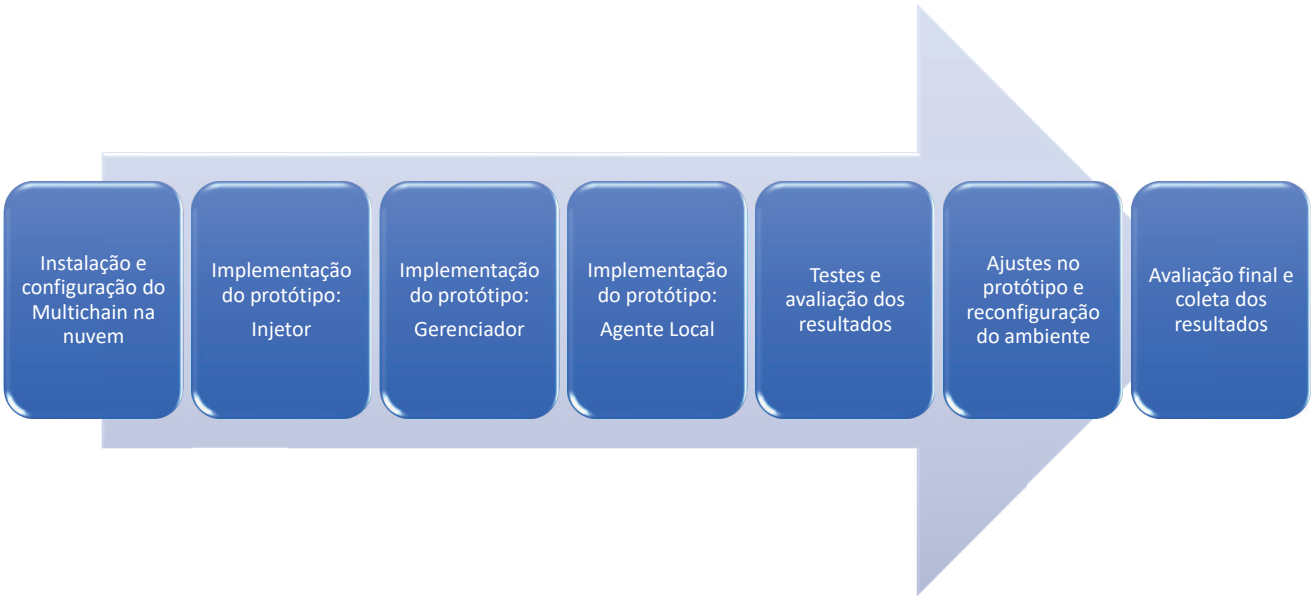

Figura 15 – Etapas de desenvolvimento do protótipo
Fonte: elaborado pelo autor

tanto, não trabalha com compilação, ofertando um ambiente menos complexo. Estas características colaboraram para que houvesse um foco maior no desenvolvimento das regras e solução do problema. Algumas bibliotecas Python foram selecionadas e utilizadas em todas as aplicações do protótipo para facilitar a parametrização e avaliação do modelo, são elas:

- logging: log em arquivo texto
- configparser: interface com arquivos de configuração
- time: controles de tempo dentro da aplicação

As aplicações desenvolvidas para o protótipo e o blockchain foram executadas em máquinas virtuais na nuvem privada da AWS, utilizando o serviço EC2. Os servidores utilizados possuem a seguinte configuração:

- Ubuntu Server 16.04 LTS (HVM)
- 1GB Memória RAM
- 7GB Armazenamento
- t2.micro (1 vCPUs, 2.5 GHz, Intel Xeon Family)

5.2.1 Gerenciador

O gerenciador funciona como um nó da rede blockchain e é executado em uma VM Linux na nuvem, disponibilizada pelo serviço EC2 da AWS. No código Python do Gerenciador são

utilizadas as bibliotecas werkzeug, json e requests para disponibilizar o servidor JSON/RPC e realizar as requisições para o Multichain. Por não ser originalmente *multithread*, foi realizada uma alteração dentro da biblioteca werkzeug para permitir que cada solicitação fosse processada em paralelo a partir da criação de uma nova *thread*. A porta TCP/IP padrão da API Multichain é a 7212 que está configurada como porta de destino. O Gerenciador L7SP escuta localmente a porta 7211 e repassa as mensagens para o Multichain na porta 7212. Esta estratégia de utilizar portas distintas permite inclusive que o Gerenciador possa rodar no mesmo ambiente de um nó Multichain, aproveitando a infraestrutura existente. O protótipo do gerenciador disponibiliza uma estrutura de configuração, que contempla os dados de comunicação com os servidores Multichain, usuário e senha utilizados nas chamadas da API, entre outras funcionalidades da aplicação. Os algoritmos de balanceamento de carga foram implementados na linguagem Python obedecendo quase que identicamente a lógica prevista no pseudocódigo apresentado no modelo. A compactação de dados é realizada através do algoritmo de compressão bz2 disponibilizado em uma biblioteca de mesmo nome, com o intuito de reduzir a quantidade de Bytes para representar um dado de *stream* armazenado na blockchain. A função BZIP2 utilizada no protótipo comprime dados usando o algoritmo de compressão de texto de classificação de bloco BurrowsWheeler e a codificação Huffman. A compressão utilizando este método geralmente é consideravelmente melhor do que a obtida pelos compressores convencionais baseados em LZ77 / LZ78, e se aproxima do desempenho da família de compressores estatísticos PPM sendo muito mais rápida. BZIP2 é um software de código aberto disponível gratuitamente no mundo Unix. Ele é usado para compactar distribuições de software e código-fonte, como o kernel do Linux, entre outras coisas.

5.2.2 Injetor

O injetor pode ser executado a partir de qualquer máquina que tenha acesso via rede ao Gerenciador e aos nós blockchain. O injetor utiliza as bibliotecas comuns já citadas e também as bibliotecas json e request para a implementação das chamadas JSON-RPC da API Multichain. Além disso, também foi utilizada a biblioteca threading para a realização de múltiplas requisições em paralelo conforme configuração pré-estabelecida e também a bilioteca numpy para a utilização e conversão de vetores de ponto flutuante.

5.2.3 Agente Local

Para a implementação do Agente local foram utilizadas as bibliotecas os e sys para viabilizar a execução de comandos do sistema operacional Linux para recuperar informações sobre o status dos recursos do ambiente. A biblioteca time foi utilizada para controlar os tempos de execução da aplicação e também para estabelecer intervalos entre a execução de cada *thread*. A biblioteca multiprocessing permitiu que cada comando de monitoramento fosse realizada em

um processo separado de forma paralela, armazenando o resultado da mesma forma em arquivo texto.

5.2.4 Middleware Blockchain

Para compor a arquitetura proposta, o *middleware* blockchain escolhido foi o Multichain (MULTICHAIN - OPEN PLATFORM FOR BUILDING BLOCKCHAINS, 2018). O MultiChain é uma plataforma para a criação e implementação de blockchains privadas, dentro de organizações ou entre elas. Destina-se a superar um obstáculo fundamental para a implantação da tecnologia blockchain no setor financeiro institucional, proporcionando a privacidade e controle necessários em um pacote de fácil utilização. O Multichain é derivado do software core do Bitcoin, e a exemplo do seu predecessor suporta servidores Windows, Linux e Mac, fornece uma API simples e uma interface de linha de comando. Uma das facilidades do MultiChain é que ele permite que o usuário defina todos os parâmetros da blockchain em um arquivo de configuração, incluindo:

- O protocolo da cadeia, blockchain privada ou bitcoin puro;
- Tempo alvo para criação de blocos, por exemplo 1 minuto;
- Tipos de permissão ativos, por exemplo, qualquer um pode se conectar, apenas alguns podem enviar/receber;
- Diversidade de mineração (somente blockchains privadas);
- Nível de consenso necessário para criar/remover; administradores e mineradores;
- Recompensas de mineração, por exemplo 50 unidades de moeda nativa por bloco, reduzindo para metade a cada 210.000 blocos;
- Portas IP para conexões P2P e API JSON-RPC, por exemplo, 8571 e 8570;
- Tipos de transação permitidos, por exemplo, pay-to-address, pay-to-multisig, pay-to-script-hash;
- Tamanho máximo do bloco, por ex. 1 megabyte;
- Máximo de metadados por transação (OP_RETURN), por exemplo: 4096 bytes.

Uma vez que o MultiChain é baseado em um *fork* do Bitcoin Core, muitos aspectos do seu design visam permitir transições suaves entre blockchains privadas e a blockchain do bitcoin. O Multichain usa a arquitetura blockchain e o protocolo de transação bitcoin, com alterações apenas no processo de *handshaking* quando dois nós se conectam inicialmente. Todos os outros recursos são implementados usando metadados e modificações nas regras de validação de transações e bloqueios. Sua interface (linha de comando e API) é totalmente compatível com a do

Bitcoin Core, inclusive um nó Multichain pode atuar como um nó na rede bitcoin regular (ou outras redes baseadas no bitcoin), através de uma simples configuração de protocolo no arquivo de configuração per-blockchain. Seu design também permite que futuros aprimoramentos na plataforma bitcoin sejam mesclados e adicionados. Com relação ao seu fluxo de dados, uma mensagem é enviada do originador para o destinatário da seguinte maneira:

1. O nó MultiChain de origem envia uma transação de mensagem ao destinatário com metadados contendo seu endereço IP e um *hash* do conteúdo da mensagem.

2. O nó receptor recebe a transação e decodifica esses metadados.

3. O nó receptor entra em contato com o nó de origem por meio de seu endereço IP para recuperar a mensagem, assinando a solicitação para provar sua identidade como o destinatário pretendido. Esta comunicação ocorre usando o protocolo blockchain P2P existente.

4. O nó de recebimento verifica a validade da mensagem, verificando seu hash em relação ao hash incorporado na transação original.

5. Se a mensagem for válida, o nó receptor concluirá o loop enviando uma segunda transação ao remetente que contém o mesmo hash de mensagem.

O Multichain possui seu próprio mecanismo de consenso, que se assemelha ao PBFT, mas em vez de múltiplos validadores por bloco, existe um validador por bloco, trabalhando em um escalonamento do tipo *round robin*. Ao restringir a mineração a um conjunto de entidades identificáveis, o MultiChain resolve o dilema colocado pelas blockchains privadas, nas quais um participante pode monopolizar o processo de mineração. A solução está em uma restrição no número de blocos que podem ser criados pelo mesmo minerador dentro de uma determinada janela. O MultiChain implementa esse esquema usando um parâmetro chamado diversidade de mineração, que é restrito por 0 diversidade de mineração 1. A validade de um bloco é verificada da seguinte forma:

1. Aplicar todas as alterações de permissões definidas pelas transações no bloco em ordem.

2. Contar o número de nós validadores permitidos que são definidos após a aplicação dessas mudanças.

3. Multiplicar o número de nós validadores pela diversidade de mineração, arredondando para obter espaçamento.

4. Se o minerador deste bloco extraiu um dos blocos anteriores de espaçamento -1, o bloco é inválido.

Isso impõe um cronograma de *round robin*, no qual os mineradores permitidos devem criar blocos em rotação para gerar uma blockchain válida. O parâmetro de diversidade de mineração

define o rigor do esquema, ou seja, a proporção de mineiros autorizados que precisariam se conspirar para enfraquecer a rede. Apesar da escolha do Multichain para a realização da prova de conceito, o modelo apresentado neste estudo é genérico e pode ser aplicado a qualquer outro middleware blockchain. Para isto, o protótipo deve ser devidamente ajustado para poder se integrar com diferentes API's.

5.3 Cenários de Avaliação

Para validar os resultados da implantação do modelo foi utilizada uma abordagem quantitativa, gerando informações numéricas sobre o comportamento do sistema. Os dados foram coletados a partir dos ambientes de execução das aplicações por meio de comandos do sistema operacional, comandos da API Multichain e tempos cronometrados pelas aplicações desenvolvidas para o próprio protótipo. Medidas precisas são importantes para avaliar o reflexo da implantação do modelo no sistema e embasar decisões mais acertadas sobre ajustes necessários ou projetos futuros. Outro fator importante é que uma vez realizada a coleta, ferramentas estatísticas podem ser aplicadas a estes dados numéricos para tentar prever comportamentos em diferentes cenários ou períodos de tempo. A avaliação foi realizada por meio de um teste de carga explorando diferentes configurações do ambiente e do modelo. Abaixo seguem os cenários que foram estipulados para a avaliação:

- **Cenário 1:**

  Execução do teste de carga no ambiente blockchain puro, realizando a injeção das requisições diretamente em um nó Multichain;

- **Cenário 2:**

  Execução do teste de carga no protótipo L7SP com a compressão de dados ativa, redirecionando as requisições para apenas um nó Multichain;

- **Cenário 3:**

  Execução do teste de carga no protótipo L7SP com o balanceamento de carga ativo, utilizando a estratégia *Round Robin*. As requisições são direcionadas para vários nós Multichain, de acordo com a configuração do pool de servidores;

- **Cenário 4:**

  Execução do teste de carga no protótipo L7SP com o balanceamento de carga ativo, utilizando a estratégia *Least Connections*. As requisições são direcionadas para vários nós Multichain, de acordo com a configuração do pool de servidores;

- **Cenário 5:**

Execução do teste de carga no protótipo L7SP com a compressão de dados e o balanceamento de carga ativos. As técnicas de balanceamento podem ser variadas para obter o melhor desempenho.

- **Cenário 6:**

  Repetição do cenário 5, aumentando a carga de entrada até o máximo suportado;

- **Cenário 7:**

  Execução do teste de carga no Multichain em 2 etapas: (a) sem compressão e (b) com compressão em L7SP. Avaliar a diferença de tamanho no armazenamento de dados na blockchain.

5.4 Métricas de avaliação

Foram realizadas medições em todos os cenários e posterior comparação, baseado nas métricas de desempenho de blockchain, conforme apresentado abaixo:

- **Latência**: Métrica para medir o tempo que uma transação é validada pelo sistema, é a diferença de tempo entre o momento da requisição até o momento da confirmação pela blockchain:

$$L = t - t_0$$

  onde *L* é latência, t_0 é o *timestamp* da hora que o cliente requisitou uma transação e o *t* é o *timestamp* da confirmação da transação (CROMAN et al., 2016; KAKAVAND; Kost De Sevres; CHILTON, 2017; DINH et al., 2017; PONGNUMKUL; SIRIPANPORNCHANA; THAJCHAYAPONG, 2017).

- **Latência Média**: Métrica para medir a média de tempo de que as transações são validadas pelo sistema. É o somatório da latência de cada transação dividido pelo número total de transações validadas:

$$LM = \frac{\sum_{i=1}^{n} L_i}{Trn}$$

  onde *LM* é a Latência Média, *L* é a latência de cada transação e o *Trn* é o total de transações realizadas pelo sistema (KAKAVAND; Kost De Sevres; CHILTON, 2017; PONGNUMKUL; SIRIPANPORNCHANA; THAJCHAYAPONG, 2017).

- **Bootstrapp Time**: Métrica para medir o tempo que um novo nó leva para realizar o *download* da blockchain e estar disponível para participar ativamente da rede validando transações:

$$BT = t - t_0$$

- onde *BT* é *Bootstrapp Time*, t_0 é o *timestamp* do início da alocação do nó, e *t* é o tempo que o novo nó fica ativo na rede (CROMAN et al., 2016).

- **Bootstrapp Time Médio**: Métrica para medir a média de tempo que os nós são alocados na rede blockchain. É o somatório do Bootstrapp Time de cada alocação dividido pelo número de máquinas alocadas:

$$BTM = \frac{\sum_{i=1}^{n} BT_i}{Nm}$$

  onde *BTM* é a Bootstrap Time Médio, *BT* é a Bootstrap Time de cada alocação e o *Nm* é o número de máquina alocadas (CROMAN et al., 2016).

- **Throuhput**: Métrica para medir a taxa em que a blockchain confirma as transações. É a quantidade de transações confirmadas pela blockchain dividido pela unidade de tempo, ou seja, transações por segundo (TPS). Está diretamente relacionado ao tamanho e intervalo de bloco utilizados:

$$TH = \frac{Trn}{t}$$

  onde *TH* é o Throughput, *Trn* é a quantidade de transações e *t* é a unidade de tempo (CROMAN et al., 2016; KAKAVAND; Kost De Sevres; CHILTON, 2017; DINH et al., 2017; PONGNUMKUL; SIRIPANPORNCHANA; THAJCHAYAPONG, 2017).

- **Throuhput Médio**: Métrica para medir a taxa média em que a blockchain confirma as transações. É a soma da quantidade total de transações confirmadas pela blockchain dividido pelo período de tempo:

$$THM = \frac{\sum_{i=1}^{n} Trn_i}{\Delta t}$$

  onde *THM* é o Throughput Médio, *Trn* é a quantidade de transações e Δt é o período de tempo. (KAKAVAND; Kost De Sevres; CHILTON, 2017; PONGNUMKUL; SIRIPANPORNCHANA; THAJCHAYAPONG, 2017).

Em complemento às métricas utilizadas nos trabalhos relacionados, este estudo sugere três novas métricas para serem utilizadas na avaliação do modelo L7SP e em futuras avaliações envolvendo blockchain, afim de facilitar a comparação de diferentes cenários e a interpretação de resultados. São elas:

- **Ganho de desempenho**: Métrica para medir o ganho de desempenho obtido na blockchain baseado na variação do *throughput*, comparando 2 configurações distintas, uma configuração de referência e outra que se deseja avaliar. O valor resultante é em percentual, e se dá pela diferença dos *throughputs* mensurados multiplicados por 100, dividido pelo *throughput* de referência:

$$GTh = \frac{(Th\_New - Th\_Ref) * 100}{Th\_Ref}$$

  onde *G* é o Ganho de desempenho, *Th_Ref* é a primeira leitura de throughput sendo o valor de referência, e *Th_New* é a segunda leitura de throughput, sendo o valor que se deseja avaliar.

- **Taxa de erro**: Métrica para medir a quantidade de erros ocorridos nas requisições submetidas à blockchain. Nos testes iniciais com o Multichain, verificou-se que quando a carga é elevada a níveis muito altos, ocorrem perdas de transações devido a *timeouts* e erros de resposta. Para fins de avaliação de uma solução, isto deve ser considerado, estipulando qual é a taxa de erro aceitável para um determinado cenário ser válido. A taxa de erro é um percentual que se dá pela diferença entre a quantidade total de transações e a quantidade de erros multiplicado por 100, tudo isto dividido pela quantidade total de transações:

$$Tx\_Err = \frac{(Total\_Trn - Trn\_Err) * 100}{Total\_Trn}$$

  onde *Tx_Err* é a taxa de erro, *Total_Trn* é a quantidade total de transações, e *Trn_Err* é a quantidade de transações com erro.

- **Quantidade de Erros**: Uma vez que se deseja calcular a taxa de erro, é necessário obter a quantidade de erros ocorridos nos cenários de avaliação. A quantidade de erros se dá pela diferença entre a quantidade total de transações e a quantidade total de transações com sucessos:

$$Trn\_Err = Total\_Trn - Trn\_Ok$$

  onde *Trn_Err* é a quantidade de transações com erro, *Total_Trn* é a quantidade total de transações, e *Trn_Ok* é a quantidade de transações com sucesso.

Além das métricas da blockchain, outras variáveis foram monitoradas para compor a verificação do desempenho do sistema:

- Tempo de resposta das requisições da API Multichain;

- Tempo médio de resposta das requisições da API Multichain;
- Consumo dos recursos dos nós (CPU, Memória e Disco)

Os tempos de resposta da API Multichain e do Gerenciador são cronometrados e coletados pelo Injetor.

5.5 Verificação dos recursos dos nós

Por meio do Agente Local, o protótipo realiza o monitoramento de cada nó, provendo uma visão do estado global do ambiente computacional em tempo real, e possibilitando a detecção de falhas com antecedência e um melhor gerenciamento do sistema. Os dados coletados localmente a partir de cada dó permitem gerar um panorama de todo o sistema, e são úteis aos usuários e administradores dos recursos. Estas informações podem ser utilizadas para manutenção preventiva e redimensionamento do sistema, reduzindo acionamentos em caso de urgência e eliminando problemas antes que eles ocorram. Para facilitar a avaliação e gerar maior estabilidade no ambiente, por definição de projeto se optou por utilizar uma rede distribuída homogênea, onde todos os nós possuem a mesma capacidade de recursos. Para verificar o espaço em disco durante a avaliação, será utilizado o comando du. As métricas de consumo de recursos de *hardware* são coletadas por meio do comando top do Linux, o resultado do comando é apresentado na figura 16. Por exemplo, as informações sobre o estado da CPU são apresentadas na terceira linha e as de memória na quarta e quinta linha.

Para realizar a coleta e gerar a saída formato adequado para o tratamento do protótipo, conforme mostrado na figura 17 em um exemplo para a CPU, serão utilizadas as seguintes opções do comando top: (-b) para formatar a saída eliminando caracteres de controle, (-d) para configurar o intervalo entre cada coleta do comando e (-n) para configurar o número de coletas que são realizadas. O comando é combinado por meio com o comando grep para filtrar o conteúdo de saída do comando top, monitorando exatamente as variáveis ou recursos de interesse. Da mesma forma que mostrado neste exemplo para a CPU, o comando nesta sintaxe foi utilizado para o monitoramento dos demais recursos. A seguir é apresentada uma visão mais detalhada das seções do comando top, que será o principal utilitário para o monitoramento local.

5.5.1 CPU

A seção de uso da CPU mostra o percentual de tempo de CPU gasto nas tarefas. O valor us é o tempo que a CPU gasta executando processos no espaço do usuário. Da mesma forma, o valor sy é o tempo gasto na execução de processos de sistema. O valor ni apresenta o tempo gasto na execução de processos com um ajuste manual de prioridade, seja mais alta ou mais baixa. O valor id, é o tempo que a CPU permanece inativa (*idle*). A maioria dos sistemas operacionais coloca a CPU em um modo de economia de energia quando está ociosa. Em seguida, vem o

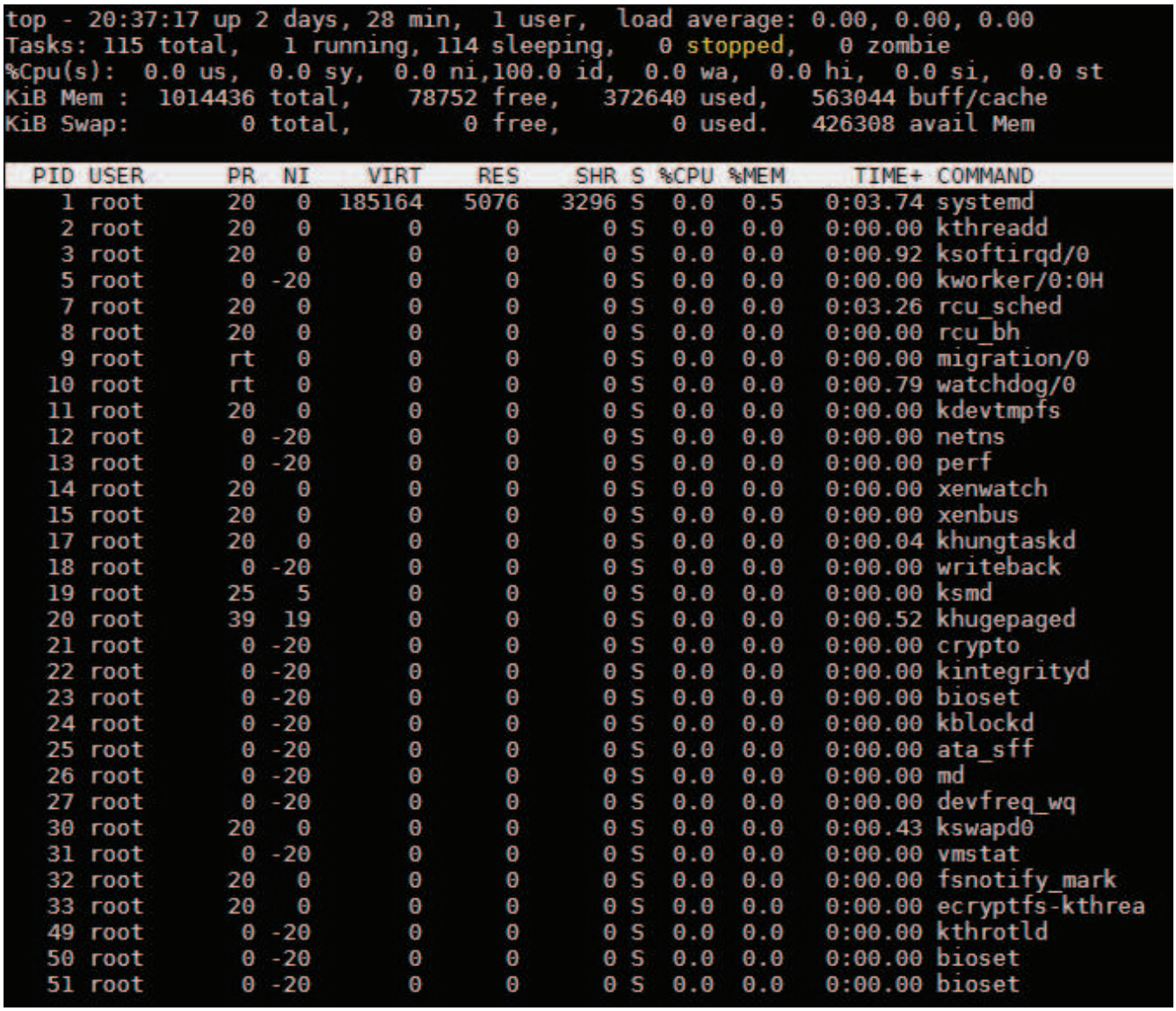

Figura 16 – Resultado do comando top
Fonte: elaborado pelo autor

valor wa, que é o tempo que a CPU gasta aguardando a conclusão de I/O. O tempo gasto no tratamento de interrupções de hardware e software é dado por hi e si, respectivamente. Em um ambiente virtualizado, a quantidade de tempo perdido quando há trabalho a ser feito, mas a CPU não pode executá-lo porque ocupada em outra VM é mostrado como st (*steal*).

5.5.2 Memória

A seção “memória” do comando top mostra informações sobre o uso de memória do sistema. As linhas na figura 16 onde contém “*Mem*” e “*Swap*” mostram informações sobre RAM e espaço de troca, respectivamente. O espaço de troca é uma parte do disco rígido que é usado como RAM. Quando o uso de RAM fica quase cheio, as regiões pouco usadas da RAM são gravadas no espaço de troca, prontas para serem recuperadas posteriormente, quando necessá-

```
ubuntu@ip-172-31-42-89:~$ top -b -d1 -n5 | grep "Cpu"
%Cpu(s):  0.3 us,  0.1 sy,  0.1 ni, 99.4 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu(s):  0.0 us,  1.0 sy,  0.0 ni, 99.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu(s):  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu(s):  1.0 us,  0.0 sy,  0.0 ni, 99.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu(s):  0.0 us,  1.0 sy,  0.0 ni, 99.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
ubuntu@ip-172-31-42-89:~$
```

Figura 17 – Resultado do comando top conforme configurado para coleta de dados no Agente Local
Fonte: elaborado pelo autor

rio. No entanto, como o acesso a disco é mais lento, a utilização de Swap pode prejudicar o desempenho do sistema. Os valores “total”, “livre” e “usado” têm seus significados usuais. O valor “avail mem” é a quantidade de memória que pode ser alocada para processos sem entrar na Swap.

5.5.3 Média de carga

A seção de média de carga (*Load average*) representa a carga média durante um, cinco e quinze minutos respectivamente, fica localizada na primeira linha do resultado do comando bem a direita conforme a figura 16. *"Load"* é a medida da quantidade de trabalho computacional que um sistema executa. No Linux, a carga é o número de processos nos estados R e D a qualquer momento. O valor *"load average"* fornece uma medida relativa de quanto tempo é necessário esperar para que as coisas sejam concluídas. A média de carga é representado em percentual, onde por exemplo 0,1 significa que o sistema está fazendo apenas 10% do que é capaz de suportar e igual ou acima de 1 representa que o sistema está sobrecarregado, estando exatamente ou acima de sua capacidade.

5.5.4 Tarefas

A seção *"tasks"* mostra estatísticas sobre os processos em execução no sistema. O valor total é o número total de processos. A CPU está inativa quando um processo executa I/O, portanto, o SO alterna para a execução de outros processos durante esse tempo. Além disso, o sistema operacional permite que um determinado processo seja executado por um período de tempo muito pequeno e depois passe para outro processo, e isto exige que o “estado” do processo seja controlado.

5.5.5 Estados dos processos

Outro ponto relevante para a interpretação do comando top e um monitoramento do ambiente mais eficiente é entender os estados possíveis para um processo no SO. No Linux, um processo pode estar nos seguintes estados:

- Runnable (R): Um processo neste estado está sendo executado na CPU ou está presente na fila de execução, pronto para ser executado;
- Interruptible sleep (S): Um processo neste estado está aguardando a conclusão de um evento;
- Uninterruptible sleep (D): Nesse caso, um processo está aguardando a conclusão de uma operação de I/O.
- Stopped (T): Estes processos foram parados por um sinal de controle de trabalho (como pressionar Ctrl + Z) ou porque estão sendo rastreados. Zombie (Z): O kernel mantém várias estruturas de dados na memória para acompanhar os processos. Um processo pode criar vários processos filhos e eles podem sair enquanto o pai ainda está vivo. No entanto, essas estruturas de dados devem ser mantidas até que o pai obtenha o status dos processos filho. Tais processos terminados, cujas estruturas de dados ainda estão por aí, são chamados de zumbis.

Processos nos estados D e S são mostrados em *"sleeping"*, e aqueles no estado T são mostrados como *stopped*. O número de zumbis é mostrado como *"zombie"*. Estes valores são exibidos na segunda linha do comando top.

5.6 Streams

Streams fornecem uma abstração natural para casos de uso de blockchain que enfocam a recuperação geral de dados, o registro de data e hora e o arquivamento, em vez da transferência de ativos entre os participantes. As *streams* podem ser usados para implementar três tipos diferentes de bancos de dados em um blockchain:

- Um banco de dados de chave-valor ou armazenamento de documentos, no estilo do NoSQL;
- Um banco de dados de séries temporais, que se concentra na ordenação de entradas;
- Um banco de dados controlado por identidade, no qual as entradas são classificadas de acordo com o autor.

Eles podem ser considerados como "o que", "quando"e "quem"de um banco de dados compartilhado. Por trás da plataforma, cada item em uma stream é representado por uma transação de blockchain.

5.6.1 Utilizando Streams no Multichain

Qualquer número de *streams* pode ser criado em uma blockchain MultiChain e cada *stream* age como uma coleção independente de itens de somente inclusão. Cada item em uma *stream*

tem as seguintes características:

- Um ou mais editores que assinaram digitalmente esse item;
- Uma chave opcional para posterior recuperação conveniente;
- Alguns dados, que podem variar de um pequeno pedaço de texto a muitos *megabytes* de binário bruto;
- Um *timestamp*, que é retirado do cabeçalho do bloco no qual o item é confirmado.

As *streams* se integram ao sistema de permissões do MultiChain de várias maneiras. Primeiro, as *streams* só podem ser criadas por aqueles que têm permissão para fazer isso, da mesma forma que os ativos só podem ser emitidos por determinados endereços. Quando uma *stream* é criada, ela é aberta ou fechada. As *streams* abertas podem ser gravadas por qualquer pessoa que tenha permissão para enviar uma transação de blockchain, enquanto as *streams* fechadas são restritas a uma lista de endereços permitidos. No último caso, cada *stream* tem um ou mais administradores que podem alterar essas permissões de gravação ao longo do tempo. Para facilitar as validações do protótipo, foram utilizadas *streams* abertas.

A confidencialidade é um dos maiores desafios em um grande número de casos de uso de blockchain. Isso ocorre porque cada nó em uma blockchain vê uma cópia completa de todo o conteúdo da cadeia. As *streams* fornecem uma maneira natural de suportar dados criptografados em uma blockchain, da seguinte maneira:

1. Uma *stream* é usada pelos participantes para distribuir suas chaves públicas para qualquer esquema de criptografia de chave pública;
2. Uma segunda *stream* é usada para publicar dados, onde cada parte dos dados é criptografada usando criptografia simétrica com uma chave exclusiva.
3. Uma terceira *stream* fornece acesso a dados. Para cada participante que deve ver uma parte dos dados, é criada uma entrada de *stream* que contém a chave secreta desses dados, criptografada usando a chave pública do participante.

Isso fornece uma maneira eficiente de arquivar dados em uma blockchain, enquanto torna visível apenas para determinados participantes.

5.6.2 Recuperação de Streams

O valor central das *streams* está na indexação e na recuperação. Cada nó pode escolher em quais *streams* se inscrever, com a blockchain garantindo que todos os nós que se inscreverem em uma determinada *stream* verão os mesmos itens. (Um nó também pode ser configurado para assinar automaticamente cada nova *stream* criada). Se um nó estiver inscrito em uma *stream*, as informações poderão ser recuperadas dessa *stream* de várias maneiras:

- Recuperando itens da *stream* em ordem;
- Recuperando itens com uma chave particular;
- Recuperando itens assinados por um determinado editor;
- Listando as chaves usadas em uma *stream*, com contagens de itens para cada chave;
- listando os editores em uma *stream*, com contagens de itens.

Como mencionado no início, esses métodos de recuperação permitem que *streams* sejam usados para bancos de dados de chave-valor, bancos de dados de séries temporais e bancos de dados orientados a identidades. Todas as APIs de recuperação oferecem parâmetros de início e contagem, permitindo que subseções de listas longas sejam recuperadas com eficiência (como uma cláusula LIMIT no SQL). Valores negativos para início permitem que os itens mais recentes sejam recuperados.

Os *streams* podem conter vários itens com a mesma chave e isso naturalmente resolve a tensão entre a imutabilidade da blockchain e a necessidade de atualizar um banco de dados. Cada entrada de banco de dados efetiva deve receber uma chave exclusiva em sua aplicação, com cada atualização dessa entrada representada por um novo item de *stream* com sua chave. As APIs de recuperação de *stream* do MultiChain podem ser usadas para:

(a) recuperar a primeira ou última versão de uma determinada entrada

(b) recuperar um histórico de versão completo de uma entrada

(c) recuperar informações sobre várias entradas, incluindo a primeira e a última versões de cada um.

Devido à arquitetura P2P da blockchain, os itens de uma *stream* podem chegar a nós diferentes em ordens diferentes e o MultiChain permite que os itens sejam recuperados antes de serem "confirmados"em um bloco. Como resultado, todas as APIs de recuperação oferecem uma opção entre a ordenação global (padrão) ou local. A ordenação global garante que, uma vez que a cadeia tenha atingido o consenso, todos os nós recebam as mesmas respostas das mesmas chamadas da API. A ordenação local garante que, para qualquer nó específico, a ordenação de itens de uma *stream* nunca será alterada entre as chamadas de API. Cada aplicação pode fazer a escolha apropriada para suas necessidades.

6 RESULTADOS

Este capítulo apresenta os resultados da avaliação do protótipo do modelo L7SP. Na Seção 6.1 é abordado sobre a configuração adotada para a realização dos testes. As seções de 6.2 a 6.8 apresentam os resultados de cada cenário definido no capítulo anterior. Por fim, na seção 6.9 é realizada uma análise dos resultados, dando ênfase para a comparação entre os cenários e demonstração dos ganhos obtidos.

6.1 Configuração inicial

Antes de avaliar os cenários definidos no capítulo 5, foram realizados testes iniciais com o objetivo de estabelecer o tempo adequado para a execução dos cenários do teste de desempenho e o número de processos paralelos a ser parametrizado no injetor. O escopo desta avaliação inicial compreendeu realizar solicitações a partir do injetor diretamente em um nó multichain arbitrário, e a cada rodada ir aumentando o tempo de execução e o número de *threads* para verificar o comportamento do ambiente. Como resultado, foi estabelecido o *setup* de avaliação, compreendendo as seguintes definições:

- **Multichain:** 6 (seis) nós Multichain. Esta quantidade de servidores permite avaliar o comportamento da rede distribuída da blockchain e as funcionalidades implementadas pelo modelo L7SP. Um número muito grande de nós resulta em um tempo maior de configuração e gerenciamento do ambiente, tornando o processo de avaliação mais lento e aumentando o custo de infraestrutura.

- **Injetor:** tempo de execução de 5 minutos e 64 *threads* injetoras. Acima de 5 minutos de execução e com mais de 64 *threads* no injetor, os experimentos resultaram em uma alta quantidade de erros e *timeouts* nas solicitações da API Multichain, além de ocorrer travamentos no *middleware*. Além disto, um tempo de execução muito alto consequentemente resulta em um volume muito alto de dados transmitidos e armazenados, dificultando a avaliação posterior ao teste e a tornando mais lenta. Esta configuração se demonstrou segura e suficiente para uma avaliação do modelo, sendo um tempo adequado para a coletada de dados, validação e execução das rodadas. Este setup de avaliação é considerado para quase todos os cenários para fins de comparação, em apenas alguns cenários específicos onde o ambiente é escalado com a aplicação dos serviços de L7SP o número de *threads* é aumentado.

Para que as métricas avaliadas tivessem um bom nível de confiança, cada cenário foi executado 3 vezes e como resultado final foi considerada a média aritmética dos valores obtidos em cada execução.

6.2 Avaliação: Cenário 1

Durante a execução dos testes, verificou-se por meio do monitoramento do Agente Local que alguns recursos não foram impactados com as variações praticadas pelos cenários. Houve influência dos testes no percentual de utilização de CPU e na carga média dos nós, não afetando de forma significativa o consumo de memória e as outras variáveis abordadas na metodologia de avaliação, logo, estas foram removidas do escopo de avaliação e não compõem os resultados apresentados. A tabela 7 mostra os resultados obtidos após a execução do cenário 1, realizando a injeção de transações diretamente em apenas um nó Multichain. Durante os 5 minutos de execução, foram publicadas com sucesso 11.529 *streams* com um tempo médio de resposta de 1,64 segundos, e ocorreram 2 *timeouts*. O *throughput* foi de 40 transações por segundo.

Tabela 7 – Resultados - Cenário 1

Métrica	**Valor obtido**
Qtde transações	11.529
Tempo médio de resposta	1,64 segs
Throughput	40 tps
Qtde de erros	2

Fonte: elaborado pelo autor

O comportamento da CPU dos nós Multichain durante a execução do teste é exibido no gráfico da figura 18. Na figura, o eixo X apresenta o tempo em segundos e o eixo Y o percentual de utilização da CPU. O Node 1 foi o servidor que recebeu as requisições do injetor, logo, foi onde ocorreu mais processamento. A utilização de CPU no Node 1 ficou em média em 10%, com alguns picos durante o teste, chegando a 40% e 50%. Nota-se que os outros 5 servidores também tiveram o comportamento afetado, devido ao consenso distribuído, processamento e armazenamento das transações no *ledger*, porém, bem menos que o Node 1 que recebeu as requisições pela API. Os demais nós tiveram média de 6% com picos de 20% de utilização de CPU.

A carga nos servidores foi avaliada e o resultado apresentado no gráfico da figura 19. Os valores coletados são a média total do número de processos em espera na lista de execução somado ao número de processos atualmente em execução, dentro do último minuto. De acordo com o gráfico, podemos ver que a carga maior ocorreu em cima do Node 1, que chegou a atingir picos de 30% da capacidade do servidor. Os demais nós trabalharam na maior parte do tempo abaixo de 15% de carga, o que demonstra que a carga realizada neste cenário é tranquilamente suportada pelo hardware dos servidores Multichain.

6.3 Avaliação: Cenário 2

Os cenários seguintes passam a usar o gerenciador do modelo L7SP como intermediário, recebendo as requisições do injetor, submetendo as mesmas aos serviços habilitados no protó-

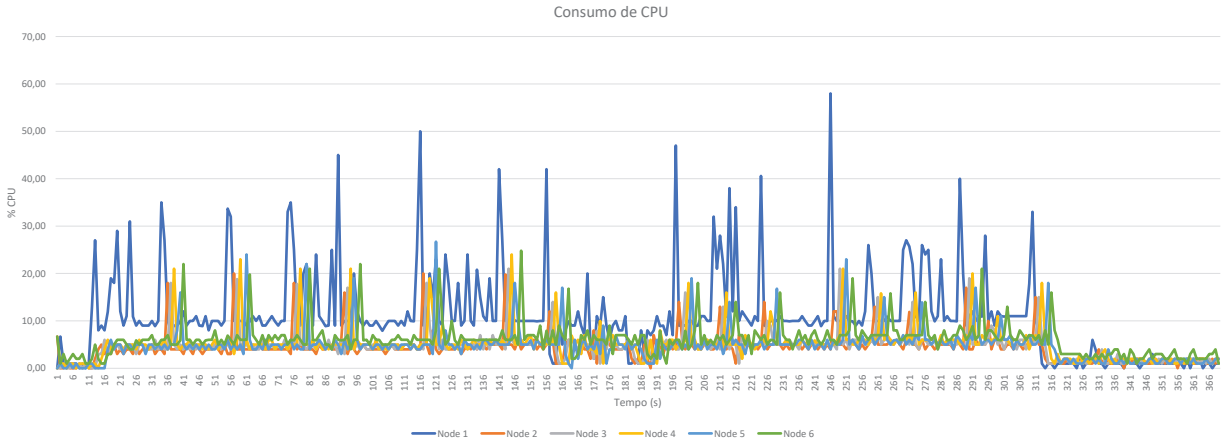

Figura 18 – Cenário 1 - Utilização de CPU
Fonte: elaborado pelo autor

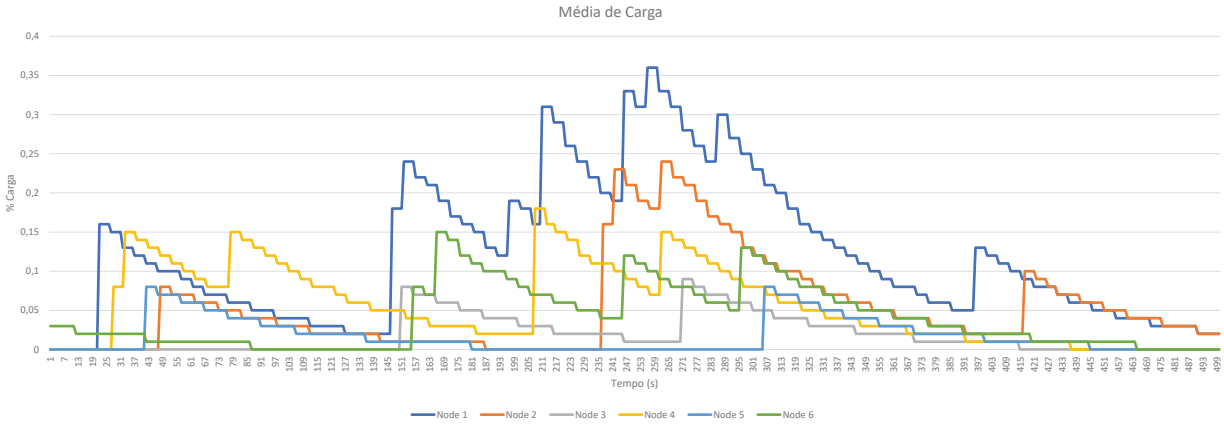

Figura 19 – Cenário 1 - Carga média
Fonte: elaborado pelo autor

tipo, e realizando a comunicação com os nós do Multichain. No cenário 2 se observa o efeito da compactação de dados no desempenho do processamento das requisições submetidas ao Multichain. A tabela 8 mostra os resultados obtidos após a execução do cenário 2, realizando a injeção de transações mais uma vez em apenas um nó Multichain, mas agora passando pelo Gerenciador L7SP e utilizando o serviço de compactação de dados. A melhora no desempenho neste cenário foi bem sensível, não havendo grande diferença nos resultados. Durante os 5 minutos de execução, foram publicadas com sucesso 13.212 *streams* com um tempo médio de resposta de 1,44 segundos, e ocorreram 4 *timeouts*. O *throughput* foi de 41,8 transações por segundo.

O comportamento da CPU dos nós Multichain durante a execução do cenário 2 é exibido no gráfico da figura 20. O Node 1 novamente foi o servidor que recebeu as requisições do

Tabela 8 – Resultados - Cenário 2

Métrica	Valor obtido
Qtde transações	13.212
Tempo médio de resposta	1,44 segs
Throughput	41,8 tps
Qtde de erros	4

Fonte: elaborado pelo autor

injetor, logo, apresentou novamente um maior consumo de CPU. O comportamento foi muito semelhante ao cenário 1, onde a utilização de CPU no Node 1 ficou em média em 10% e os outros 5 servidores em média 6%, apresentando também o mesmo comportamento de picos mais altos. Apesar de haver uma pequena diferença na capacidade de processamento do sistema com a aplicação da compactação de dados, o consumo dos recursos nos nós praticamente não sofreu modificação.

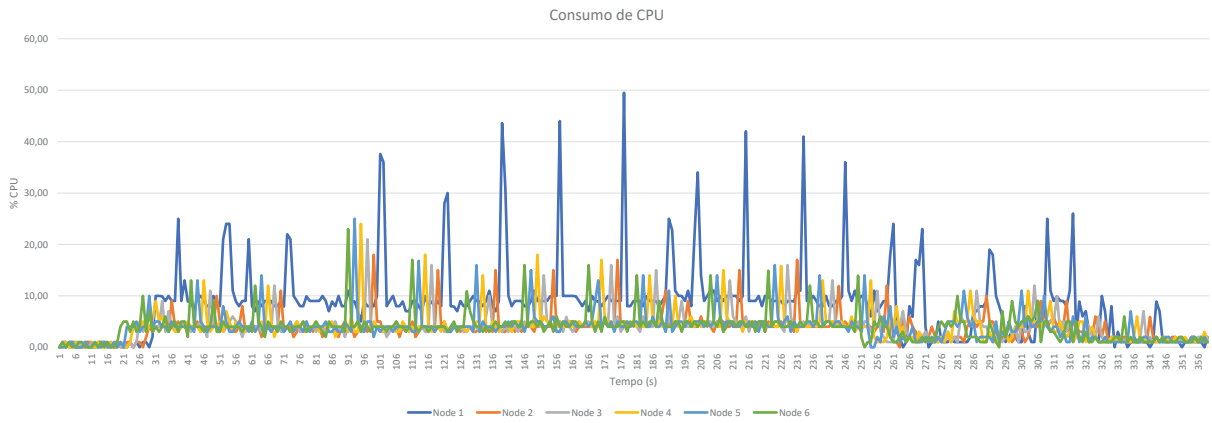

Figura 20 – Cenário 2 - Utilização de CPU
Fonte: elaborado pelo autor

A carga média nos servidores é apresentada no gráfico da figura 21. De acordo com o gráfico, é possível ver que o comportamento é semelhante ao cenário 1, mantendo a mesma média de valores, e variando somente o momento de ocorrência dos picos de carga.

6.4 Avaliação: Cenário 3

No cenário 3 se observa o efeito do balanceamento de carga no desempenho do processamento das requisições submetidas ao Multichain. Neste cenário foi aplicada a estratégia de balanceamento Round Robin, para distribuir de forma equilibrada a carga de processamento entre os 6 nós Multichain. A tabela 9 mostra os resultados obtidos após a execução do cenário 3, realizando a injeção de transações no Gerenciador L7SP, aplicando a lógica de balanceamento de carga e roteando as transações para os nós Multichain. A melhora no desempenho neste

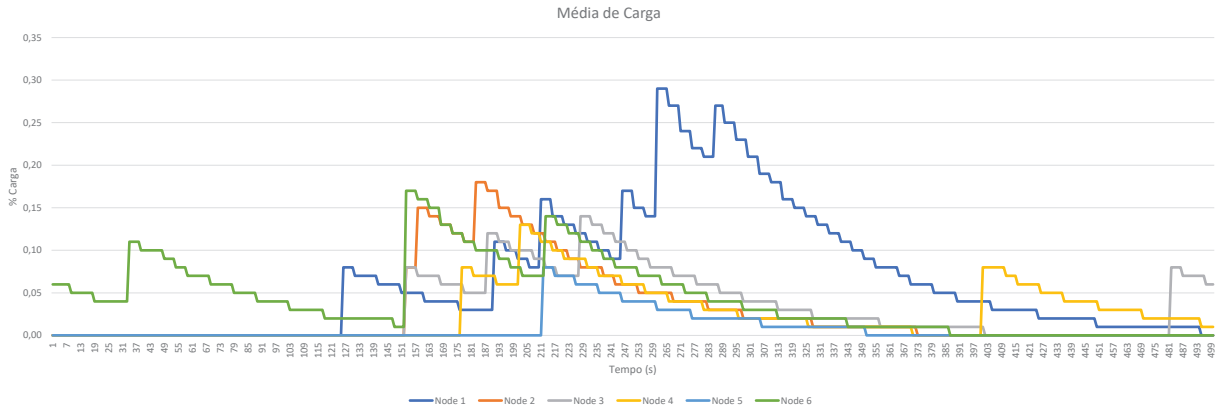

Figura 21 – Cenário 2 - Carga média
Fonte: elaborado pelo autor

cenário foi bem considerável, aumentando em um pouco mais de 400% a capacidade de processar transações simultâneas com relação ao cenário 1, considerando os mesmos parâmetros do injetor. Durante os 5 minutos de execução, foram publicadas com sucesso 62.590 *streams* com um tempo médio de resposta de 0,07 segundos, não ocorrendo nenhum *timeout*. O *throughput* foi de 208,6 transações por segundo.

Tabela 9 – Resultados - Cenário 3

Métrica	**Valor obtido**
Qtde transações	62.590
Tempo médio de resposta	0,07 segs
Throughput	208,6 tps
Qtde de erros	0

Fonte: elaborado pelo autor

O comportamento da CPU dos nós Multichain durante a execução do cenário 3 é exibido no gráfico da figura 22. Este cenário demonstrou um perfeito equilíbrio do consumo de CPU entre os nós do pool Multichain, onerando muito menos os processadores das máquinas. A utilização de CPU durante o teste variou quase todo o tempo entre 0 e 1%, com picos esporádicos de 2%.

Da mesma forma neste cenário, a média de carga foi muito mais equilibrada, conforme demonstrado na figura 23. Na maior parte do tempo na maioria dos nós, ou não houve carga, ou a mesma foi muito baixa. Os poucos picos de carga que houveram atingiram apenas 2 nós e chegaram no máximo a 15% da carga.

6.5 Avaliação: Cenário 4

No cenário 4 se observa novamente o efeito do balanceamento de carga no desempenho do processamento das requisições, mas agora com a aplicação da estratégia *Least Connections* para

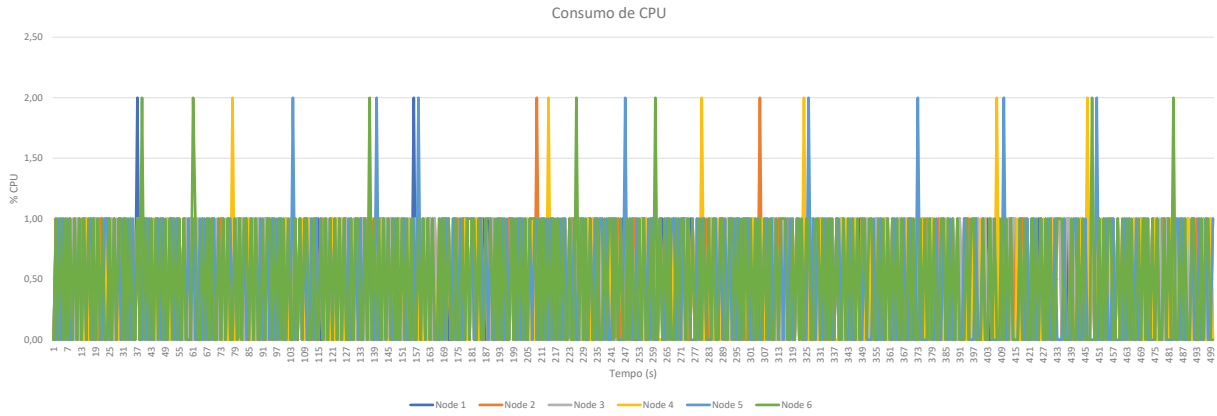

Figura 22 – Cenário 3 - Utilização de CPU
Fonte: elaborado pelo autor

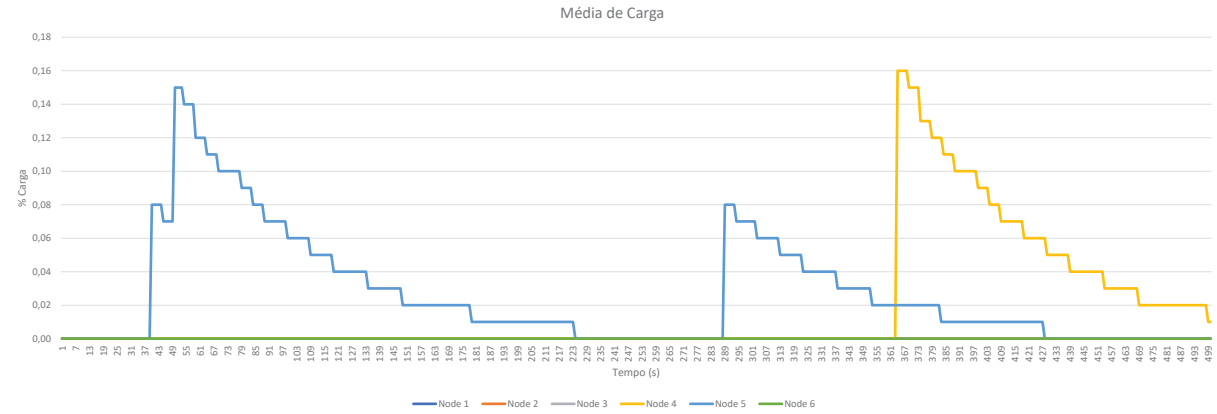

Figura 23 – Cenário 3 - Carga média
Fonte: elaborado pelo autor

distribuir a carga de processamento entre os 6 nós Multichain. A tabela 10 mostra os resultados obtidos após a execução do cenário 4, realizando a injeção de transações no Gerenciador L7SP, aplicando a lógica de balanceamento de carga e roteando as transações para os nós Multichain. Neste cenário houve uma degradação do desempenho, comparado inclusive ao cenário 1. O *overhead* para a aplicação da regra de balanceamento que varre a lista de conexões dos nós e problemas na implementação do algoritmo são as possíveis causas sugeridas para a obtenção deste resultado. Durante os 5 minutos de execução, foram publicadas com sucesso 7.995 *streams* com um tempo médio de resposta de 2,4 segundos, ocorrendo 6 *timeouts*. O *throughput* foi de 26,7 transações por segundo.

Conforme apresentado na figura 24, o comportamento da CPU dos nós Multichain durante a execução do teste do cenário 4 foi equilibrado, se mantendo estável quase todo o tempo. Foi

Tabela 10 – Resultados - Cenário 4

Métrica	Valor obtido
Qtde transações	7.995
Tempo médio de resposta	2,4 segs
Throughput	26,7 tps
Qtde de erros	0

Fonte: elaborado pelo autor

observado uma variação entre 2 a 3% de consumo de CPU em média e pequenos picos no Node 1 em torno de 6% e em um único momento a 17%.

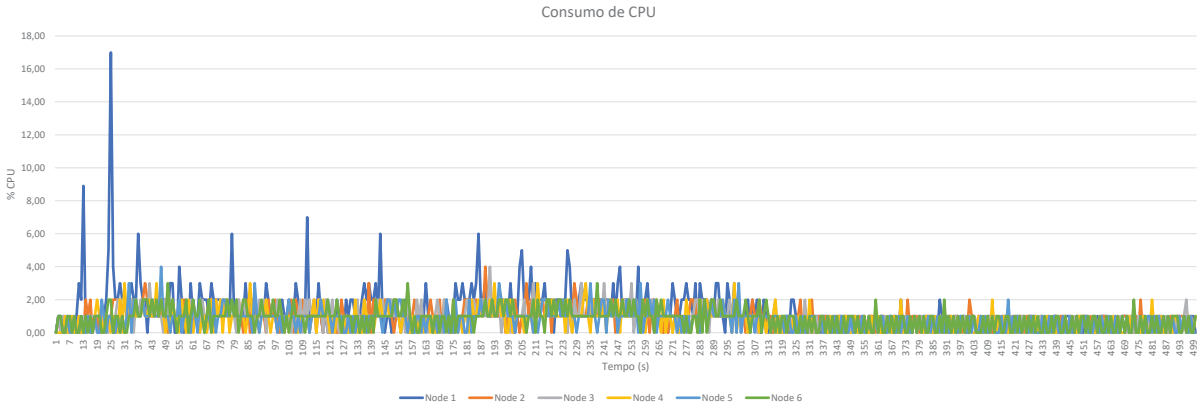

Figura 24 – Cenário 4 - Utilização de CPU
Fonte: elaborado pelo autor

Conforme demonstrado na figura 25, a média de carga no cenário 4 também se manteve estável com valores semelhantes ao cenário 3, porém, com a diferença de haver um volume um pouco maior de picos de carga.

6.6 Avaliação: Cenário 5

No cenário 5 as transações são submetidas aos serviços de balanceamento de carga e compactação de dados do protótipo L7SP com o objetivo de obter o melhor desempenho no processamento das requisições Multichain. Neste cenário a estratégia de balanceamento de carga aplicada foi o *Round Robin*, que apresentou o melhor desempenho nos cenários anteriores. A tabela 11 mostra os resultados obtidos após a execução do cenário 5, realizando a injeção de transações no Gerenciador L7SP, aplicando as lógicas de balanceamento de carga e compactação de dados, e roteando as transações para os nós Multichain. Com certeza este cenário superou os anteriores no resultado, mas a melhora de desempenho com relação ao cenário 3 foi sensível, em torno de 5,4% de ganho. Durante os 5 minutos de execução, foram publicadas com sucesso 65.979 *streams* com um tempo médio de resposta de 0,04 segundos, não ocorrendo

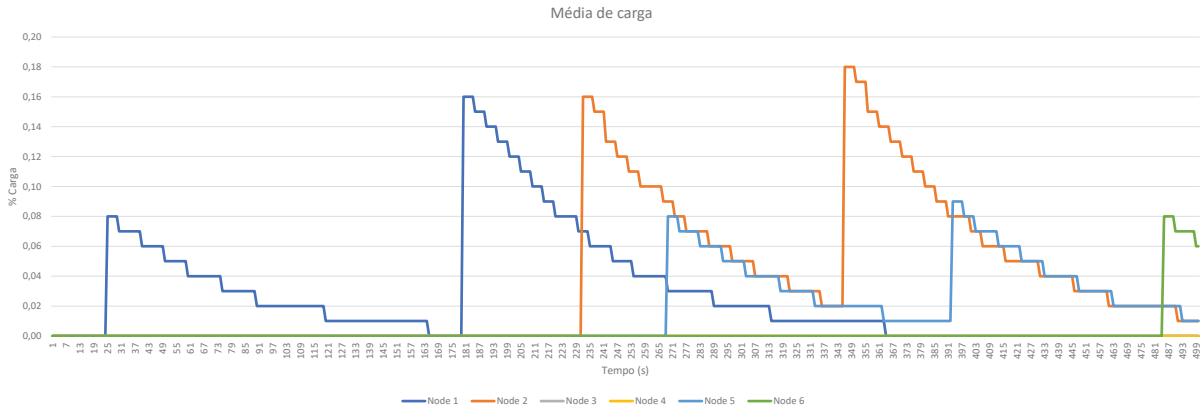

Figura 25 – Cenário 4 - Carga média
Fonte: elaborado pelo autor

nenhum *timeout*. O *throughput* foi de 219,9 transações por segundo.

Tabela 11 – Resultados - Cenário 5

Métrica	**Valor obtido**
Qtde transações	65.979
Tempo médio de resposta	0,04 segs
Throughput	219,9 tps
Qtde de erros	0

Fonte: elaborado pelo autor

O comportamento da CPU dos nós Multichain durante a execução do cenário 5 é exibido no gráfico da figura 26. Da mesma forma que no cenário 3, este cenário se demonstrou também muito equilibrado no consumo de CPU. A diferença é que o consumo neste caso foi sensivelmente maior, a utilização de CPU durante o teste variou a maior parte do tempo entre 0 e 1%, com muitos picos chegando a 2% e poucos picos de 3%.

Conforme demonstrado na figura 25, a média de carga no cenário 5 também se manteve estável com carga quase nula da mesma forma que nos 2 cenários anteriores, porém, com a diferença de haver poucos picos e estes chegarem somente a 8% da carga no máximo.

6.7 Avaliação: Cenário 6

O cenário 6 possui exatamente a mesma configuração L7SP do cenário anterior, porém, com a diferença de que o injetor é configurado para aumentar o número de *threads* para 256. Este cenário explora principalmente a escalabilidade da solução, avaliando se ela consegue suportar o aumento do volume de entrada de transações. A tabela 12 mostra os resultados obtidos após a execução do cenário 5, realizando a injeção de transações no Gerenciador L7SP, aplicando as lógicas de balanceamento de carga e compactação de dados, e roteando as transações para os

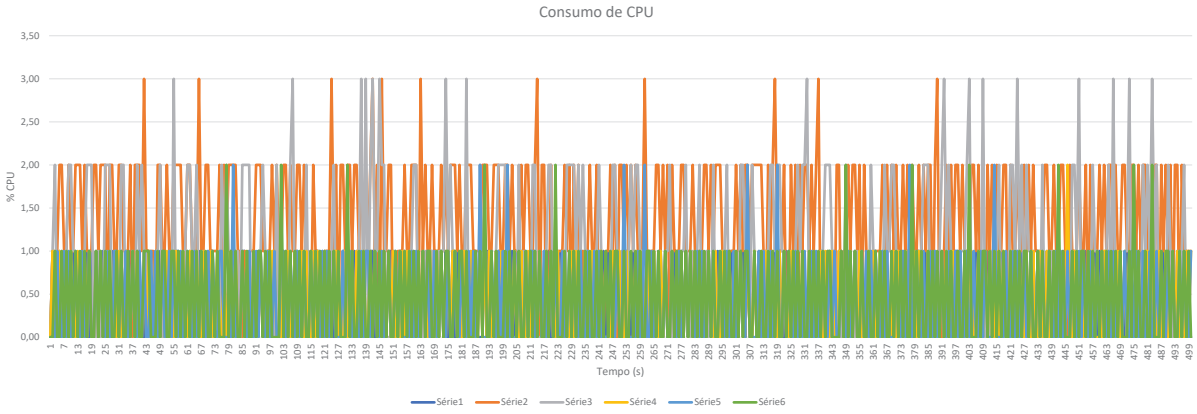

Figura 26 – Cenário 5 - Utilização de CPU
Fonte: elaborado pelo autor

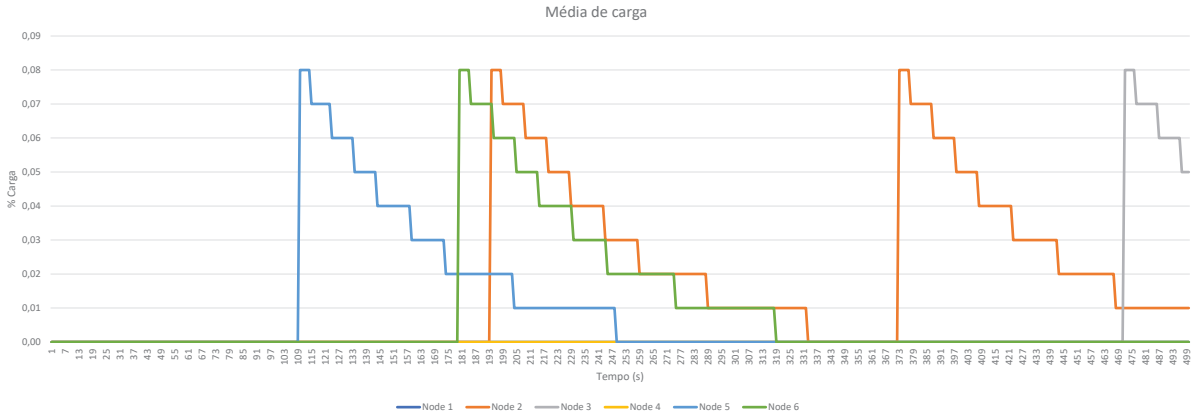

Figura 27 – Cenário 5 - Carga média
Fonte: elaborado pelo autor

nós Multichain. Este foi o cenário onde foi obtido o melhor resultado, superando os anteriores em todos os aspectos. Durante os 5 minutos de execução, foram publicadas com sucesso 68.762 *streams* com um tempo médio de resposta de 0,02 segundos, não ocorrendo nenhum *timeout*. O *throughput* foi de 229,2 transações por segundo.

O comportamento da CPU dos nós Multichain durante a execução do cenário 6 é exibido no gráfico da figura 28. Da mesma forma que no cenário 3 e 5, este cenário se demonstrou também muito equilibrado no consumo de CPU. Os valores do percentual de utilização de CPU são idênticos aos do cenário 5.

Conforme demonstrado na figura 29, a média de carga no cenário 6 decresce no início da execução do programa e apresenta picos partir da metade da apuração, estes chegando ao máximo de 18% da capacidade dos servidores.

Tabela 12 – Resultados - Cenário 6

Métrica	Valor obtido
Qtde transações	68.762
Tempo médio de resposta	0,02 segs
Throughput	229,2 tps
Qtde de erros	0

Fonte: elaborado pelo autor

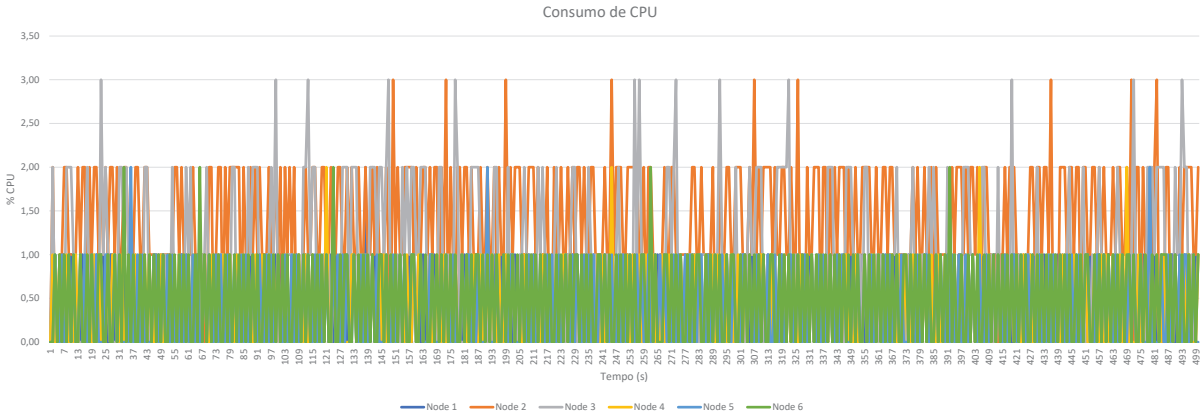

Figura 28 – Cenário 6 - Utilização de CPU
Fonte: elaborado pelo autor

6.8 Avaliação: Cenário 7

Para a avaliação dos níveis de compressão obtidos e do impacto no espaço de armazenamento foram utilizados 2 fontes: (a) implementação do comando *du -s* no Agente Local para verificar o espaço em disco dentro do diretório da blockchain e (b) o registro em logs do gerenciador com o resultado obtido na compressão, exibindo o tamanho original e tamanho compactado. A *stream* utilizada em todos os cenários de avaliação possuía 4.972 bytes em seu tamanho original. Quando submetida a compressão bzip2 implementada em L7SP o resultado era um bloco de 1303 bytes. Porém, o resultado gerado pela compressão é em formato binário, e não serve para a transmissão por meio da API Multichain. Realizando a conversão da *stream* compactada para o formato ASCII, chegou-se a um tamanho final de 2.606 bytes, resultando em uma redução de aproximadamente 47,6 % no tamanho dos dados transmitidos. No espaço de armazenamento verificou-se que os testes sem compressão resultaram em um aumento de 24,2 KB enquanto os testes com compressão resultaram em apenas 1,7 KB de aumento. Acredita-se que esta diferença mais significativa no espaço acumulado em disco se deve não apenas ao armazenamento no *ledger*, mas também ao registro em logs e outros eventos, lembrando que as transações são replicadas para todos os nós. Para este cenário de avaliação não se torna algo tão relevante, mas ao longo prazo com o crescimento da blockchain e o volume transacional a

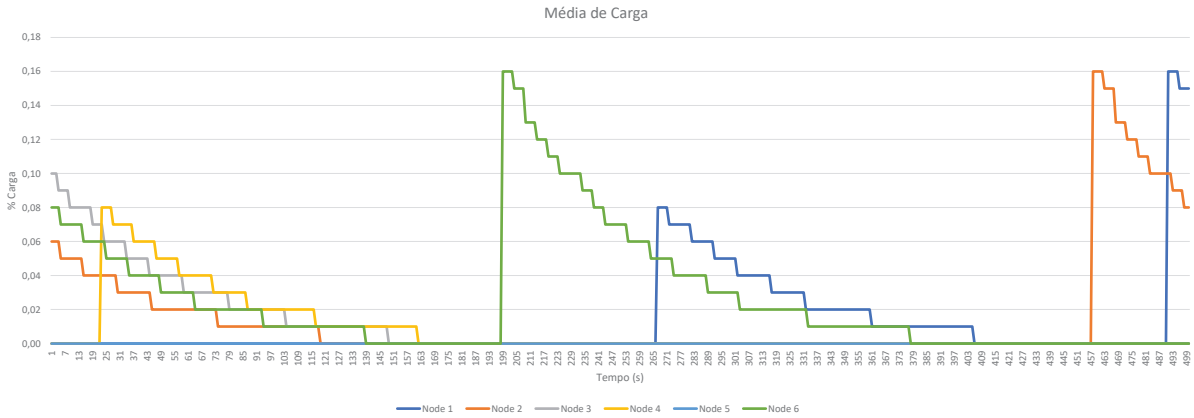

Figura 29 – Cenário 6 - Carga média
Fonte: elaborado pelo autor

compressão pode ser uma ótima alternativa para reduzir a utilização de recursos e obter melhor desempenho na transmissão dos dados.

6.9 Análise dos resultados

Após a execução dos cenários de avaliação, esta seção apresenta uma análise mais detalhada com a interpretação dos resultados, comparando os diferentes cenários com base nos BKPI's definidos no capítulo 5.

6.9.1 Eficiência de L7SP

A figura 30 apresenta uma comparação da capacidade de processamento do sistema nos diferentes cenários.

Nota-se que o cenário 6 onde as transações foram submetidas aos serviços de compressão de dados e balanceamento de carga foi o que apresentou melhor resultado, porém as *threads* de injeção tiveram seu número elevado para 256 para simular uma carga mais alta. Embora este cenário seja interessante para a avaliar o limite de capacidade da solução, acaba não sendo válido para comparação. O cenário 3 possui a mesma configuração do cenário 1, com 64 *threads* de injeção de transações. A diferença é que o cenário 3 utiliza L7SP com o serviço de balanceamento de carga *Round Robin* habilitado. Comparando o cenário 3 com o cenário 1, com base na métrica estabelecida no capítulo 5, houve um ganho de 421,5 % no desempenho, devido ao balanceamento de carga *Round Robin* com 6 nós no pool de conexões. Uma vez que *Round Robin* foi o algoritmo de balanceamento de carga que apresentou melhor desempenho nas validações iniciais, foi a estratégia considerada para a evolução dos testes. No cenário 5, combinando os serviços de balanceamento de carga RR com compressão de dados se obtém o

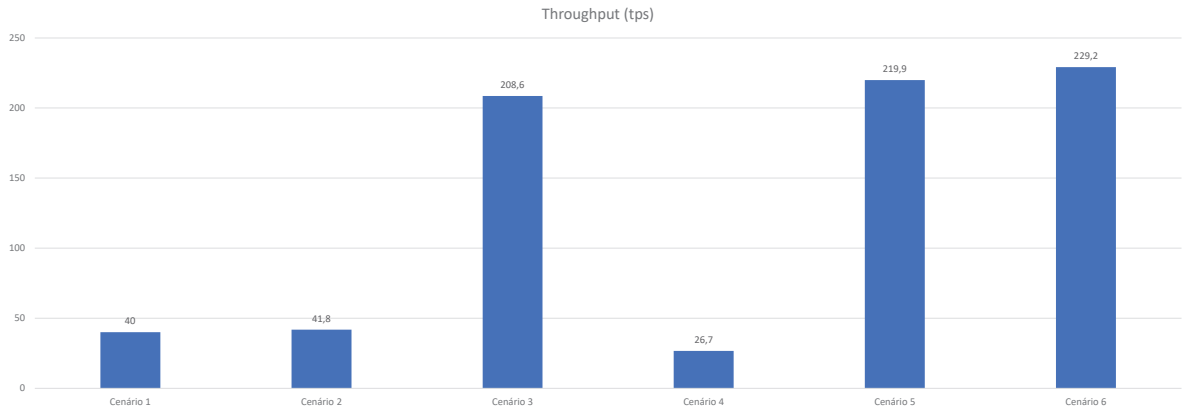

Figura 30 – Comparação - Throughput
Fonte: elaborado pelo autor

melhor resultado para fins de comparação. Comparado ao cenário 3, a configuração do cenário 5 conseguiu ainda melhorar o desempenho em mais 5,42%. A tabela 13 apresenta um resultado consolidados de todos os cenários, facilitando a comparação entre eles.

Tabela 13 – Comparação de resultados

Cenário	L7SP	LB - Round Robin	LB - Least Conn	Compressão	Num Threads	Qtde Trns	Throughput	Tempo Médio Resp	Qtde Erros
1					64	11.529	40	1,64	2
2	✓			✓	64	13.212	41,8	1,44	4
3	✓	✓			64	62.590	208,6	0,07	0
4	✓		✓		64	7.995	26,7	2,4	6
5	✓	✓		✓	64	65979	219,9	0,04	0
6	✓	✓		✓	256	68762	229,2	0,02	0

Fonte: elaborado pelo autor

A figura 31 mostra os resultados com relação ao tempo médio de resposta das requisições realizadas na API Multichain em cada cenário. Verifica-se que os cenários que apresentaram melhor resultado de *throughput* são os que também apresentaram um tempo médio de resposta mais curto, é o caso dos cenários 3, 5 e 6. Com uma carga mais leve devido ao processamento mais equilibrado e menos oneroso, os nós conseguem processar mais transações simultâneas e responder as solicitações mais rapidamente.

Outro ponto relevante que também é apresentado na tabela 13 é a quantidade de erros ocorridos nos cenários de avaliação, seja por timeout ou erros retornados pela API Multichain. Para

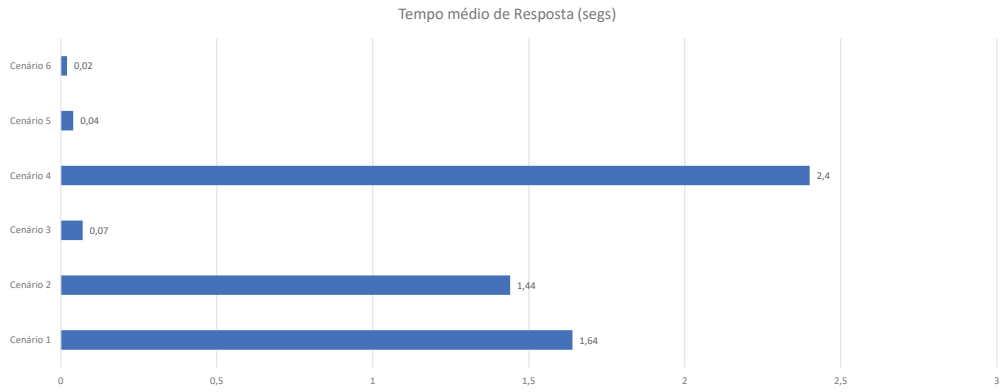

Figura 31 – Comparação - Tempo médio de resposta
Fonte: elaborado pelo autor

obter uma avaliação de alta qualidade, estipulou-se como aceitável uma taxa de erro no máximo 0,1%, isto significa 1 erro a cada 1.000 transações. Verificou-se na avaliação dos cenários que tentativas de aumento da carga além das configurações aqui apresentadas resultaram em um número excessivo de erros nos testes, e algumas vezes até no travamento do *middleware* Multichain.

6.9.2 L7SP e o empilhamento de serviços

É notável os ganhos gerados pelo empilhamento de serviços provido por L7SP analisando os resultados obtidos. Dos serviços que foram implementados no protótipo e avaliados neste estudo, é possível destacar 2 pontos:

- Balanceamento de Carga: a estratégia *Round Robin* apresentou melhor desempenho que *Least Connections* no balanceamento de carga, e se atribui este resultado pelo fato de que *Round Robin* apresenta desempenho ótimo quando os recursos são homogêneos, que foi o caso do ambiente utilizado, onde todos os nós possuíam exatamente a mesma configuração de hardware e software. A técnica *Least Connections* decepcionou com relação ao seu resultado, não mostrando ser atrativa para utilização neste contexto. Acredita-se que este resultado obtido no cenário 4 se deve ao *overhead* causado pelo sincronismo e verificações na lista de conexões, ou ainda mesmo a problemas de implementação no protótipo.

- Compressão de dados: a compressão de dados utilizando a função bzip2 apresentou ótimos resultados. Além de diminuir o espaço de armazenamento na blockchain, demonstrou ganhos de desempenho por possibilitar a otimização do tráfego de rede com a transmissão de pacotes menores. Futuramente outros métodos podem ser incorporados como

serviço para oferecer diferentes níveis de compressão e desempenho, podendo se aplicar o que for mais adequado dependendo do cenário.

7 CONCLUSÃO

A popularidade alcançada pela tecnologia blockchain e sua crescente utilização nos mais diversos mercados demandam a implementação de soluções descentralizadas robustas, que sejam capazes de processar um alto volume de transações simultâneas em uma rede distribuída e também suportar um rápido crescimento sem impactar na operação. As soluções propostas atualmente para melhorar o desempenho e a escalabilidade da blockchain são de difícil implementação, necessitando de alterações estruturais na aplicação, sendo incompatíveis com as soluções existentes. O modelo L7SP é proposto como uma alternativa às soluções existentes, provendo o empilhamento de novos serviços e se acoplando de forma transparente à blockchain, mantendo as plataformas de *ledger* distribuído e as aplicações que se integram à elas sem alteração em suas configurações. A seção 1.2 apresentou a seguinte pergunta de pesquisa:

Como seria um modelo flexível de arquitetura para a Blockchain Privada capaz de melhorar o desempenho e gerenciamento do sistema sem comprometer a segurança e descentralização, sendo compatível com as soluções existentes?

Para responder esta questão, L7SP introduziu uma camada de gerenciamento para intermediar as transações com a blockchain e habilitar diferentes serviços na camada de aplicação. O protótipo L7SP foi avaliado por meio de vários cenários de teste de desempenho, e apresentou as seguintes características que se destacam:

- Ganho de desempenho do sistema, processando um maior volume de transações simultâneas mais rapidamente. O empilhamento de serviços ofertados pelo modelo possibilitou que fossem atingidos valores de *throughput* e tempos de resposta melhores dos que são praticados atualmente na configuração padrão da blockchain.

- Gerenciamento mais eficiente, por meio do monitoramento online de forma centralizada, contemplando o estado dos recursos locais dos nós e o desempenho da rede blockchain. A visão em tempo real ofertada pela camada de monitoramento de L7SP, além de ter viabilizado a geração de dados para avaliação do modelo, permite um acompanhamento constante do estado global do sistema. Isto é essencial para antecipar problemas e realizar modificações no ambiente de forma mais ágil e com menor esforço.

- Flexibilidade, devido aos serviços ofertados na camada de gerenciamento de L7SP, podendo ser configurados e combinados conforme a necessidade do usuário, gerando resultados e características diferenciadas. A camada de gerenciamento permite ainda que novos serviços possam ser adicionados à blockchain ao longo do tempo e a solução seja configurada para que os serviços sejam acionados de forma estática por intervenção do usuário ou de forma dinâmica por ação do sistema estimulado por variáveis monitoradas na rede e *thresholds* pré-definidos.

- Compatibilidade, devido ao acoplamento transparente de L7SP à blockchain e demais

componentes do sistema, sem gerar atrito para os usuários finais, alterações de código ou configuração nas aplicações. L7SP não interfere nas funcionalidades de consenso distribuído, permissões de acesso e publicação dos blocos no *ledger*, mantendo as características de segurança e descentralização da rede blockchain. L7SP não atua como uma terceira parte confiável e sim como um provedor de serviços para a rede blockchain, logo, continua mantendo os benefícios que são o diferencial da tecnologia.

7.1 Contribuições

Este trabalho apontou as limitações de desempenho existentes na blockchain por intermédio do referencial teórico consolidado na revisão de literatura e na avaliação realizada. Uma vez acoplado o protótipo de L7SP, testes de desempenho foram realizados comparando-o com a solução blockchain em sua configuração padrão. Os serviços de balanceamento de carga e compressão de dados de L7SP resultam em uma melhora na escalabilidade do sistema, diminuindo a latência e aumentando o *throughput* transacional. A solução blockchain privada tornou-se mais eficiente, otimizando a utilização dos recursos, consequentemente gerando mais economia para a rede. Destaca-se como contribuição científica deste trabalho os seguintes itens:

- Introdução do BKPI: Os indicadores chave de desempenho foram sugeridos neste trabalho como uma visão essencial a ser implantada em qualquer blockchain. O modelo proposto sugere que cada administrador de sistema deve definir seus BKPI's, que são as métricas mais impactantes para o seu negócio que é suportado pela blockchain. Estas métricas são monitoradas em tempo real e permitem que ações rápidas possam ser tomadas para garantir o desempenho da rede e a estabilidade da operação.

- Novas métricas: As métricas de ganho de desempenho baseada na variação do *throughput* e a taxa de erro foram introduzidas neste trabalho para a prover a avaliação de desempenho da blockchain com maior qualidade e garantir uma comparação de cenários e interpretação de resultados de forma mais clara.

- Paradigma parcialmente descentralizado: Um pouco diferente dos conceitos originais da blockchain e da maioria dos estudos sobre a tecnologia, este trabalho sugere uma topologia parcialmente descentralizada. Isto se deve a inclusão da camada de gerenciamento para prover 2 contribuições específicas: (a) empilhamento de serviços para ofertar maior desempenho e escalabilidade para a blockchain e (b) monitoramento da rede centralizado e em tempo real.

Em complemento e de forma mais ampla, os resultados obtidos neste trabalho geram maior confiança, contribuindo para aumentar a adesão à blockchain em ambientes privados e de consórcio de corporações. Além disso, pode incentivar pesquisadores a buscar novas soluções para as lacunas que ainda existem.

7.2 Limitações e Trabalhos Futuros

Durante a especificação do modelo e sua posterior avaliação, verificou-se algumas limitações que não foram atendidas na implementação do protótipo de L7SP e também oportunidades que podem ser exploradas em futuros estudos. A estrutura de L7SP da maneira que foi concebida, permite que o modelo seja evoluído e integrado com outras soluções. Desta forma, dentre possíveis itens que podem vir a ser alvos de trabalhos futuros, destacam-se os seguintes:

- Alocação dinâmica de recursos: Uma linha de pesquisa interessante que pode ser explorada é a alocação dinâmica de recursos incorporada como um serviço de L7SP para também prover maior escalabilidade, velocidade no processamento das transações, equilíbrio e economia na utilização dos recursos. A topologia e estrutura de L7SP pode ser aproveitada para a implementação de técnicas de elasticidade para a blockchain em nuvem, baseado em *thresholds* pré-definidos, utilizando a camada de gerenciamento como um atuador.

- Monitoramento online: Como foi possível verificar ao longo deste estudo, medir o desempenho da blockchain e monitorá-lo de maneira eficiente é uma tarefa complexa, L7SP oferece alternativas de monitoramento online dos recursos dos nós blockchain e também o monitoramento de variáveis de desempenho do sistema por meio dos BKPI's. Como trabalhos futuros, se sugere um aprimoramento da camada de monitoramento com a implementação de orientação a eventos, possibilitando a geração de alertas e a invocação de serviços baseados em *thresholds* pré-definidos. Além disto, pode-se prover dashboards conectados a camada de monitoramento, facilitando a visualização do estado global do sistema por parte do usuário, permitindo decisões e ações corretivas mais ágeis.

- Configuração centralizada e automática: Uma dificuldade existente em redes blockchain com um grande número de nós é realizar a configuração dos parâmetros da solução. Alguns parâmetros e configurações de sistema não são passíveis de ajuste por API em tempo de execução, necessitando de replicação manual em todos os nós, gerando um esforço considerável por parte do usuário. L7SP possui o Agente Local que funciona como um sensor em cada nó, coletando informações locais do ambiente. Em trabalhos futuros, pode-se evoluir o Agente Local para que funcione como atuador, e receba estímulos do Gerenciador para realizar a atualização automática de parâmetros e configurações de sistema. Estes estímulos são intervenções do usuário na camada de gerenciamento, que são realizadas uma vez de forma centralizada e são replicadas automaticamente para todos os nós da rede. Para viabilizar esta solução também é preciso prover mecanismos de segurança específicos e robustos, garantindo que somente usuários autorizados e devidamente autenticados possam realizar as ações de configuração centralizadas.

- Novos serviços L7SP: Este trabalho provê uma camada de gerenciamento que realiza o empilhamento de serviços para melhorar o gerenciamento e desempenho da blockchain

privada. Duas estratégias foram sugeridas para melhorar as características desejadas: balanceamento de carga e compressão de dados. No futuro, novos serviços podem ser incorporados ao gerenciador ou criadas novas soluções para interagir em conjunto com eles, gerando novos ganhos para as soluções blockchain.

- Plano de armazenamento: O banco de dados da blockchain cresce linearmente, com o passar do tempo e o aumento do volume transacional, devem surgir problemas de desempenho devido ao tamanho excessivo do *ledger*. Este trabalho oferta a compressão de dados como solução para reduzir o tamanho do *ledger*, mas outras alternativas podem ser exploradas futuramente e incluídas como serviço no modelo L7SP para gerar maiores ganhos no plano de armazenamento da blockchain.

- Interoperabilidade: A solução do modelo L7SP para fins de avaliação foi focada na integração com o *middleware* Multichain. Uma futura solução poderia explorar a construção de uma camada de abstração para realizar a conversão de diferentes protocolos de mensagem blockchain, tornando L7SP agnóstico às plataformas de *ledger* distribuído. Isto o tornaria compatível com as plataformas mais relevantes e disponíveis no mercado. O *middleware* gerenciador de L7SP pode ser adaptado para viabilizar a comunicação entre diferentes blockchains e prover recursos de gerenciamento e serviços para todos eles.

- Tolerância a falhas: Neste este estudo foram ressaltadas as vantagens de uma solução blockchain privada parcialmente descentralizada, abordando fatores que favorecem a implementação do modelo L7SP. No entanto, por ser um componente centralizado nesta topologia, L7SP torna o sistema mais suscetível a falhas. Para mitigar este risco, sugere-se em trabalhos futuros a configuração e avaliação de L7SP com um pool de servidores, utilizando balanceamento de carga na camada de transporte, não necessitando de alterações no código do protótipo. Esta estratégia deve aumentar a escalabilidade do modelo e o tornar mais tolerante a falhas. Uma vez que L7SP se acopla de forma transparente, outra alternativa que pode ser explorada é o redirecionamento de rede dos clientes externos diretamente para os nós do *middleware* blockchain.

REFERÊNCIAS

ANIELLO, L. et al. A Prototype Evaluation of a Tamper-resistant High Performance Blockchain-based Transaction Log for a Distributed Database. **IEEE**, [S.l.], p. 151–154, 2017.

B, K.; S, C. Guidelines for performing systematic literature reviews in software engineering. technical report. **Keele University**, [S.l.], 2007.

BASESCU, C.; KOKORIS-KOGIAS, E.; FORD, B. A. Poster: low-latency blockchain consensus. **IEEE**, [S.l.], 2017.

BIOLCHINI J MIAN P, N. A.; G, T. Systematic review in software engineering. technical report. **PESC**, [S.l.], 2005.

BITCOIN and ethereum vs visa and paypal – transactions per second. 2018.

BITCOIN.ORG. **Bitcoin developer guide**. Accessado: nov. 2017, `https://bitcoin.org/en/developer-guide`.

BITSHARES. **Bitshares - your share in the decentralized exchange**. 2018.

BOZIC, N.; PUJOLLE, G.; SECCI, S. A tutorial on blockchain and applications to secure network control-planes. **2016 3rd Smart Cloud Networks and Systems, SCNS 2016**, [S.l.], 2016.

BRANCO, K. R. L. J. C. Indices de carga e desempenho em ambientes paralelos/distribuídos — modelagem e métricas. **Tese (Doutorado em Ciência da Computação) Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos.**, [S.l.], 2004.

BRUCE, J. D. **The mini-blockchain scheme**. Disponível em: <https://cryptonite.info/files/mbc-scheme-rev3.pdf>. Acesso em: nov. 2017.

BUTERIN, V. **On public and private blockchains**. Accessado: nov. 2017, `https://blog.ethereum.org/2015/08/07/on-public-and-private-blockchains`.

CASAVANT T. L; KUHL, J. G. A taxonomy of scheduling in general-purpose distributed computing systems. **IEEE Trans. Softw. Eng., Piscataway, NJ, USA, v. 14, p. 141—154**, [S.l.], 1998.

CASTRO M, L. B. Practical byzantine fault tolerance. In: IN: PROCEEDINGS OF THE THIRD SYMPOSIUM ON OPERATING SYSTEMS DESIGN AND IMPLEMENTATION, OSDI - PP. 173–186, 1999. **Anais...** [S.l.: s.n.], 1999.

CHUEN, D. L. K. **Handbook of digital currency, 1st ed.** [S.l.]: Elsevier, 2015.

CROMAN, K. et al. On Scaling Decentralized Blockchains (A position paper). **Springer Link**, [S.l.], 2016.

D. JOHNSON, A. M.; VANSTONE, S. "the elliptic curve digital signature algorithm (ecdsa)". **International Journal of Information Security, vol. 1, no. 1, pp. 36–63, 2001.**, [S.l.], 2001.

DAVID SCHWARTZ NOAH YOUNGS, A. B. **“the ripple protocol consensus algorithm”**. [S.l.]: Ripple Labs Inc, 2014.

DEV, J. A. Bitcoin mining acceleration and performance quantification. **Canadian Conference on Electrical and Computer Engineering**, [S.l.], p. 1–6, 2014.

DIFFIE, W.; HELLMAN, M. “privacy and authentication: an introduction to cryptography”. **IEEE Vol. 67, No. 3, Mar. 1979 pp. 397- 427**, [S.l.], 1979.

DINH, T. T. A. et al. BLOCKBENCH: a framework for analyzing private blockchains. **ACM**, [S.l.], p. 1085–1100, 2017.

ETHEREUM - blockchain app platform. 2018.

EYAL, I. et al. Bitcoin-NG: a scalable blockchain protocol. **13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)**, [S.l.], p. 45–59, oct 2015.

GERVAIS, A. et al. On the Security and Performance of Proof of Work Blockchains. **Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security - CCS’16**, [S.l.], p. 3–16, 2016.

GÖBEL, J. et al. Bitcoin Blockchain Dynamics: the selfish-mine strategy in the presence of propagation delay. **Performance Evaluation**, [S.l.], v. 104, p. 23–41, 2016.

HYPERLEDGER fabric. 2018.

INSTITUTE, N. R. **Survey on blockchain technologies and related services - fy2015 report**. [S.l.]: Japan’s Ministry of Economy, Trade and Industry (METI), 2016.

KAKAVAND, H.; Kost De Sevres, N.; CHILTON, B. The Blockchain Revolution: an analysis of regulation and technology related to distributed ledger technologies. **SSRN Electronic Journal**, [S.l.], 2017.

KARAME, G. O. Two bitcoins at the price of one? double-spending attacks on fast payments in bitcoin. In: IN PROC. OF CONFERENCE ON COMPUTER AND COMMUNICATION SECURITY, 2012. **Anais...** [S.l.: s.n.], 2012.

KWON, J. **“tendermint**: consensus without mining”. [S.l.]: Tendermint, 2014.

L. LAMPORT, R. S.; PEASE, M. "the byzantine generals problem". **ACM Transactions on Programming Languages and Systems (TOPLAS) Vol. 4, No. 3, pp. 382–401**, [S.l.], 1982.

LU K.; SUBRATA, R. Z. A. Towards Decentralized Load Balancing in a Computational Grid Environment. **CHUNG, Y.-c.; MOREIRA, J. (Ed.). Advances in Grid and Pervasive Computing. [S.l.]: Springer Berlin / Heidelberg, 2006. p. 466—477**, [S.l.], 2004.

LU K.; SUBRATA, R. Z. A. A Survey of Load Balancing in Grid Computing. **CIS’04**, [S.l.], 2004.

MAZIèRES, D. **“the stellar consensus protocol**: a federated model for internet-level consensus”. [S.l.]: Stellar Development Foundation, 2014.

MOUGAYAR, W. **The business blockchain**: promise, practice, and application of the next internet technology. [S.l.]: John Wiley & Sons, 2016.

MULTICHAIN - open platform for building blockchains. 2018.

NAKAMOTO, S. Bitcoin: a peer-to-peer electronic cash system. **Www.Bitcoin.Org**, [S.l.], p. 9, 2008.

NATOLI, C.; GRAMOLI, V. The Blockchain Anomaly. **Proceedings - 2016 IEEE 15th International Symposium on Network Computing and Applications, NCA 2016**, [S.l.], p. 310–317, 2016.

OZISIK, A. P. et al. Graphene: a new protocol for block propagation using set reconciliation. **Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)**, [S.l.], v. 10436, p. 420–428, 2017.

PETTICREW, M.; ROBERTS, H. Systematic reviews in the social sciences. **Blackwell Publishing**, [S.l.], 2006.

PILKINGTON, M. Blockchain Technology: principles and applications. **SSRN**, [S.l.], 2015.

PONGNUMKUL, S.; SIRIPANPORNCHANA, C.; THAJCHAYAPONG, S. Performance Analysis of Private Blockchain Platforms in Varying Workloads. In: INTERNATIONAL CONFERENCE ON COMPUTER COMMUNICATION AND NETWORKS (ICCCN), 2017., 2017. **Anais...** IEEE, 2017. p. 1–6.

PRPIC PHD, J. "unpacking blockchains". **Collective Intelligence 2017. NYU, Tandon School of Engineering. June 15-16, 2017.**, [S.l.], 2017.

QICHAO ZHANG ZHUYUN QI, X. L. T. S.-K. L. Research and application of bft algorithms based on the hybrid fault model. In: IEEE INTERNATIONAL CONFERENCE ON HOT INFORMATION-CENTRIC NETWORKING (HOTICN 2018), 2018., 2018. **Proceedings...** [S.l.: s.n.], 2018.

RIZUN, P. R. Subchains: a technique to scale bitcoin and improve the user experience. **Ledger**, San Juan, Puerto Rico, v. 1, 2016.

SANKAR, L. S.; SINDHU, M.; SETHUMADHAVAN, M. Survey of consensus protocols on blockchain applications. In: INTERNATIONAL CONFERENCE ON ADVANCED COMPUTING AND COMMUNICATION SYSTEMS (ICACCS), 2017., 2017. **Anais...** IEEE, 2017. p. 1–5.

SCHERER, M.; ERIKSSON, J. Performance and Scalability of Blockchain Networks and Smart Contracts. **Springer Link**, [S.l.], 2017.

SIKORSKI, J. J.; HAUGHTON, J.; KRAFT, M. Blockchain technology in the chemical industry: machine-to-machine electricity market. **Applied Energy**, [S.l.], v. 195, p. 234–246, 2017.

SIMMONS, G. J. "symmetric and asymmetric encryption". **ACM Computing Surveys (CSUR), 11(4), 305-330**, [S.l.], 1979.

SOMPOLINSKY, Y.; ZOHAR, A. Secure high-rate transaction processing in bitcoin. **Financial Cryptography and Data Security 2015, 19th International Conference**, San Juan, Puerto Rico, Jan. 2015.

SPASOVSKI, J.; EKLUND, P. Proof of Stake Blockchain : performance and scalability for groupware communications. **ACM**, [S.l.], n. September, 2017.

SWAN, M. **Blockchain blueprint for a new economy**. Sebastopol, CA: O'Reilly Media, Inc., 2015.

TASCA, P.; THANABALASINGHAM, T.; TESSONE, C. J. Ontology of Blockchain Technologies. Principles of Identification and Classification. **SSRN**, [S.l.], p. 1–58, 2017.

TSENG L.-Y.; CHIN, Y.-H. W. S.-C. A minimized makespan scheduler with multiple factors for Grid computing systems. **Expert Systems with Applications, [S.l.], v. 36, n. 8, p. 11118 - 11130**, [S.l.], 2009.

TWEETS per second record. 2018.

VISA fact sheet. 2018.

VUKOLIĆ, M. Rethinking Permissioned Blockchains. **Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts - BCC '17**, [S.l.], p. 3–7, 2017.

WANG X.; ZHU, Z. D.-Z. L. S. Multi-cluster Load Balancing Based on Process Migration. In: advanced parallel processing technologies. **Springer Berlin / Heidelberg, 2007. p. 100—110. (Lecture Notes in Computer Science)**, [S.l.], 2007.

XIANG, F. et al. JCLedger: a blockchain based distributed ledger for jointcloud computing. **Proceedings - IEEE 37th International Conference on Distributed Computing Systems Workshops, ICDCSW 2017**, [S.l.], p. 289–293, 2017.

XIWEI XU INGO WEBER, M. S. L. Z.-J. B. L. B. C. P. P. R. "a taxonomy of blockchain-based systems for architecture design". **IEE. 2017 IEEE International Conference on Software Architecture (ICSA). Gothenburg, Sweden**, [S.l.], 2017.

YLI-HUUMO, J. et al. Where is current research on Blockchain technology? - A systematic review. **PLoS ONE**, [S.l.], v. 11, n. 10, p. 1–27, 2016.

ZHENG, Z. et al. Blockchain Challenges and Opportunities: a survey. **International Journal of Web and Grid Services**, [S.l.], n. January, p. 1–24, 2016.

ZHENG, Z. et al. An Overview of Blockchain Technology: architecture, consensus, and future trends. **2017 IEEE International Congress on Big Data (BigData Congress)**, [S.l.], p. 557–564, 2017.