UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Middleware para detecção de anomalias no compartilhamento de conteúdo para sistemas baseados em blockchain

Alef Vinicius Cardoso e Silva

Dissertação de Mestrado do Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional (PPG-CCMC)

ICMC USP
SÃO CARLOS

Alef Vinicius Cardoso e Silva

Middleware para detecção de anomalias no compartilhamento de conteúdo para sistemas baseados em blockchain

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Mestre em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientador: Prof. Dr. Jó Ueyama

USP – São Carlos
Novembro de 2020

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

S586m

Silva, Alef
Middleware para detecção de anomalias no compartilhamento de conteúdo para sistemas baseados em blockchain / Alef Silva; orientador Jó Ueyama. -- São Carlos, 2020.
101 p.

Dissertação (Mestrado - Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2020.

1. Middleware. 2. Anomalia. 3. Blockchain. 4. Compartilhamento. 5. Conteúdo. I. Ueyama, Jó, orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Alef Vinicius Cardoso e Silva

Middleware for detecting anomalies in content sharing for blockchain-based systems

Master dissertation submitted to the Institute of Mathematics and Computer Sciences – ICMC-USP, in partial fulfillment of the requirements for the degree of the Master Program in Computer Science and Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Prof. Dr. Jó Ueyama

USP – São Carlos
November 2020

Este trabalho é dedicado a Deus e aos meus pais que,
desde meu nascimento, sonharam em me tornar uma boa pessoa e um bom profissional.
Em especial, aos pesquisadores e colaboradores do Instituto de Ciências Matemáticas e de Computação (ICMC).

AGRADECIMENTOS

Agradeço primeiramente a Deus por ter me proporcionado a oportunidade de estar aqui realizando o sonho dos meus pais que não tiveram a oportunidade de estudar e conhecer uma instituição de ensino tão incrível como a Universidade de São Paulo. Em agradecimento a todos que me incentivaram a obter conhecimento e nunca desistir, aos professores da Escola Estadual Marina Amarante Ribeiro Vasques Sanches pelo apoio e ensino no período fundamental e médio, aos professores e colaboradores do Instituto Federal de São Paulo (IFSP), pela base de conhecimento concebida na graduação e o incentivo que me foi dado para realizar a pós-graduação. Agradeço, em especial, à Universidade de São Paulo, por toda a infraestrutura que me permitiu desenvolver este trabalho, a todos os colaboradores e, claro, ao Dr. Jó Ueyama, que me acolheu como filho, me ensinou muito mais do que ser um pesquisador, ele me demonstrou como ser um bom profissional da Educação. Esta pesquisa foi financiada em parte pela Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) - Código Financeiro 001.

“Homo possit facere meliora.
Fortis Fortuna Adiuvat.”

RESUMO

SILVA, A. V. C. **Middleware para detecção de anomalias no compartilhamento de conteúdo para sistemas baseados em blockchain**. 2020. 100 p. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2020.

A evolução das plataformas digitais, bem como a geração massiva de dados, possibilitou um cenário mundial orientado por dados. Por exemplo, decisões profissionais, processos físicos, emocionais e produtivos têm sido feitos baseados em dados. Nesta dissertação, apresenta-se o MADCS, que serve para alcançar o compartilhamento de conteúdo sensível e envolver questões de privacidade e segurança, especialmente, ao alterar ou infringir conteúdo confidencial. MADCS é um *middleware* para sistemas baseados em redes descentralizadas, como o Blockchain, para o compartilhamento de conteúdo sensível. Devido ao avanço das redes descentralizadas e às diversas criações de moedas virtuais, o trabalho teve como foco o estudo das tecnologias, sistemas e plataformas que compõem essas criptomoedas para verificar a possibilidade de utilizá-las em outros contextos como o compartilhamento de conteúdo sensível. Apesar dessas plataformas oferecerem vantagens em termos de imutabilidade, privacidade, segurança e confiabilidade, algumas lacunas ainda persistem para tornar tais plataformas mais seguras e confiáveis. Diante disso, o *middleware* MADCS tem o objetivo de detectar anomalias no compartilhamento de conteúdos sensíveis tendo como foco o experimento de verificar o conteúdo de cada transação dentro de uma organização e, a partir de *clusters* criados com base em transações históricas, detectar se existe alguma fora do fluxo de normalidade e que possa distinguir as transações válidas e inválidas. Dessa forma, conseguimos verificar anomalias impostas pelas próprias partes interessadas que visam se beneficiar da rede. Para validar o *middleware*, construímos um conjunto de dados de prescrição médica seguindo o modelo de prescrições médicas genérico, e aplicou-se o algoritmo K-means para identificar anomalias simuladas no *middleware* proposto. O teste de zero conhecimento foi utilizado para a conduta confidencial da etapa de negociação entre os usuários, de modo que, para que uma transação possa ser confirmada, o detector de anomalias deve investigar e aprovar suas informações e, posteriormente, o consenso deverá permitir o envio da transação à estrutura. Os resultados das experiências demonstram uma acurácia e precisão de 75 % e 85 % na identificação de anormalidades no processo de compartilhamento de conteúdo. Realizou-se o estudo de caso com e sem MADCS cujos resultados demonstraram que o uso da blockchain combinado com nosso *middleware* pode contribuir para um compartilhamento mais seguro do conteúdo da rede. Foi verificado que o *middleware* associado ao *ledger* não produz um ruído grande em termos de latência e processamento, levando a crer que a inserção de um mecanismo para aumentar a confiança em sistemas distribuídos não alterará a vazão do sistema de forma significativa.

Palavras-chave: blockchain, anomalia, prescrições médicas, privacidade, compartilhamento de conteúdo sensível.

ABSTRACT

SILVA, A. V. C. **Middleware for detecting anomalies in content sharing for blockchain-based systems**. 2020. 100 p. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2020.

The evolution of digital platforms, as well as the massive generation of data, has enabled a common data-driven scenario. For example, professional decisions, physical, emotional, and productive processes have been made based on data. In this dissertation, we present MADCS, to achieve the sharing of sensitive content and to involve privacy and security issues, especially when altering or infringing confidential content. MADCS is a middleware for systems based on decentralized networks, such as blockchain, for sharing sensitive content. Due to the advancement of decentralized nets and the various creations of virtual currencies, the work focused on studying the technologies, systems, and platforms that make up these cryptocurrencies to verify the possibility of using them in other contexts, such as sharing sensitive content. Although these platforms offer advantages in terms of immutability, privacy, security, and reliability, some gaps persist in making such platforms secure and reliable. Therefore, MADCS middleware aims to detect anomalies in the sharing of sensitive content, focusing on the experiment of verifying the content of each transaction within an organization and, based on clusters created based on in actual sales, detect if there is one outside the normal flow and that can distinguish between valid and invalid transactions. In this way, we were able to verify anomalies imposed by the interested parties themselves aim to benefit from the network. To validate the middleware, we built a medical prescription data set following a generic medical prescription model. We applied the K-means algorithm to identify simulated anomalies in the proposed middleware. The zero-knowledge test was used for the confidential conduct of the negotiation stage between users, so that, for a transaction to be confirmed, the anomaly detector must investigate and approve its information. Subsequently, the consensus must allow the sending of the sale to the structure. The results of the experiments demonstrate the accuracy and precision of 75 % and 85 % in the identification of abnormalities in the content sharing process. We carried out the case study with and without MADCS, and the results demonstrated that the use of blockchain combined with our middleware could contribute to a more secure sharing of the network's content. It was verified that the middleware associated with ledger does not produce a big noise in terms of latency and processing, leading to the belief that the insertion of a mechanism to increase the confidence in distributed systems will not alter the system flow significantly.

Keywords: blockchain, anomaly, medical prescriptions, privacy, sharing sensitive content.

LISTA DE ILUSTRAÇÕES

LISTA DE ALGORITMOS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

CPU	Central Process Unit
DApp	Decentralized Application
DDoS	Distributed Denial of Service
DLTs	Distributed Ledger Technologies
DoS	Denial of Service
GDPR	General Data Protection Regulation
IDS	Intrusion Detection System
MADCS	Middleware for Anomaly Detection and Content Sharing
PBFT	Practical Byzantine Fault Tolerance
PoS	Proof of Stake
PoW	Proof of Work
SVM	Support Vector Machine

SUMÁRIO

CAPÍTULO

1

INTRODUÇÃO

Grandes empresas adotaram a utilização de seus dados históricos e de empresas com a mesma finalidade para entender a tendência, produtos mais vendidos, perfil de usuários, além de outras dezenas de relações que podem ser encontradas nesses dados. Como retorno, essas empresas conseguiram identificar os pontos fracos e fortes dos serviços que prestavam e obtiveram grande vantagem no mercado. A partir do estudo científico de dados, a maioria das empresas passou a adotar o estudo de seus próprios dados para se beneficiar e projetar seu futuro. Algumas empresas foram além e começaram a vender dados do seu comércio local para outras empresas, que associados aos dados das redes sociais contribuíram para um panorama de futuras aplicações na região. Entretanto, o compartilhamento desses dados, de redes sociais e empresas sem autorização dos clientes, trouxe problemas judiciais para as empresas, que precisam se adaptar aos novos regulamentos propostos pelo General Data Protection Regulation (GDPR) para repassarem esses dados sem exporem informações sensíveis (WACHTER; MITTELSTADT; RUSSELL, 2017; TANKARD, 2016).

O Artigo 4, item 12, do GDPR (VOIGT; BUSSCHE, 2017) trata sobre as definições de compartilhamento de dados e, mais especificamente, sobre violação de dados pessoais. A violação de dados pode ser vista como uma violação da segurança que conduz à destruição acidental ou ilegal, perda, alteração, divulgação não autorizada ou acesso a dados pessoais transmitidos, armazenados ou processados de outra forma. Além disso, podemos observar no item 15 do GDPR (VOIGT; BUSSCHE, 2017), que ele está relacionado a dados de saúde e define como dados pessoais aqueles que estão relacionados à saúde física ou mental de uma pessoa singular, incluindo a prestação de serviços de saúde, que revelam informações sobre o seu estado de saúde. De fato, o compartilhamento ou alteração desses dados pode ocasionar grandes escândalos, envolvendo pessoas que confiaram nessas empresas, nas quais compartilham seu conteúdo. Isso trouxe um novo campo de estudo, relacionado ao compartilhamento de conteúdo sensível que seja totalmente confiável para privar a identificação das pessoas relacionadas aos dados.

1.1 Contexto

Uma solução revolucionária que permitiu a garantia da imutabilidade dos dados de seus usuários e ao mesmo tempo transportar seu capital foi o Bitcoin (NAKAMOTO, 2008). O BitCoin foi uma das primeiras moedas virtuais utilizadas para realizar transações pela internet de forma anônima e segura, sem a mediação de um terceiro confiável. Por trás dessa criptomoeda, termo mais popularmente conhecido, existe a tecnologia Blockchain, que é o sistema distribuído e descentralizado que executa as transações. A relação que temos dessa arquitetura Blockchain para criptomoedas e o compartilhamento de conteúdo está no fato dessas plataformas terem a característica de serem imutáveis e seguras, privando dados de seus clientes e usando chaves criptográficas no lugar, dando a oportunidade de transitar moedas sem utilizar identificadores na rede. Apesar dessas plataformas ainda sofrerem com ataques dos próprios usuários da rede, que tentam se beneficiar de alguma brecha nos contratos inteligentes, ou do mecanismo de consenso, ainda se tornam mais promissoras para ter maior êxito em gerenciar dados que não podem ser alterados.

A utilização de sistemas centralizados para criação de aplicações que exigem alto nível de segurança e integridade como os bancos, sempre foram bem vistos. Porém, desde que os clientes se perguntam como seus dados são protegidos ou utilizados pela empresa, uma caixa preta surge como resposta e não se sabe o que acontece com os dados dos clientes. Esses dados podem ser classificados como dados sensíveis (nome, RG, CPF, saldo, gênero, etc.), que precisam ser protegidos contra ataques e utilizações indevidas por parte da empresa e até externos, quando fornecido. Quando se toma o termo dado sensível, logo é possível visualizar outras aplicações, como economia (CHENG *et al.*, 2019; NASIR *et al.*, 2019; Wu *et al.*, 2019), saúde (HÖLBL *et al.*, 2018; Mettler, 2016; RANDALL; GOEL; ABUJAMRA, 2017), multimédia (Bhowmik; Feng, 2017; Fotiou; Polyzos, 2016; ICHIKAWA; KASHIYAMA; UENO, 2017) e sistemas para carros inteligentes (SHARMA; MOON; PARK, 2017; SWAN, 2015; Yuan; Wang, 2016), entre outros como Iot, Smart homes e Smart cities, que produzem uma grande quantidade de dados. Importante destacar que muitos desses dados são informações confidenciais e privadas de usuários, que não permitem o compartilhamento.

A descentralização desses tipos de aplicações citadas acima seria inevitável, pois muitas dessas aplicações estão interligadas a outros sistemas e podem interagir entre si. Uma tecnologia que é um exemplo prático de descentralização de um desses serviços é o Blockchain (NAKAMOTO, 2008). O blockchain é uma tecnologia similar ao livro razão, mas com o aspecto de ser distribuído. Ele foi introduzido no contexto das moedas virtuais ao qual se aplica o nome de *ledger*. Nesse *ledger* são registradas as transações financeiras que os usuários podem realizar na rede sem a intermediação de um terceiro confiável. Na primeira moeda virtual utilizada, o Bitcoin (ANTONOPOULOS, 2014), os dados são públicos para toda rede, porém a privacidade do usuário é mantida, a ponto de um usuário, mesmo não conhecendo o outro, realizarem transações de forma segura. Claro que ao longo do tempo, foi notado que a utilização das chaves

criptográficas no blockchain, que permitem a privacidade dos usuários, ainda têm uma brecha quanto à privacidade, pois as chaves podem ser interligadas e os usuários ou uma entidade maliciosa podem saber que um usuário com a chave Y realizou X transações e com N usuários diferentes. Ainda sim, esse sistema garante a imutabilidade das transações e a integridade dos dados armazenados na cadeia (FENG *et al.*, 2019).

Para utilizar essa plataforma e prover um compartilhamento de conteúdo confiável, criou-se o projeto Middleware for Anomaly Detection and Content Sharing (MADCS) (*Middleware for Anomaly Detection and Content Sharing*), um *middleware* que intercepta a entrada entre a organização e o *ledger*, para verificar o conteúdo que está chegando com base em transações anteriormente salvas. Dessa forma, podemos utilizar todo o aspecto da rede e aprimorar a forma de inspecionar os dados, verificando se houve alguma alteração fora dos padrões salvos.

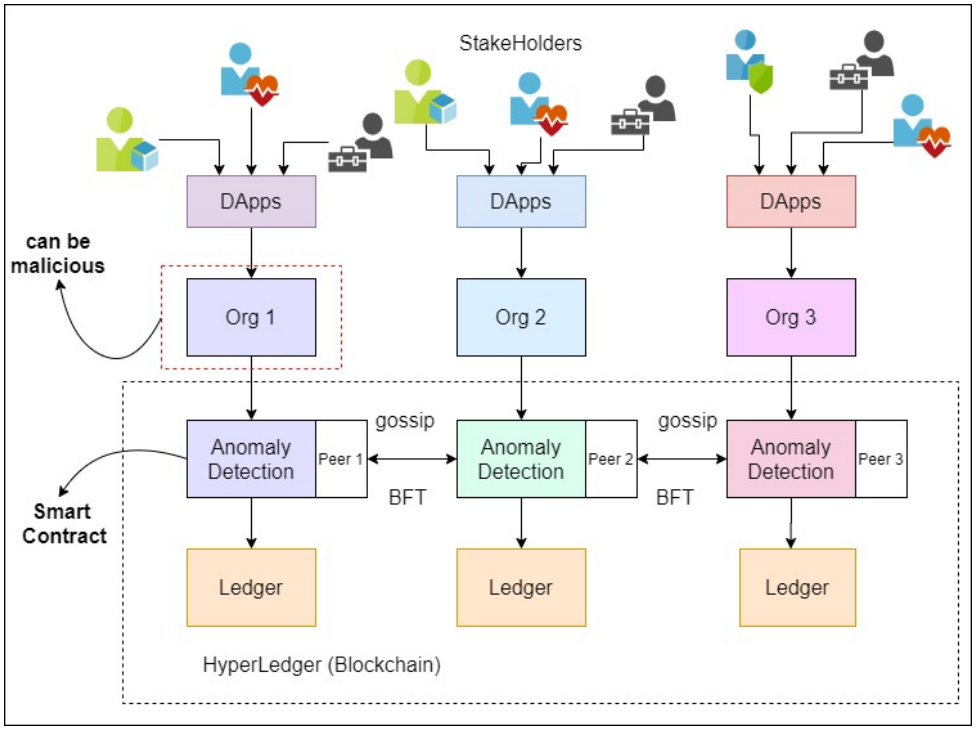
O MADCS possui uma arquitetura baseada em seis camadas (entrada, saída, pré-processamento, análise de dados, negócios e blockchain), com as camadas de entrada, saída e blockchain provenientes da plataforma que foi utilizada neste caso o *HyperLedger* da IBM (COMPOSER, 2019). Conseguimos detectar alterações maliciosas na rede pelos próprios usuários da organização, pois a estrutura do MADCS se beneficia dos dados da plataforma e, por meio da criação de *clusters* que utilizam o algoritmo k-means(KRISHNA; MURTY, 1999), assimilam as semelhanças entre os dados. Para realizar esse teste, foi utilizado um *dataset* empírico, com base nos dados obtidos da plataforma *Kaggle* (ANALYTICS, 2018) de prescrições médicas. Esses dados foram alterados para a criação de prescrições médicas de forma a simular um ambiente de várias organizações que compartilham um ambiente virtual único, armazenado em um *ledger*, o que traz o benefício de tornar os dados acessíveis por qualquer instituição dentro da organização e de forma segura. Assim, as permissões podem ser dadas conforme o nível de acesso que cada organização pode ter e integrar a dispensação farmacêutica de medicamentos com prescrições emitidas em qualquer estado, unificando e simplificando a forma de gerar e dispensar receitas médicas. Com esse estudo, simularam-se esses processos criando *peers* com diferentes papéis e realizando operações de criação, inserção, e tentativas de alterações na prescrição. Como resposta, o *middleware* conseguiu detectar até 85 % das alterações impróprias que tentou-se inserir na rede, obtendo a mesma precisão.

1.2 Problema

Para chegar a esse cenário de simulação e no *middleware* proposto, um problema geral foi exemplificado na Figura 1: a organização X deseja gerenciar seus dados de prescrições médicas que estão distribuídas em várias organizações independentes (Hospitais). Cada organização possui partes interessadas com papéis específicos que exercem funções diferentes na rede. Além das organizações que prescrevem medicamentos, temos as organizações que recebem e vendem as prescrições. Todas essas organizações compartilham um mesmo *ledger* que armazena as

prescrições com os dados dos clientes. Eventualmente, uma dessas organizações pode se tornar maliciosa na rede e tentar explorar erros ou inserir conteúdo inconsistente de seus *stakeholders* para se beneficiar disso.

Figura 1 – Descrição do problema.



Fonte. Próprio autor.

Para solucionar este problema, um *middleware* foi desenvolvido para atuar com o contrato inteligente dentro do Blockchain, e a plataforma *HyperLedger* foi a base para a implementação do modelo proposto de dispensação de prescrições. Então, o *middleware* recebe os dados das organizações, avalia o histórico de transações, cria os *clusters* baseados nas transações anteriores de modo a identificar as similaridades dos dados introduzidos na rede. Assim, antes de passar para o contrato, o dado é analisado para identificar anomalias.

Vale destacar que em uma rede Blockchain, podemos construir sistemas para diferentes organizações, que compartilham o mesmo livro. Cada organização pode implementar seu próprio Decentralized Application (DApp) (Aplicação Descentralizada da organização que interage com a rede) para que as partes interessadas conduzam transações na rede. À medida que cada organização gerencia sua entrada de dados, é possível que, eventualmente, uma dessas partes se torne maliciosa, tentando, de alguma forma, fornecer informações inválidas. No caso em questão, considera-se uma anomalia como a informação que apresenta um padrão diferente das informações do histórico contábil.

O detector de anomalias é baseado em transações históricas, procurando a semelhança entre as transações e separando-as em *clusters*. Quando uma nova transação é transmitida pela

organização, o detector avalia o grau de similaridade da transação e pode bloqueá-la, se sua similaridade estiver distante ou fora dos *clusters* criados. Essas informações são propagadas através do protocolo de Fofoca que envia os dados para os demais *peers* e, desta forma, os detectores saberão quando uma organização está enviando informações maliciosas. Após uma avaliação positiva da transação, o Detector permite verificar com o contrato e o fluxo prossegue normalmente. Caso contrário, a organização pode ser notificada sobre o que aconteceu e avaliar o incidente, sabendo que está perdendo a confiabilidade que possui na rede.

1.3 Questões de Pesquisa

Dado o contexto e o problema, este trabalho teve a intenção de investigar a tecnologia Blockchain para o compartilhamento de conteúdo por meio da verificação de possíveis alterações indevidas, para melhorar a confiabilidade dessas aplicações. Com isso, explorou-se:

1. Como a rede Blockchain contribui para garantir privacidade e confiabilidade no compartilhamento de conteúdo confidencial?
2. Como identificar anomalias no compartilhamento de conteúdo antes de se registrar no *ledger* da rede descentralizada?

Para responder às perguntas de pesquisa utilizou-se o MADCS: um *middleware* baseado em sistemas por trás da recente tecnologia Blockchain 3.0, que implementa contratos inteligentes e uma camada de análise para verificar cada dado antes de ser enviado para contratos inteligentes e a rede blockchain para fornecer análise de dados no compartilhamento de conteúdo confidencial. Para validar o *middleware*: (i) simulou-se uma rede blockchain, (ii) construiu-se um banco de dados de prescrição, (iii) conduziu-se um estudo de caso usando o banco de dados criado e (iv) comparou-se o *middleware* com o blockchain básico (sem o MADCS).

1.4 Contribuições

Diante do problema descrito e as soluções parciais dos trabalhos relacionados, este documento tem como objetivo contribuir para os seguintes itens: 1) Um *middleware* para compartilhamento de conteúdo com base na tecnologia Blockchain e 2) um *dataset* para testes de compartilhamento de conteúdo.

Uma das premissas para se alcançar a confiabilidade em uma rede blockchain é reduzir o número de transações duplicadas ou se, de alguma forma, inseridas de forma equivocada na cadeia, o que levará a um erro na criação dos próximos blocos. Com o MADCS, ao analisar as transações antes de serem inseridas na cadeia, o número de transações errôneas é consideravelmente reduzido.

1.5 Motivação Social

Para realizar os testes de validação do modelo foi proposto um projeto para atender a criação e a dispensação de prescrições médicas. Este estudo de caso traz um exemplo de contexto social sobre o qual se pode unificar diferentes organizações (hospitais, clínicas e farmácias) em um sistema para criar receitas e dispensar medicamentos, tudo dentro de regras pre-estabelecidas em contrato. O diferencial desse modelo é deixar o paciente como proprietário dos seus dados. Isso permite que ele compartilhe uma informação ou dê permissão apenas às instituições que achar interessante, como hospitais, pesquisadores e empresas interessadas. A proposta reduz a burocracia de criar prescrições médicas para pessoas que tomam medicamentos periodicamente, podendo ser estabelecido um tempo de vida para as prescrições e quantidade de vezes que pode ser utilizada.

Apesar de não ser o objeto base de estudo, o modelo desenvolvido permite que uma moeda de incentivo possa ser introduzida na rede e agregar instituições privadas, de modo que estas possam oferecer descontos em medicamentos conforme o usuário utiliza os serviços médicos dessas instituições, diminuindo a pressão em hospitais públicos e beneficiando os pacientes na compras de medicamento, além de aumentar a demanda dos hospitais privados. Porém, o foco deste trabalho não está na constituição e arquitetura da moeda, mas na forma de compartilhar os dados desses usuários de forma confiável.

1.6 Motivação Científica

As tecnologias blockchain têm ganhado espaço no mercado pela sua capacidade de tratar os dados de forma distribuída e também por prover um retorno aos usuários que participam dela. Além disso, a rede possui a possibilidade de usabilidade em outras áreas, como Internet das Coisas. Esse outro contexto exige que as redes descentralizadas sejam remodeladas para não aumentar a latência de aplicações que precisam executar e validar as transações em um curto período de tempo. Além da latência, outras questões como o contrato inteligente, questões de segurança e rastreabilidade ainda estão em aberto e podem ser melhoradas.

1.7 Objetivo Geral

Projetar e avaliar um *middleware* que possa detectar anomalias dentro do contexto de uma rede distribuída, onde as organizações podem assumir um papel de intruso e tentar alterar dados para se beneficiar de alguma forma sobre um determinado contrato.

1.8 Objetivos Específicos

1. Explorar a tecnologia blockchain com contratos inteligentes para desenvolver um modelo que possa agregar ao *middleware*.

2. Projetar o *middleware* para incluir os componentes necessários no processo de detecção de anomalias dentro do contexto e dos dados fornecidos pela rede Blockchain.

3. Avaliar a rede distribuída com e sem o *middleware* para mensurar o desempenho da rede e desempenho de cada *peer*, avaliando a Latência, *overhead* de Central Process Unit (CPU) e Memória.

1.9 Estruturação do Trabalho

Além deste capítulo, esta dissertação de mestrado está organizada da seguinte maneira:

- No Capítulo 2 são apresentados os principais conceitos relacionados à tecnologia blockchain, contratos inteligentes e algoritmos de detecção de anomalias utilizados para explorar soluções ao problema proposto.

- No Capítulo 3 são apresentados os trabalhos mais relevantes ao tema de pesquisa, com proximidade na resolução para detectar anomalias em um contexto de redes.

- No Capítulo 4 são apresentadas a arquitetura do *middleware* proposto e a explicação de cada camada incorporada ao *middleware* bem como a função que cada uma exerce.

- No capítulo 5 é apresentado o estudo de caso com o modelo proposto em um contexto de prescrições médicas e dispensação farmacêutica.

- No Capítulo 6 são apresentados os experimentos para verificar a eficiência do *middleware* na detecção de anomalias. Também é avaliado o sistema Blockchain como um todo e a alteração de performance com a inserção do *middleware*.

- No capítulo 7 é apresentado o resultado final e as discussões sobre esses resultados. Uma avaliação geral do projeto é feita para identificar a sua utilização em um ambiente comercial. Com essa avaliação pode-se perceber quais impactos o MADCS poderia trazer e quais os pontos positivos de se incorporar um sistema ao Blockchain.

- Finalmente, o capítulo 8 são apresentadas as conclusões sobre o trabalho apresentado, limitações e trabalhos futuros além das referências citadas nos capítulos anteriores necessárias para o entendimento e conceitualização do projeto realizado.

CAPÍTULO

2

REFERENCIAL TEÓRICO

Esta seção tem por objetivo apresentar todos os conceitos, técnicas e aplicações que foram necessários para elaborar e construir este trabalho.

2.1 Sistemas Distribuídos

A tecnologia ganhou espaço no mundo e a internet foi o maior marco desse crescimento em larga escala. Com isso, diversas arquiteturas de sistemas foram projetadas para que os usuários pudessem trocar informações, acessar páginas web, acessar arquivos distantes geograficamente e outras diversas aplicações que estão disponíveis na internet. Essas arquiteturas, que serão comentadas no texto a seguir, tiveram melhor desempenho com o crescimento de usuários com dispositivos de borda na rede, como, por exemplo, celulares e computadores pessoais. Esses dispositivos com capacidade de processamento alto, permitiu a utilização de sistemas distribuídos para compartilhar e acelerar processos na internet (COULOURIS *et al.*, 2013).

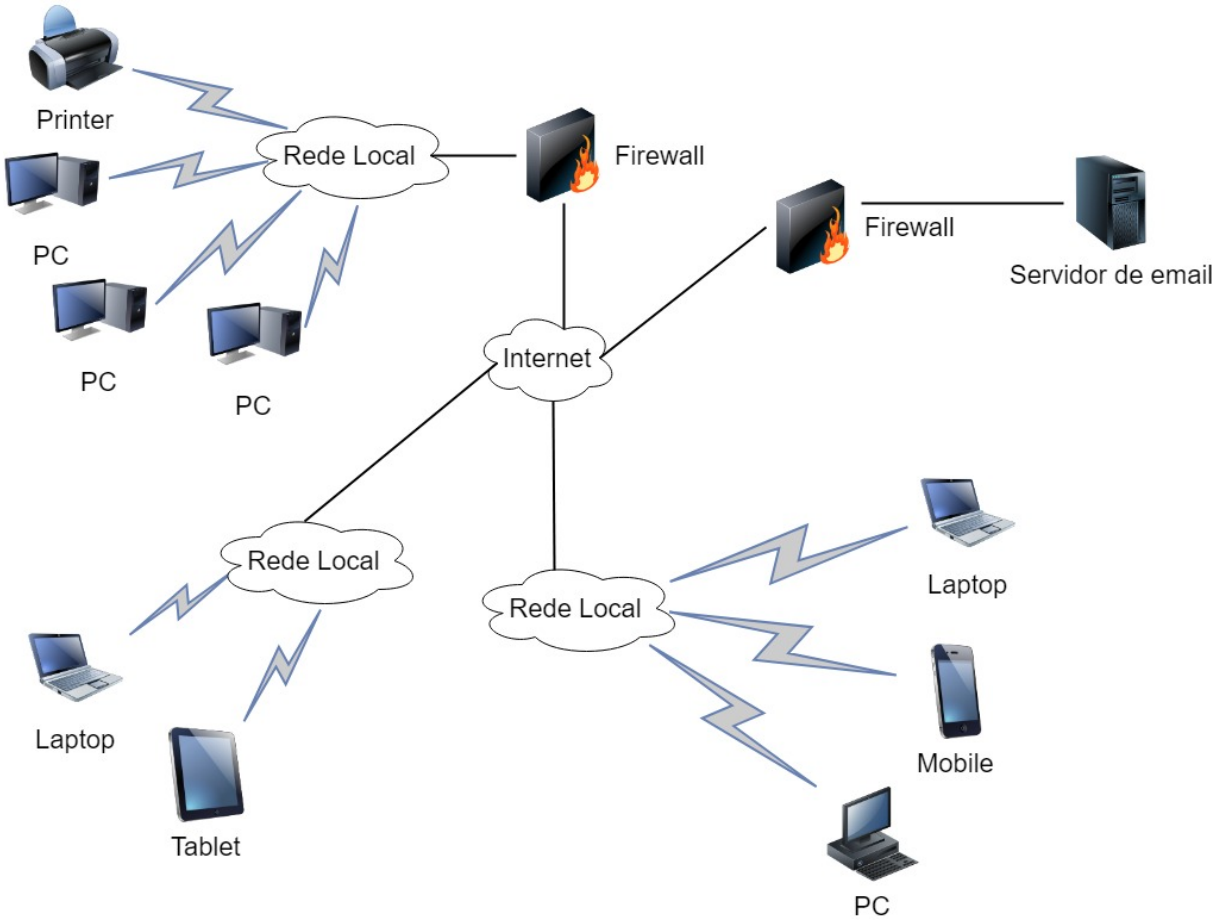
Um sistema distribuído trata-se de componentes interligados em rede, que se comunicam e coordenam suas ações através de mensagens. Esse conceito tem sido amplamente utilizado pelos inúmeros serviços que os usuários da internet concede a outros usuários. Basicamente, esses serviços se concentram no compartilhamento de recursos entre usuários, que podem estar geograficamente distantes. Dependendo da arquitetura, esses recursos podem ser gerenciados por um servidor central ou simplesmente gerenciado por cada usuário da rede, seguindo um protocolo de comunicação, que coordena como a comunicação deve ser feita e que tipo de dado pode ser transmitido (COULOURIS *et al.*, 2013).

2.1.1 Caracterização e Exemplos

A maior motivação para criar sistemas distribuídos está na facilidade de se compartilhar recursos de forma virtual na internet. O primeiro grande exemplo de sistema distribuído é a

própria Internet. A internet é um conjunto de dispositivos heterogêneos interligados, que trocam serviços e informações. A Figura 2 mostra um conjunto de redes locais de empresas (rede interna), que se conecta a rede de internet. As redes locais são gerenciadas por organizações, que podem prover serviços para os usuários, como o serviço de endereço eletrônico. Esse serviço, pode ser compartilhado com outras redes quando conectado por um enlace (backbone), que permite a troca de dados entre as redes locais (COULOURIS *et al.*, 2013).

Figura 2 – Exemplo da internet e intranet.



Fonte (COULOURIS *et al.*, 2013).

A partir desta definição acima, algumas situações podem concorrer na comunicação entre os elementos da rede:

1. Concorrência: Em redes de computadores, os programas em execução estão quase sempre concorrendo para permanecerem ativos. Um servidor, por exemplo, provê um site web para vários usuários, que acessam o site. Esse servidor chegará em um ponto máximo que pode receber de usuários e começar a banir alguns que estão inativos e prover acesso a outros. Internamente, o processo do site web pode estar concorrendo com outros processos e o próprio sistema operacional da máquina. Porém, a capacidade física do sistema pode ser

aumentada com a inserção de mais computadores e paralelizar o processo (COULOURIS *et al.*, 2013).

2. Inexistência de um Relógio Global: quando os processos precisam cooperar entre si. Para isso, deve existir um mecanismo para sincronizar as ações, pois o processo de comunicação entre os sistemas é feito por mensagens. Entretanto, estabelecer uma sincronia entre os computadores é um ponto de difícil resolução, pois cada computador e sistema pode estar em um tempo diferente do outro e em estados diferentes (COULOURIS *et al.*, 2013).

3. Falhas Independentes: Todo sistema de computador pode falhar, cabe aos desenvolvedores estabelecerem normas e regras para contornar tais situações. Uma falha pode resultar no isolamento de um computador ou interrupção de um processo de votação sem solá-lo permanentemente da rede. Tendo sido bem projetada, a rede continuará em funcionamento mesmo sem esse elemento que falhou (COULOURIS *et al.*, 2013).

A função do *firewall*, na Figura 2, é de proteger a intranet com possíveis ataques ou mensagens indevidas vindas da rede externa. Esse software filtra as mensagens e autoriza apenas as mensagens que têm acesso ao serviço solicitado. A importância de se ter mecanismo de segurança se dá ao fato dos tipos de arquiteturas que têm na internet para prover serviços, cliente-servidor e as redes *peer-to-peer*.

2.1.2 Arquiteturas de Sistemas Distribuídos

A arquitetura cliente-servidor é a mais comum, seu maior exemplo é um sistema web. Por exemplo: um computador, denominado servidor, armazena uma página que contém notícias diárias sobre a cidade de São Paulo, e os clientes são as pessoas que acessam o site web por meio da url fornecida para ver as notícias postadas. Neste caso o cliente apenas consome um serviço que é gerenciado pelo servidor web.

Já os sistemas peer-to-peer apresentam uma arquitetura de construção mais parecida com sistemas distribuídos. Os recursos computacionais dessa rede são compartilhados para que os hosts participantes possam compartilhar uma informação ou realizar um serviço monetário.

2.1.3 Segurança

A segurança em sistemas distribuídos é crucial e ao mesmo tempo complexo de gerenciar, pois todos os participantes da rede precisam assumir um papel de honestidade e manter a rede no melhor estado possível. Ataques em redes podem acontecer de diferentes formas, como, por exemplo, negação de serviço, falsificação, mascaramento e intromissão. Levando em consideração o cenário de compartilhamento de recursos, o sistema deve garantir a proteção desses recursos compartilhados, da etapa de armazenamento até o compartilhamento. Para evitar

problemas, principalmente com vazamento de informações e falsificação, é necessário projetar o sistema com técnicas de segurança previamente inseridas (COULOURIS *et al.*, 2013).

Uma vez reunidas as possibilidades de ataques que um sistema pode ter, os mecanismos de segurança podem ser empregados para controlar e neutralizar as ameaças. Para isso, existem várias técnicas para tornarem seguros sistemas e aplicações distribuídas. Dentre elas, podem-se elencar as seguintes técnicas a seguir.

2.1.3.1 Algoritmos de Criptografia

Criptografia é o processo de codificar uma mensagem de maneira a ocultar o conteúdo. Alguns algoritmos utilizam um segredo para codificar e decodificar a mensagem, chamado chave criptográfica. A chave criptográfica é um parâmetro de entrada mais o dado cifrado que é utilizado na criptografia, e, dessa forma, a criptografia não poderá ser revertida sem a chave (COULOURIS *et al.*, 2013).

Existem duas classes de algoritmos de criptografia, simétrica e assimétrica. A primeira classe utiliza apenas uma chave, que precisa estar com o emissor e destinatário da mensagem. Caso essa chave seja descoberta por um terceiro, o conteúdo cifrado pode ser revelado. A segunda classe utiliza um par de chaves, pública e privada. A chave pública é utilizada para cifrar a mensagem e pode ficar disponível na Web. Porém, apenas quem possuir o par de chaves que completa aquela chave pública, pode decifrar a mensagem (COULOURIS *et al.*, 2013)m.

A criptografia pode ser utilizada em três partes do sistema: na autenticação, integridade e assinaturas digitais. Para exemplificar cada uma, o seguinte cenário é proposto com base no contexto de experimento realizado para avaliar o modelo do *middleware* proposto.

Patrícia é médica de uma clínica particular. Ela participa do sistema nacional de prescrições médicas, que regulamenta a prescrição e dispensação de medicamentos. Maria é paciente de Patrícia e precisa de uma receita para retirar um medicamento de alto risco que precisa de autorização. E, por fim, Bob é o farmacêutico que normalmente atende Maria quando ela vem à farmácia. Para que Patrícia entre no sistema, lhe é solicitado que insira seu número da carteira de médica e uma senha. A autenticação, nesse caso, permite que o sistema dê acesso apenas aos usuários cadastrados. Patrícia acessa o sistema, realiza a consulta de Maria e prescreve o medicamento necessário. Para proteger os dados de Maria, conforme o regulamento do sistema, a prescrição é cifrada com a chave pública que foi gerada para Maria. Assim, a chave privada de Maria pode ficar com ela, para que possa gerenciar seus próprios dados. Isso de certa forma, garante a integridade e mantém os dados em segredo, sobre qualquer vazamento da informação na internet. Além da cifragem, uma assinatura é gerada para ser entregue ao farmacêutico, confirmando que a receita não foi modificada. A assinatura digital é um vínculo irreversível, que torna possível verificar se um conteúdo é uma cópia inalterada do original. Assim, o farmacêutico poderá verificar a validade da prescrição do paciente no sistema.

A cifragem é o processo de aplicar uma transformação na mensagem que está com o texto e ter como resultado um texto não compreensível. Para que o destinatário possa entender a mensagem, será necessário conhecer a regra inversa da criptografia e reverter o processo, para que a mensagem torne o texto puro e legível novamente. Normalmente, utiliza-se uma chave ou um par de chaves para realizar a criptografia e descriptografia do texto (COULOURIS *et al.*, 2013).

2.1.3.2 Algoritmos Simétricos

A criptografia simétrica, para realizar o processo de cifragem da mensagem, utiliza apenas uma chave criptográfica. Assim, a comunicação entre os usuários pode permanecer ativa e entre vários usuários simultaneamente. A segurança do algoritmo pode ser comprometida caso a chave compartilhada pelos usuários seja, de alguma forma, obtida por outro usuário. Além disso, a segurança do algoritmo de criptografia pode variar conforme o tamanho (bits) da chave utilizada (COULOURIS *et al.*, 2013).

2.1.3.3 Algoritmos Assimétricos

A criptografia de chave assimétrica utiliza duas chaves criptográficas, uma pública e uma privada. A chave pública é disponibilizada e usada para criptografar a mensagem. Apenas o usuário, que possuir a chave privada complementar à chave pública utilizada, poderá decifrar e ver a mensagem (COULOURIS *et al.*, 2013).

2.1.3.4 Assinaturas Digitais

A assinatura digital simula a função das assinaturas convencionais. Com isso, pode-se verificar com as chaves se o documento ou recurso é uma cópia inalterada que foi produzida pelo signatário. Essa técnica pode ser utilizada compactando um resumo da mensagem (método *digest*), e realizando a cifragem, com uma chave conhecida apenas pelo signatário. Normalmente, utiliza-se a criptografia de chave pública para realizar esta técnica (COULOURIS *et al.*, 2013). O remetente da mensagem gera uma assinatura da chave privada e o destinatário pode decifrar utilizando a chave pública correspondente. Para explicar melhor a utilização da assinatura digital, é apresentado a seguir o seguinte cenário exemplo:

Um usuário A deseja enviar um arquivo texto para o usuário B, mas antes, precisa assinar o documento D, para que o usuário B possa verificar se este não foi alterado antes de B acessar o conteúdo. Primeiramente, A faz um resumo do documento de comprimento fixo, $digest(D)$. O usuário A, cifra o conteúdo resumido do arquivo com a chave privada e anexa a D. $digest(D)K_Apriv$. Dessa forma, o arquivo ficará disponível para os usuário que tiverem acesso ao conteúdo e à chave pública correspondente. O usuário B obtém o documento assinado, extrai D e calcula $digest(D)$. Se o resultado obtido por B ao utilizar a função $digest(D)$ com a chave

pública de A for igual à assinatura anexada no documento, a assinatura é válida e o documento não foi alterado (COULOURIS *et al.*, 2013).

2.1.4 Sistema Peer-to-Peer

O compartilhamento de recursos é um dos principais tópicos de interesse em sistemas distribuídos, e uma das arquiteturas mais utilizadas são as redes *peer-to-peer*. A rede *peer-to-peer* permite o compartilhamento de dados e recursos em larga escala, eliminando qualquer intermediário ou servidor gerenciado. O objetivo maior dessa rede é suportar serviços e aplicativos distribuídos úteis, usando dados e recursos computacionais disponíveis nos computadores pessoais e estações de trabalho. Isso acontece graças ao aumento do desempenho dos computadores pessoais, que estão no mesmo patamar dos servidores (COULOURIS *et al.*, 2013). Para entender melhor como a rede funciona, são elencadas algumas características importantes:

1. Cada usuário da rede contribui com recursos para o sistema.
2. Independente do recurso que será compartilhado, todos os nós da rede *peer-to-peer* têm a mesma capacidade e responsabilidade funcional.
3. Não existem sistemas centrais de gerenciamento.
4. É necessário existir um meio que possa tornar os usuários anônimos o mínimo possível.
5. O algoritmo de consenso precisa equilibrar a carga de trabalho entre os *hosts* para não prejudicar os usuários da rede.

Este tipo de sistema passou por algumas fases, e na terceira geração foram desenvolvidos vários *middlewares* de exemplos. Um *middleware* simplifica a construção de serviços implementados em uma rede distribuída. O middleware *peer-to-peer* facilita a comunicação dos *host* para compartilhamento de recursos independente do local que está armazenado, além de facilitar processos de exclusão e inserção de arquivos na rede. Dentre os requisitos que esse sistema precisa ter, a escalabilidade e segurança são fatores de grande importância para que a rede funcione de forma correta (COULOURIS *et al.*, 2013).

Em sistemas distribuídos, é notável perceber, que uma das maiores dificuldades para realizar o gerenciamento dos recursos compartilhados é ter o comprometimento dos participantes da rede em gerenciar de forma correta seus sistemas locais. Sem isso, os processos defeituosos podem comprometer a confiabilidade do sistema e dos recursos que estão sendo compartilhados. Para que isso não se torne um problema, é necessário ter um consenso distribuído entre os pares da rede, para evitar alguns problemas decorrentes a falhas em sistemas individuais. Esse consenso pode ter um líder para gerenciar o processo de validação dos recursos compartilhados, ou pode ser votado por todos os pares da rede dependendo de como foi projetada(COULOURIS *et al.*, 2013).

Os protocolos de consenso são necessários para obter uma troca de informações confiáveis entre os elementos que compõem a rede. Um protocolo trata-se de um conjunto de regras lógicas do processo físico de comunicação. Com esse protocolo, pode-se padronizar aspectos sobre os elementos que serão transacionados, convenções para se estabelecer na rede e quais os caminhos de comunicação (COULOURIS *et al.*, 2013).

Padronização do elemento, significa definir a unidade (informação, *token*, ativo), que será trocada entre os elementos da rede. Um exemplo disso, são as cadeias de caracteres que são transmitidas pelo protocolo de comunicação e se tornam mensagens para um usuário final (COULOURIS *et al.*, 2013).

As convenções estão relacionadas às regras que vão direcionar a forma de representar o dado na rede. Também estão relacionadas a velocidade de transmissão, a sequência de mensagens e as diretrizes para manter a comunicação. Os caminhos da comunicação incluem a criação de um meio confiável e eficiente para transmitir a informação com as características necessárias do ponto de vista da aplicação (COULOURIS *et al.*, 2013).

2.2 Tecnologias de Ledger Distribuídos (DLTs)

A arquitetura *peer-to-peer*, mencionada anteriormente, permitiu o desenvolvimento de diversos sistemas distribuídos, e, entre eles, sistemas com armazenamento distribuído e sincronizado. Atualmente o sistema mais falado é o Blockchain, porém o conceito de livro-distribuído (*ledger*) compreende, muito antes, os universos da Distributed Ledger Technologies (DLTs), assim é possível dizer que o Blockchain é um tipo de DLT, mas com funções e especifidades únicas que o tornam habitualmente incorporado ao meio financeiro e em larga escala. Porém, a dimensionalidade que as DLTs trouxeram para aumentar a vontade de investidores e organizações de usarem sistema distribuídos é um ponto importante para conceitualizar esse tema entender um pouco melhor sobre o surgimento de sistemas mais robustos como o que é utilizado neste projeto.

As DLTs permitem que seus usuários, dentro de uma organização ou entre organizações, armazenem e acessem informações relacionadas a um determinado conjunto de ativos e seus proprietários em um banco de dados compartilhado de transações ou contas-saldos. Essas informações são distribuídas entre os usuários que podem usá-las para liquidar suas transferências de, por exemplo, contratos mobiliários e dinheiro, sem a necessidade de confiar em sistema de validação central. Essa tecnologia trouxe uma mudança progressiva de negociações com entrega física e documentação feita em papel, para um sistema de transferências de livros em bancos de dados digitais. O que permanece inalterada é a necessidade de manter um registro incorruptível autoritário das explorações por infra-estruturas específicas do mercado financeiro e por intermediários envolvidos no processo de liquidação atualizarem seus bancos de dados individuais, comunicando-se com as outras instituições envolvidas, nos diferentes níveis de

pós-negociação, para poder refletir as alterações feitas nos registros uns dos outros. O alto custo desse tipo de processo de consenso levou muitos participantes do mercado a considerar os livros distribuídos como uma alternativa aos sistemas de validação central (PINNA; RUTTENBERG, 2016).

O compartilhamento um banco de dados sem um sistema de validação central pode criar dificuldades quando diferentes usuários e organizações têm incentivos conflitantes. Devido à latência da comunicação via rede, um usuário mal-intencionado pode creditar para diferentes usuários o seu ativo em saldo, o que levará aos negociantes acreditarem, naquele momento, que a transação terá sucesso. Isso pode levar a um problema maior mais pra frente quando comprovado que existem duplicidades nas transações e falta de saldo para cumpri-las. Um resultado igualmente indesejável também pode surgir quando a disfunção na entrada de informações por um usuário de boa-fé, na rede, resulta em informações inconsistentes, que são registradas nas cópias do razão e mantidas por os vários usuários. Antes de todas as cópias distribuídas do razão serem reconciliadas, o beneficiário de uma transação pode executar o que erroneamente parece ser uma entrega versus pagamento. As DLTs permitem que seus usuários cheguem a um consenso em uma versão específica do razão distribuída, em particular, na ordem sequencial das transações. Isso significa que não pode haver nenhuma dúvida sobre as respectivas participações dos usuários. Existe um conjunto de soluções criptográficas e incentivos econômicos que são combinadas para evitar atualizações ilícitas e reconciliar discrepâncias. O livro produzido pode, portanto, ser considerado autoritário, embora seu gerenciamento seja compartilhado entre usuários com incentivos conflitantes (PINNA; RUTTENBERG, 2016).

As DLTs possuem espaço para serem exploradas em diversos contexto, porém, após pesquisas sobre o uso do DLT, concluiu-se que ele é adequado para ser explorado em dois contextos chaves que justificam o seu uso:

- quando não há “confiança” no elemento central (por exemplo, quando não há confiança no hospital em um ambiente de saúde); e por isso faz-se necessário usar uma arquitetura distribuída e não centralizada para armazenar e manter os dados.

- quando há várias partes interessadas, isto é, múltiplos *stakeholders* que podem ter interesses distintos, onde um pode lucrar e outro sofrer prejuízos. Um cenário financeiro é um exemplo deste cenário, no qual temos um banco, um cliente e um órgão regulamentador (p. ex., o Banco Central).

2.3 Blockchain

A seguir é apresentada uma introdução sobre a tecnologia blockchain, tipos de blockchains e suas aplicações. Também é apresentado um modelo de estruturação dos contratos

inteligentes, bem como são implementados no blockchain, inserindo um exemplo para cada tipo de blockchain.

2.3.1 Conceito

Blockchain é uma tecnologia baseada na arquitetura de redes ponto-a-ponto, que possibilita realizar transações de forma distribuída (NAKAMOTO, 2008). Possui grande potencial de mercado, pois elimina o intermediário nas transações e reduz as taxas de custo nas transações. Essa tecnologia possui uma arquitetura completamente nova para os negócios, formando uma nova geração de aplicativos transacionais, que estabelecem confiança e transparência, simplificando os processos de negócios (Nath, 2016). As transações são organizadas e armazenadas em blocos, começando com o bloco inicial chamado Gênesis, e, a partir desse bloco, um *hash* com os valores do bloco é inserido em um campo no próximo bloco, formando uma cadeia imutável.

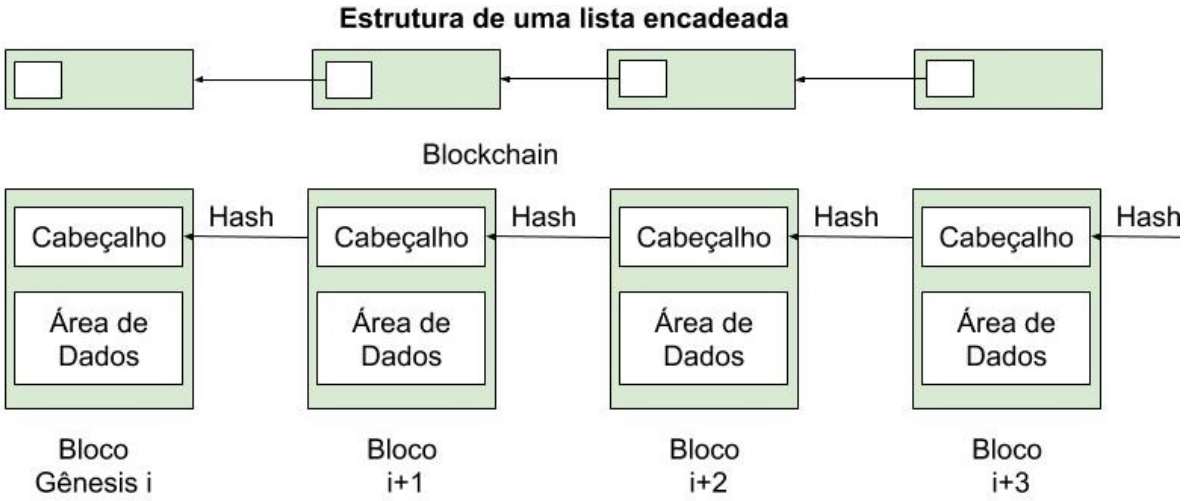
A rede blockchain teve seu primeiro exemplo com a moeda virtual Bitcoin, e, em 2009, o primeiro *whitepaper* publicado por uma pessoa ou grupo denominado Satoshi Nakamoto (NAKAMOTO, 2008). Até então, o blockchain era considerado uma tecnologia apenas para o uso de moedas virtuais e limitado ao financiamento, porém essa tecnologia oferece uma gama de recursos que garantem a imutabilidade e integridade dos dados nela armazenados, além de prover privacidade aos usuários que estão conectados na rede. O interesse em estudar e conhecer essa arquitetura está no fato da moeda virtual Bitcoin ter ganhado milhões de usuários no mundo todo por mostrar um bom funcionamento e tem sido resistente a ataques e tentativas de fraudes.

O uso do Blockchain não está limitado apenas às moedas virtuais, mas pode ser aplicado em outras áreas que a integridade dos dados seja exigida. É importante entender que, para aplicar essa tecnologia em outras áreas, pode ser necessário escolher um tipo específico de blockchain. Cada tipo de blockchain possui a característica de ser imutável, mas o algoritmo de consenso, que, muitas vezes, é o que diminui o desempenho da rede, pode ser diferente para melhorar esse desempenho que não se tem no blockchain do Bitcoin. Basicamente a tecnologia demonstra que quanto mais segurança, mais a aplicação exige tempo para garantir que todos os pares da rede cumpram de forma correta suas obrigações (Nath, 2016).

2.3.2 Estrutura de um Blockchain

Teoricamente, a tecnologia blockchain pode ser definida como uma lista ligada, que segue apenas uma direção. Na área de finanças, o blockchain é entendido como um livro razão de registros, que tem a característica de ser quase impossível de ser modificado na rede, mesmo estando distribuído e público para todos os usuários. Como pode ser observado na Figura 3, cada componente da lista é um bloco, e dentro deste bloco estão as transações válidas e algumas informações adicionais do próprio bloco. A ligação desses blocos é feita de forma sequencial ligando um atributo interno do bloco chamado hash. Uma hash é gerada do bloco atual e inserida

Figura 3 – Estrutura básica de um blockchain.



Fonte. Próprio autor.

no próximo bloco a ser criado na rede, no campo hash anterior. Lembrando que o primeiro bloco não tem nenhuma hash anterior ligada a ele, se tornando o bloco gênesis da cadeia.

Como pode ser visto na figura 3, o bloco basicamente possui um cabeçalho e a área de dados. O cabeçalho de um bloco normalmente é composto pelos seguintes atributos: versão do bloco (indica o conjunto de regras de validação do bloco a seguir); *Merkle tree root hash* (valor de hash de todas as transações no bloco); *timestamp* (hora atual com segundos); nBits (limite de destino de um *hash* de bloco válido) e o *nonce* (um campo de 4 Bytes, que aumenta até que a *hash* atinja a regra necessária para ser válida). A área de dados do bloco contém as transações e um contador de transações. O número máximo de transações vai depender do valor limite do bloco ou da regra de transações máxima que pode ser definida.

Até o presente momento, definiu-se a estrutura do *ledger*, como são organizadas às transações nos blocos e como eles são encadeados sequencialmente. Porém, a rede blockchain possui muitos outros métodos e processos até que a transação seja validada. O primeiro passo para realizar as transações é com a criação de uma *wallet*. Uma *wallet* é uma carteira digital que, ao invés de conter dados sensíveis do usuários como documento de identidade, data de nascimento e senhas, ele possui apenas chaves criptográficas, uma pública e outra privada. Essas chaves são geradas por um algoritmo de criptografia assimétrica e essas chaves são complementares. A chave pública do usuário fica na rede e pode ser acessada por qualquer usuário mas a chave privada fica somente com o dono.

Para exemplificar esse processo de criar transações será utilizado dois personagens Alice e Bob. Quando Alice quer realizar negócios com Bob utilizando a rede blockchain, ela pega o conteúdo e criptografa com a chave pública de Bob, assinando o conteúdo com sua própria chave privada. Quando Bob recebe a mensagem criptografada de Alice, ele consegue verificar,

com a chave pública de Alice, se a informação não foi alterada no meio do caminho, e após isso, utiliza sua chave privada e complementar à chave pública para decifrar a mensagem e continuar a negociação. Note que em nenhum momento Bob e Alice precisam se conhecer e saber seus nomes, somente precisam utilizar os métodos para garantir que estão fazendo a negociação de forma confiável.

Após a negociação terminar, a transação é divulgada na rede e, no momento, ainda não é válida. Diferente dos sistemas tradicionais e centralizados, que são gerenciados por uma terceira pessoa negociação, como o banco central, que tem acesso a todos os dados dos usuários, o blockchain utiliza o algoritmo de consenso e conta com a participação dos mineradores. Os mineradores são responsáveis por validar as transações da rede e ganham uma recompensa por isso, como retribuição pelo trabalho. Além de validar as transações, eles reúnem um conjunto de transações e geram um bloco. A partir desse bloco, eles criam uma *hash* e esperam a aceitação de outros mineradores. No final da rodada, apenas um bloco é selecionado na disputa e adicionado na cadeia.

O algoritmo de consenso é um procedimento através do qual todos os pares da rede blockchain chegam a um acordo comum sobre o estado atual do livro contábil distribuído. Dessa maneira, os algoritmos de consenso alcançam confiabilidade na rede blockchain e estabelecem confiança entre pares desconhecidos em um ambiente de computação distribuído. Essencialmente, o mecanismo de consenso garante que cada novo bloco adicionado à blockchain seja a única versão da verdade que é acordada por todos os nós na blockchain. O algoritmo de consenso da blockchain consiste em alguns objetivos específicos, como chegar a um acordo, colaboração, cooperação, direitos iguais a todos os nós e participação obrigatória de cada nó no processo de consenso. Assim, um algoritmo de consenso visa encontrar um acordo comum que seja uma vitória para toda a rede. O processo de consenso explicado até aqui não precisa ser necessariamente igual a todas as tecnologias blockchain. Existem diversos tipos e com características que os tornam únicos, no entanto, existem três tipos de redes blockchain e serão discutidas na seção a seguir.

2.3.3 Tipos de Blockchain

Até o momento, foi explicada a arquitetura básica da rede blockchain e como são estruturados os blocos e as transações. Porém, atualmente existem vários tipos dessa rede desenvolvidos e utilizados em outras criptomoedas. O Bitcoin foi o primeiro a ser reconhecido e funciona até hoje, por ter o aspecto de ser totalmente distribuído e público, ou seja, qualquer usuário pode criar uma carteira ou se tornar um minerador e começar a participar da rede sem precisar ter um pré-requisito.

Como essa tecnologia pode ser integrada a diferentes contextos, alguns detalhes e funções precisam ser alterados para que se encaixe de forma correta e aplicação. Atualmente, o blockchain é divido em três tipos: público, consórcio e privado.

Tabela 1 – Características de cada tipo de blockchain.

	Blockchain público	**Blockchain consórcio**	**Blockchain privado**
Participação de usuários	Aberta	Aberta/Privada	Privada
Imutabilidade	Quase impossível de alterar	Pode ser alterado	Pode ser alterado
Permissão de leitura	Pública	Público/Privado	Público/Privado
Eficiência	Baixa	Alta	Alta
Centralizado	Não	Parcial	Sim
Processo de consenso	público	Organizações selecionadas	Privado
Desempenho	Baixo	Alto	Alto

Adaptado de: (Alkurdi *et al.*, 2018; Zheng *et al.*, 2017)

Blockchain público: Um blockchain público é uma rede aberta e qualquer um poder participar da rede. Além disso, o usuário pode participar da execução da tarefa aprovando as transações (mineração) ou como um simples usuário fazendo transações. O Bitcoin é a maior rede pública de blockchain atualmente.

Blockchain privado: Um blockchain privado é considerado uma rede centralizada, uma vez que é totalmente controlado por uma organização. O mecanismo de convite pode ser implementado para verificar a partir de um conjunto de condições a serem atendidas antes que um usuário possa participar da rede, seja por processo de autenticação ou controle de acesso. Esse tipo de blockchain pode ser considerado meio-termo para organizações dispostas a implementar a tecnologia blockchain nessas infraestruturas.

Blockchain consórcio: O blockchain de consórcio pode ser construído por grupos de organizações interessadas e tem a característica de ser parcialmente descentralizado, pois essas organizações podem determinar quem irá participar do processo de consenso, bem como se auto designarem como os mineradores dos blocos na rede.

Na Tabela 1, são organizadas as diferentes características dos três tipos de redes blockchain. Desta forma, é possível realizar a escolha do tipo mais apropriado para utilizar, dado o contexto preestabelecido. Em alguns casos, o preço de ter a característica de imutabilidade como predominante leva o sistema e a infraestrutura a terem uma alta capacidade de desempenho para processar todas as informações da rede e construção dos Blocos além do esquema de votação. Além desses tipos de redes, a seguir, são discutidas e explicadas as características apresentadas na tabela 1.

Participação dos usuários: A participação é aberta no blockchain público tanto para transacionar moedas como para minerar blocos. No blockchain consórcio, a participação é limitada para o consenso, que é gerenciado por algumas instituições. Já o blockchain privado,

tem regras mais definidas para que os participantes ingressam na rede e o consenso também é restrito e centralizado (Alkurdi *et al.*, 2018; Zheng *et al.*, 2017).

Imutabilidade: Como os registros são armazenados em um grande número de participantes, é quase que impossível adulterar transações em um blockchain público. Diferentemente dos blockchains privadas e de consórcio que podem ser adulterados facilmente, pois há apenas um número limitado de participantes (Alkurdi *et al.*, 2018; Zheng *et al.*, 2017).

Permissão de leitura: As transações do blockchain público são visíveis por todos os participantes, enquanto os blockchains privados e de consórcio vão depender de como foram configurados e podem ser tanto abertas como fechadas as visualizações das transações (Alkurdi *et al.*, 2018; Zheng *et al.*, 2017).

Eficiência: Leva muito tempo para propagar transações e bloqueios, pois há um grande número de nós na rede pública de blockchain. Como resultado, o rendimento da transação é limitado e a latência é alta. Por ter menos validadores, o blockchain do consórcio e o blockchain privado podem ser mais eficientes (Alkurdi *et al.*, 2018; Zheng *et al.*, 2017).

Centralizado: A principal diferença entre os três tipos de blockchains é que o blockchain público é descentralizado, o blockchain do consórcio é parcialmente centralizado e o blockchain privado é totalmente centralizado, pois é controlado por um único grupo (Alkurdi *et al.*, 2018; Zheng *et al.*, 2017).

Processo de consenso: Em blockchains públicos, cada nó pode participar do processo de consenso. Nos blockchains de consórcio, apenas um conjunto selecionado de nós é responsável por validar o bloco. Quanto aos privados, eles são totalmente controlados por uma organização que pode determinar o consenso final (Alkurdi *et al.*, 2018; Zheng *et al.*, 2017).

Desempenho: Por ser algoritmo de consenso, atualmente o blockchain público têm baixo índice para gerar os blocos e consome uma boa parte da energia produzida no planeta. Já os blockchains privados e de consórcio são mais rápidos e possuem menos etapas para validar as transações, e na maioria, o processo de criação de blocos é realizado por outros algoritmos de consenso menos complexos (Alkurdi *et al.*, 2018; Zheng *et al.*, 2017).

2.4 Protocolos de Consenso

Um aspecto importante da tecnologia distribuída citada até aqui é a obtenção de um consenso mútuo ou pela maioria, sem a interferência de um terceiro confiável ou de um sistema central de avaliação. Destacam-se aqui os algoritmos de consenso responsável pelos mecanismos que dispõem para ocorrer as validações, dentre esses algoritmos destacam-se alguns principais:

- Consenso Baseado nas Práticas Bizantinas a Tolerância a Falhas (PBFT): Algoritmo que fornece uma solução para o problema dos generais bizantinos que funciona em ambientes

descentralizados como as tecnologias de *ledgers* distribuídos que funcionam na internet. Envolve um protocolo de três fases e um nó (líder) que atua como minerador de blocos. O líder pode ser mudado pelo restante da rede por meio de votação, por motivos ou comportamentos indevidos. O PBFT trabalha com a suposição de que menos de um terço dos nós estão com defeito (f), razão pela qual diz que requer pelo menos $3f+1$ nós. No que diz respeito a eficiência é um dos algoritmos com melhor performance (Sukhwani *et al.*, 2017).

- *Proof of Work* (POW): Proof of Work (PoW) é o algoritmo de consenso usado no bitcoin que utiliza direitos e recompensas contábeis por meio da competição entre os nós. Com base nas informações do bloco anterior, calculam um novo *hash* para o bloco atual somado às informações do bloco em um problema matemático. O primeiro nó que resolve esse problema de matemática pode criar o próximo bloco e obter uma certa recompensa de bitcoin. Nesse processo de consenso tem-se quatro etapas: I) Obter a dificuldade: Após a produção de todos os blocos de 2016, o algoritmo de mineração de bitcoin ajustará dinamicamente o valor da dificuldade de acordo com a taxa de *hash* de toda a rede; II) Coletar transações: Colete todas as transações pendentes na rede após a produção do último bloco. Em seguida, calcule a raiz do *Merkle* dessas transações e preencha o número da versão do bloco, o valor do *hash* de 256 bits do bloco anterior, o valor atual do *hash* de destino, o número aleatório do *Nonce* e outras informações; III) Cálculo: Atravesse o *Nonce* de 0 a 232 e calcule o valor de *hash* duplo SHA256 na etapa 2. Se o valor de *hash* for menor ou igual ao valor alvo, o bloco poderá ser transmitido. A contabilidade completa do nó Após a verificação de outros nós; IV) Reiniciar: Se o nó não puder calcular o valor do *hash* em um determinado momento, ele repetirá a etapa dois. Se qualquer outro nó concluir o cálculo, ele será reiniciado a partir da etapa 1. O comprimento da cadeia é proporcional à quantidade de carga de trabalho. Todos os nós confiam na cadeia mais longa. Se alguém quiser adulterar a blockchain, ele precisa controlar mais de 50% do poder de *hash* do mundo para garantir que ele possa se tornar o primeiro a gerar o bloco mais recente e dominar a cadeia mais longa. Os ganhos com a adulteração podem ser muito maiores que o custo. Portanto, o PoW pode efetivamente garantir a segurança do blockchain (Yang; Chen; Chen, 2019).

- Proof of Stake (POS): Proof of Stake (PoS) foi desenvolvido com um dos objetivos principais diminuir o consumo de energia na mineração de Bitcoins, mudando o modo de selecionar o nó com mais chance de ganhar a produção do próximo nó. A seleção dos participantes é feita pela quantidade de cunhagem (taxas) mais alta, ou seja, ao invés de escolher o nó pelo poder computacional, a escolha é feita com base em apostas ou valor financeiro de cada nó, podendo reduzir a dificuldade na criação do bloco dando oportunidade para nós menos robustos terem chance. Além disso, as velocidades de geração de blocos e de confirmação de transações são mantidas a taxas constantes relativamente

baixas pelas redes PoW para garantir a segurança, pois existem muitos blocos diferentes propostos pelas mineradoras. Por outro lado, como apenas um bloco é feito em cada rodada de mecanismos de PoS, a geração de blocos e as velocidades de confirmação de transação são geralmente muito mais rápidas e, portanto, o mecanismo de PoS começa a se popularizar (Niya *et al.*, 2019).

2.5 Contratos Inteligentes

Os contratos inteligentes são considerados a terceira geração do blockchain ou **Blockchain 3.0**. Os contratos no blockchain tratam-se de uma nova forma de salvar e visualizar os dados em um blockchain. Assim como os contratos do mundo real, os contratos inteligentes executam condições que foram estabelecidas previamente. Porém, como dito, os contratos inteligentes são eletrônicos e executados em cima de um blockchain. Com isso, herda alguns atributos da estrutura blockchain, como por exemplo a imutabilidade. A imutabilidade nos contratos permite que, uma vez que sejam criados, não possam ser alterados ou adulterados por nenhuma das partes. Outro fator importante herdado é a distribuição. O contrato fica publicamente distribuído para todos e a saída é validada por todos os nós. Um usuário sozinho não conseguirá obter recursos de forma fácil na rede, pois outros participantes podem anular essa ação (Leka; Selimi; Lamani, 2019).

2.5.1 Estrutura do Contrato

Os contratos podem ser estruturados utilizando cláusulas condicionais previamente especificadas pela organização, cujos termos são postos em forma de condições e o contrato é executado quando todas as condições impostas são satisfeitas. Além disso, os contratos possuem dois atributos, valor e estado. Quando todos os atributos são definidos, o contrato é transmitido na rede e verificado pelos mineradores. Assim que o contrato é validado, o criador do contrato recebe o endereço do documento eletrônico e, dessa forma, pode invocar o contrato enviando uma transação. Os mineradores recebem uma transação ou criação de um contrato e executam de forma local em seu computador (normalmente uma máquina virtual) a verificação do contrato. Nessa verificação é determinado se o cenário atual (contrato ou transação + entrada de dados) atende às condições necessárias para completar a transação. Após a validação, os mineradores recebem uma recompensa como incentivo do sistema, pois contribuem com seus recursos computacionais para validar a transação. Assim que a transação tem sua validação, ela é inserida em um bloco no blockchain e a rede entra em consenso (Wang; Zhang; Kim, 2018).

A seguir serão exemplificados alguns tipos de redes com base em contratos inteligentes. A escolha do tipo de contrato está diretamente associada à rede distribuída escolhida. Dependo da aplicação, um modelo pode ser utilizado, um tipo de rede blockchain específica pode ser escolhida para atender às demandas da aplicação em que a rede será situada. A escolha do contrato e sua

implementação também está associada à tecnologia e existem atualmente diferentes modelos para criar redes públicas, privadas ou consórcio. A seguir são mostrados alguns modelos utilizados.

2.5.2 Exemplos de Redes que Implementam Contratos

Ethereum: *Ethereum* é um projeto parecido com o blockchain do Bitcoin e também possui a sua moeda digital chamada *Ether*. Porém, o *Ethereum* tem uma característica muito importante que o difere da rede blockchain: a capacidade de ser reprogramado e utilizado em aplicações distintas. Esses aplicativos descentralizados (ou "DApps") obtêm os benefícios da tecnologia de criptomoeda e blockchain. Eles podem ser confiáveis, o que significa que uma vez que eles são "enviados" para o *Ethereum*, eles sempre rodarão como programados. Eles podem controlar ativos digitais para criar novos tipos de aplicativos financeiros. Eles podem ser descentralizados, o que significa que nenhuma entidade ou pessoa individual os controla (WOOD *et al.*, 2014).

Ethereum pode ser visto como uma máquina de estado baseada em transação, que começa com o estado gênesis e incrementalmente executa transações para transformá-lo em alguns estados finais. Essa tecnologia introduz o conceito de contas sendo elas as contas de propriedade externa (EOAS) e contas de contrato. A diferença é que o primeiro é controlado por chaves privadas sem código associado a ele, enquanto o segundo é controlado pelo código de contrato com código associado (WOOD *et al.*, 2014).

Os usuários só podem iniciar uma transação por meio de um EOA. A transação pode incluir dados binários (carga útil) e *Ether*. Se o destinatário de uma transação for a conta zero, um contrato inteligente será criado. Por outro lado, se o destinatário for uma conta de contrato, a conta será ativada e seu código associado será executado no EVM local (a carga útil é fornecida como dados de entrada). A transação é então transmitida para a rede blockchain, onde os mineradores a verificarão. Para evitar problemas de abuso de rede e evitar os problemas inevitáveis decorrentes da integridade de Turing, todos os cálculos programáveis (por exemplo, criar contratos, fazer chamadas de mensagens, utilizar e acessar o armazenamento de contas e executar operações na máquina virtual) no *Ethereum* estão sujeitos às taxas - uma recompensa para os mineiros que contribuem com seus custos. A unidade usada para medir as taxas necessárias para os cálculos é chamada de gás (WOOD *et al.*, 2014).

HyperLedger: O *Hyperledger* é um framework colaborativo para criar uma estrutura e uma base de código de contabilidade distribuída de código aberto de nível corporativo. Tem como objetivo avançar tecnologia blockchain, identificando e realizando uma plataforma padrão aberta de vários setores para livros distribuídos, que pode transformar a forma como as transações comerciais são conduzidas globalmente. Ele seleciona apenas uma coleção de organizações relacionadas ao negócio, que pode participar por meio de um provedor de serviços de associação, e sua rede é criada a partir dos pares que são de propriedade e contribuídos por essas organizações. Os pares são hospedeiros para *ledgers* e *chaincodes* (contratos inteligentes). O *ledger* é o registro

sequenciado e resistente a violações de transações / transições de estado. A transição de estado é um resultado da invocação do *chaincode* (transação). Cada transação resulta em um conjunto de pares de chave-valor do ativo que são confirmados no razão como criações, atualizações ou exclusões (CACHIN, 2016).

1. Proposta. Um aplicativo envia uma proposta de transação para endossantes de diferentes organizações (também chamados de endossantes que validam transações contra políticas de endosso e aplicam às políticas). A proposta é uma solicitação para invocar uma função *chaincode* para que os dados possam ser lidos e / ou escritos no livro. Os resultados da transação incluem um valor de resposta, um conjunto de leitura e um conjunto de gravação. O conjunto desses valores junto com as assinaturas dos endossantes são retornados ao aplicativo como uma resposta da proposta de transação (CACHIN, 2016).

2. Embalagem: O aplicativo verifica as assinaturas dos endossantes e verifica se as respostas da proposta são as mesmas. Em seguida, o aplicativo envia a transação ao serviço de pedidos (solicitante) para atualizar o razão. O ordenador classifica as transações recebidas da rede e agrupa os lotes de transações em um bloco que está pronto para ser distribuído de volta para todos os pares conectados a ele (CACHIN, 2016).

3. Validação: Os pares conectados ao solicitante validam todas as transações dentro do bloco para garantir que ele tenha sido endossado de forma consistente pelas organizações exigidas de acordo com a política de endosso. Vale a pena notar que esta fase não requer a execução do *chaincode*, pois isso é feito apenas na fase de proposta. Após a validação, cada par anexa o bloco à cadeia e o *ledger* é atualizado (CACHIN, 2016).

Arquitetura do Hyperledger

O *Hyperledger* é uma implementação exclusiva de tecnologia de razão distribuída (DLT) que garante integridade e consistência de dados, ao mesmo tempo em que oferece responsabilidade, transparência e eficiências incomparáveis com outras tecnologias de *blockchain* ou DLT. O *Hyperledger* implementa um tipo específico de rede de *blockchain* com permissão na qual os membros podem rastrear, trocar e interagir com ativos digitalizados usando transações que são governadas por contratos inteligentes chamado de *chaincode* e trás uma maneira robusta dos participantes na rede interagir de uma maneira que garante que as transações e dados possam ser restritos a um subconjunto identificado chamado de canal (COMPOSER, 2019).

A rede *blockchain* oferece suporte para que os membros estabeleçam registros compartilhados que contêm a fonte da verdade sobre esses ativos digitalizados e transações registradas, que são replicadas de maneira segura apenas para o conjunto de nós que participam desse canal (COMPOSER, 2019).

A arquitetura do *Hyperledger* é composta dos seguintes componentes: nós de mesmo nível, nós de pedido e os aplicativos clientes que provavelmente estão aproveitando um dos SDKs

do *Hyperledger* específicos de uma linguagem. Esses componentes têm identidades derivadas de autoridades de certificação. O *Hyperledger* também oferece um serviço de autoridade de certificação, fabric , mas você pode substituí-lo pelo seu próprio. Todos os nós de mesmo nível mantêm o razão/estado ao confirmar transações. Nessa função, o par é chamado de *committer*. Alguns pares também são responsáveis por simular transações executando *chaincodes* (contratos inteligentes) e endossando o resultado. Nessa função, o par é chamado de endossante. Um par pode ser um endossante para certos tipos de transações e apenas um mantenedor do razão (*committer*) para outras (COMPOSER, 2019).

O razão fornece um histórico verificável de todas as alterações de estado bem-sucedidas e tentativas malsucedidas de alterar o estado, ocorrendo durante a operação do sistema. O razão é construído pelo serviço de pedidos como uma cadeia de *hash* totalmente ordenada de blocos de transações. O *hash chain* impõe a ordem total dos blocos em um razão e cada bloco contém uma matriz de transações totalmente ordenadas. Isso impõe ordem total em todas as transações (COMPOSER, 2019).

O razão é mantido em todos os pares e, opcionalmente, em um subconjunto de pedidos. No contexto de um pedido, nos referimos ao razão como a *OrdererLedger*, enquanto no contexto de um par nos referimos ao razão como a *PeerLedger*. *PeerLedger* difere do *OrdererLedger* em que os pares localmente mantêm uma máscara de bits que diferencia as transações válidas das inválidas. Os pares podem podar *PeerLedger*. Os encomendantes mantêm *OrdererLedger* para tolerância a falhas e disponibilidade e podem decidir podá-lo a qualquer momento, desde que as propriedades do serviço de pedidos sejam mantidas. O razão permite que os pares reproduzam o histórico de todas as transações e reconstruam o estado (COMPOSER, 2019).

Os nós são as entidades de comunicação do *blockchain*. Um nó é apenas uma função lógica no sentido de que vários nós de diferentes tipos podem ser executados no mesmo servidor físico. O que conta é como os nós são agrupados em domínios confiáveis e associados às entidades lógicas que os controlam. Existem três tipos de nós: I) Cliente ou cliente de envio: um cliente que envia uma solicitação de transação real aos endossantes e transmite propostas de transação ao serviço de pedidos. II) *Peer*: um nó que confirma as transações e mantém o estado e uma cópia do razão. Além disso, os pares podem ter um papel especial de endossante e III) Nó de serviço de pedido ou solicitante: um nó que executa o serviço de comunicação que implementa uma garantia de entrega, como a transmissão atômica ou total do pedido (COMPOSER, 2019).

O projeto *Hyperledger* está entregando uma plataforma de *blockchain* projetada para permitir a troca de um ativo ou o estado de um ativo a ser consentido, mantido e visualizado por todas as partes em um grupo autorizado. Uma característica chave do *Hyperledger* é que o ativo é definido digitalmente, com todos os participantes simplesmente concordando em sua representação / caracterização. O *Hyperledger* pode oferecer suporte a uma ampla variedade de tipos de ativos; variando do tangível ao intangível (COMPOSER, 2019).

A tecnologia é baseada em um conceito de *blockchain* padrão, um livro razão comparti-

lhado e replicado. No entanto, o *Hyperledger* é baseado em um glossário do *Hyperledger*, o que significa que todos os participantes devem ser autenticados para participar e realizar transações no *blockchain*. Além disso, essas identidades podem ser usadas para controlar certos níveis de controle de acesso. Essa dependência da identidade é uma grande vantagem, pois algoritmos de consenso variáveis podem ser implementados no lugar das variedades de Prova de Trabalho e Prova de *Stake* mais intensivas em computação. Como resultado, as redes permitidas tendem a fornecer taxas de transferência de transações e desempenho mais altos (COMPOSER, 2019).

2.6 Algoritmo K-means

Por fim, para a elaboração do trabalho foi utilizado um algoritmo de clusterização tradicional denominado K-means (KRISHNA; MURTY, 1999; JAIN, 2010). O objetivo comum em algoritmos baseados em *clusters* é descobrir agrupamentos, padrões e similaridades de um conjunto de dados, pontos ou objetos. Dessa forma, podem-se encontrar grupos que possuem uma característica em comum e que normalmente não é observável aos olhos em um conjunto de dados.

Alguns objetivos principais são necessários para evidenciar a utilização do algoritmo, dentre esses objetivos podemos destacar:

- **Estrutura subjacente**: para obter informações sobre os dados, gerar hipóteses, detectar anomalias e identificar recursos importantes (JAIN, 2010).
- **Classificação natural**: para identificar o grau de similaridade entre formas ou organismos (relação filogenética) (JAIN, 2010).
- **Compactação**: como um método para organizar e resumir os dados através de protótipos de cluster (JAIN, 2010).

Dado o conjunto $X = \{x_i\}, i = 1,...,n$ onde n representa vários pontos dimensionais que serão agrupados em um outro conjunto K de *clusters*, $C = \{c_k, k = 1,...,K\}$. O algoritmo deverá encontrar o erro quadrático mínimo entre a média empírica de um *cluster* e os pontos no *cluster*. A média do cluster c_k é definida como u_k. Assim , podemos definir o erro quadrático como:

$$J(c_k) = \sum_{x_i \in c_k} ||x_i - u_k||^2.$$

Como o objetivo é minimizar a soma dos erros quadráticos sobre todos os K *clusters*, temos:

$$J(C) = \sum_{k=1}^{K} \sum_{x_i \in c_k} ||x_i - u_k||2.$$

Esse problema é conhecido como NP difícil mesmo para um $K = 2$. O K-means começa com uma partição inicial com os *clusters* K e atribui padrões aos *clusters* para reduzir o erro ao quadrado. Como o erro ao quadrado sempre diminui com um aumento no número de *clusters*

$K(comJ(C) = 0quandoK = n)$, ele pode ser minimizado apenas para um número fixo de *clusters*. Por fim, mais algumas etapas são importantes para o decorrimento do algoritmo:

- Selecionar uma partição inicial com *clusters* K ; repetir as etapas 2 e 3 até a associação do *cluster* estabilizar.
- Gerar uma nova partição, atribuindo cada padrão ao centro do *cluster* mais próximo.
- Calcular os novos centros de *cluster*.

Após evidenciar todos os termos, conceitos que fazem parte deste trabalho, na próxima seção estão descritos os trabalhos relacionados que contém aspectos ou arquiteturas próximas ao proposto. Alguns trabalhos citados estão em andamento, mas o objetivo não é mostrar falhas dos outros trabalhos, e sim discorrer sobre suas arquiteturas e mostrar semelhanças e diferenças entre os trabalhos. Apesar de cada arquitetura ter pontos em comum entre as outras o foco do trabalho e a atuação são diferentes.

2.7 Validação pelo Teste de Conhecimento Zero

Para validar a camada de entrada do MADCS, realizou-se o teste de conhecimento zero (FEIGE; FIAT; SHAMIR, 1988) para verificar as condições e validar as transações no MADCS. Assim, é possível provar que a informação é válida, mesmo que seja uma referência fora da cadeia sem que o usuário realmente veja a informação. Para realizar essa avaliação, tem-se a seguinte hipótese:

O participante A deve provar ao participante B que as informações da transação são válidas e verdadeiras, para que possam receber o que está contido nelas. O participante B não verá essas informações, mas ele sabe que pode comparar dados criptografados com o que foi gerado inicialmente quando as informações são consultadas com um Y possivelmente confiável.

A tem uma informação X e o participante B é um verificador que precisa saber se a informação de X_A é verdadeira, mas A não será revelada antes informações para B até o sistema retornar que A pode ter o que as informações contêm. A precisa provar que X é verdadeiro para B através de um teste. Sabe-se que X é uma informação importante e que está armazenada em um local especial e secreto, e que se A tiver acesso a essas informações, poderá provar a B que é verdade . Assim, B solicita A uma prova que saiba onde X está. A fornece um caminho criptografado de X para B e B pode verificar se esse caminho é verdadeiro.

Para X ser verdadeiro, precisa-se:

1. A deve conhecer o caminho de X.
2. X não foi alterado.

3. X não foi invalidado.

Depois de verificar se o caminho de X é válido, B pode analisar se X não foi alterado ou invalidado antes de chegar a suas mãos. Para X ser inválido, é necessário que: X foi usado mais do que o permitido. X não era válido anteriormente. X mudou frequentemente.

Além disso, a análise B verifica se todas as condições são verdadeiras e satisfeitas para validar X_A. Dessa forma, B incrementa X_A o uso e o deixa válido. Observe que B não sabe quantas vezes X_A foi usado e seu estado anterior, mas sabe que não era inválido. B também não conhece as informações X_A, exceto o que você precisa entregar para A. B sabe que X é válido mesmo sem saber o Y emitido, pois apenas A teria X se um Y estiver dentro S (domínio de quem conhece o sistema), associado (A,X).

Agora tem-se que o domínio contém X,A,Y,SeB e as funções P e H (associação e herdeiro), onde B é um conjunto de $B'(B = B1,B2,B3,...,Bn)$ A é um conjunto de $Y(S = Y1,Y2,Y3,...,Yn)$, A é um conjunto de $A'(A = A1,A2,A3,...,An)$ e X é um conjunto de transações $X'(X = X1-t,X1,X1+t,X2,X3,...,Xn)$ e as relações temporais disso X pode indicar se é válido ou não.

Dado o domínio e conjunto, tem-se que:

1. $Y \in S$. (Y pertence a S).

2. $X' \in X$. (X' pertence a X).

3. $B' \in B$. (B' pertence a B).

A função P é uma função de associação à qual tem-se três variáveis. O primeiro e o objeto, a segunda ação $(\in, \notin)$ e a terceira o solicitante. A função H é uma função de associação para o segundo proprietário para o qual temos quatro variáveis. O primeiro e o objeto, a segunda ação $(\in, \notin)$, o terceiro o requerente e, finalmente, o primeiro proprietário.

Dado o momento em que X solicita informações para Y, as informações geradas X' são primeiro associadas a Y. A função de associação A é usada para fazer a primeira associação. $P(X', \in, Y)$. A função de associação diz que um X' pertence a um Y escolhido. Para A ser o Dono de X', é necessário que ele herda esse objeto por um Y. $H(X', \in, A, Y)$. A função do herdeiro diz que a X' pertence a um A a a Y.

Assim, tem-se que: $X' \in A$.

Para validar o XA, é necessário verificar se ele permanece igual ao emitido por Y. Isso pode ser observado a partir da seguinte função:

$$G(X_A, X_Y, Y) : (X_Y \rightarrow (X_A \equiv X_Y)) \wedge (P(X_A, \in, Y))$$

A partir dessas definições, temos as seguintes observações sobre como será o estado de A.

1. se $A \notin Y$, então A é Falso.
2. se $H(X_Y, \notin, A, Y)$, então A é Falso.
3. se $H(X_Y, \in, A, Y)$, então A pertence a X.
4. X é Verdadeiro se $(X \equiv X_Y \equiv X_A)$.

Concluí-se que para verificar B para analisar as informações é necessário que haja a criptografia de A. Após obter essa criptografia, é necessário analisar se o conteúdo de A possui o caminho no local secreto. Caso confirmado as informações é necessário comparar se esse conteúdo secreto é o mesmo que pertenceu ao Y que passou nesse conteúdo. Se este Y não emitiu o conteúdo ou foi alterado após A herdá-lo de Y, o conteúdo é considerado inválido.

CAPÍTULO

3

TRABALHOS RELACIONADOS

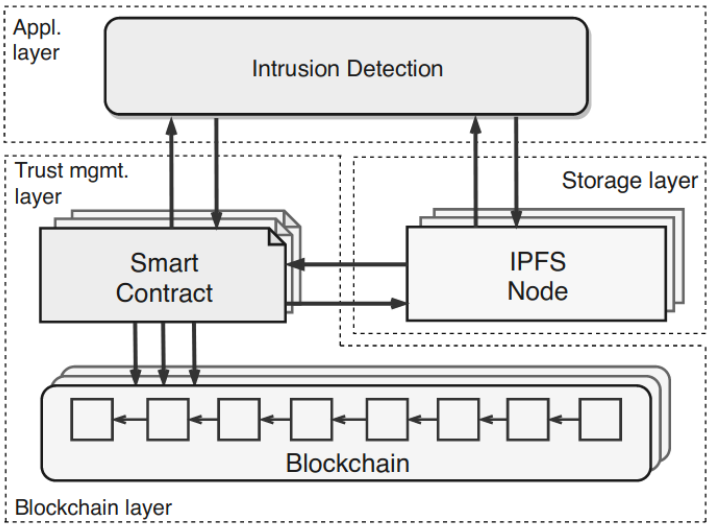
Nesta seção são apresentados os trabalhos relacionados sobre o tema proposto. Uma breve explicação sobre cada trabalho é feita, bem como a comparação de ambas as arquiteturas propostas e as diferenças com a arquitetura proposta neste trabalho. O intuito é explicar como as diferentes arquiteturas, que apesar de seguirem um mesmo propósito de encontrar intrusões ou anomalias, atuam em diferentes pontos da rede e são configuradas de formas diferentes.

3.1 Visão Geral

O primeiro trabalho relacionado é denominado *Towards Scalable and Trustworthy Decentralized Collaborative Intrusion Detection System for IoT* (2020) (PUTRA *et al.*, 2020). Este trabalho, possui uma arquitetura muito semelhante ao *middleware* desta dissertação e por este fato está incluso com um trabalho relacionado e que dá peso ao tema tratado que é a detecção de anomalias em uma rede Blockchain. Como é possível observar na Figura 4, o modelo trata-se de uma arquitetura de quatro camadas baseada em Sistemas de Detecção de Intrusão na sigla em inglês Intrusion Detection System (IDS). A camada de aplicação é composta por este módulo IDS que treina e executa modelos de detecção e coleta assinaturas de intrusão, além de executar um nó leve de blockchain. Utiliza-se para a detecção de anomalias um modelo Support Vector Machine (SVM) (*Support Vector Machine*) supervisionado e a detecção baseada em assinaturas funciona combinando eventos observados com as assinaturas de intrusão sem mencionar como ocorre esse processo de determinar o que é ou não uma intrusão. Um ponto interessante é o armazenamento externo à rede Blockchain utilizando o banco distribuído IPFS. Esse banco permite armazenar dados maiores, como o próprio SVM, para gerenciar os modelos e atualizações desses modelos em cada *peer*, vinculando o modelo a um metadado armazenado no Blockchain. Essa arquitetura permite que o detector de intrusão se comunique com o IPFS e com os contratos inteligentes.

Na camada de aplicação, os IDS são inseridos nos *gateways*, pois são necessários para executar operações que precisam de armazenamento e quantidades consideráveis de CPU e

Figura 4 – Arquitetura do modelo proposto e denominado Collaborative-IDS (CIDS) do trabalho relacionado.



Fonte: (PUTRA *et al.*, 2020).

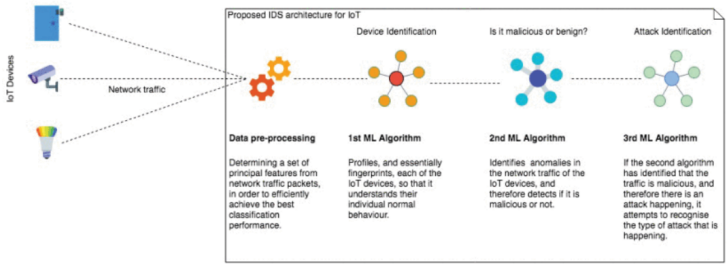
memória como criptografia e o aprendizado de máquina.

O segundo trabalho com o título *A Supervised Intrusion Detection System for Smart Home IoT Device* foi realizado por (Anthi *et al.*, 2019). Este trabalho também apresenta uma arquitetura IDS em um ambiente IoT para detecção de intrusões. Como pode ser visto na Figura 5, esta arquitetura apresenta uma única camada com várias fases. A primeira camada identifica os dispositivos IoT com base em endereços MAC e tem uma primeira classificação conforme o comportamento da rede. Na segunda fase, o pacote do dispositivo é identificado por meio de um algoritmo de machine *learning* para reconhecer os dispositivos inteligentes. Na terceira fase, os pacotes passam por um classificador para verificar se são maliciosos ou não. Na última fase, um terceiro classificador recebe os resultados dos classificadores das outras camadas e determina se o pacote é ou não malicioso.

A rede é treinada para detectar tipos de ataques como DoS, DDoS e tipos de ataques que visam tentativas de vazar informações dos dispositivos. Para avaliar o desempenho dos algoritmos de classificação, foi utilizada a medida de precisão. A precisão mede o número de pacotes que foram classificados corretamente. No entanto, o problema do uso da precisão para medir a eficácia de um classificador é que, se o classificador sempre predizer uma determinada classe, uma estratégia que derrota o propósito de criar um classificador atingirá alta precisão. Os experimentos foram feitos utilizando bases de dados de dispositivos Iot, com uma divisão de 60% para treino e 40% para teste. Um detalhe, foi a escolha do algoritmo com base no seu desempenho.

O terceiro trabalho relacionado estuda e desenvolve uma estrutura genérica de IDS utilizando assinaturas no blockchain para gerar regras confiáveis no ambiente descentralizado

Figura 5 – Arquitetura do modelo proposto e denominado Proposed IDS Architecture for IoT.



Fonte: (Anthi *et al.*, 2019).

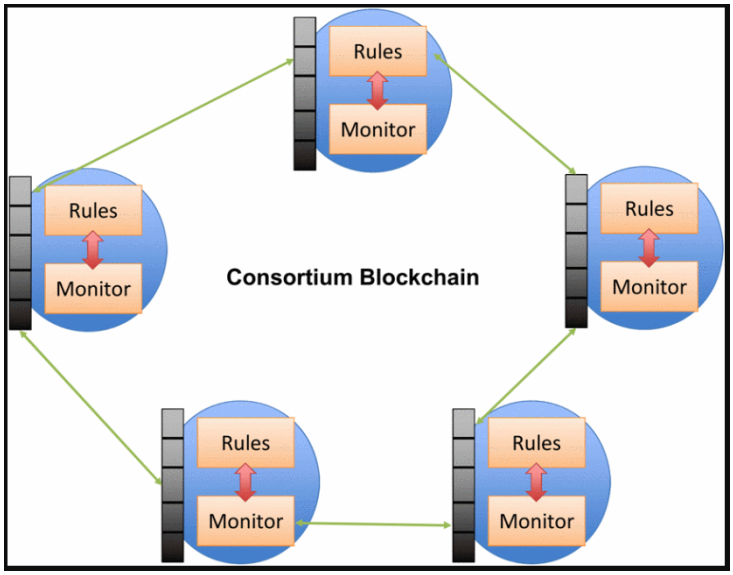
e colaborativo, cujo título é CBSigIDS: *Towards Collaborative Blockchained Signature-Based Intrusion Detection* (Tug; Meng; Wang, 2018). O tipo de blockchain escolhido é o de consórcio, pois o objetivo foi ter um grupo de nós válidos que possam avaliar e compartilhar essas informações referentes aos dados que trafegam no seu perímetro, de modo que esses nós colaborem na detecção de intrusões de nós clientes ou atividades de invasão na rede.

Na arquitetura mostrada na Figura 6, o foco está em três componentes principais em uma rede distribuída, a comunicação entre os pares, a colaboração e o componente capaz de gerenciar e prover confiança na rede. O último componente é necessário para implementar abordagens apropriadas para avaliar a confiança e indicar as reputações na rede. O componente de colaboração ajuda um nó IDS a coletar informações necessárias para avaliar a confiabilidade dos nós de destino ou enviar dados relevantes solicitados por outros nós.

Essa arquitetura tem um aspecto interessante e supõe que um invasor possa controlar um ou dois nós por métodos de infecção, mas não pode gerenciar com êxito um grande número de nós IDS dentro de um curto período, além da dificuldade de manipular por muito tempo, esses nós devido às chaves de criptografia utilizadas. Com isso, a arquitetura pretende identificar intrusões, compartilhar para os outros nós de consórcio essa informação e avaliar a reputação por grupos de nós para identificar os intrusos.

O Quarto Trabalho é um dos mais relacionados com o tema dessa dissertação, pois se trata de detecção de anomalias através do Blcokchain em um ambiente IoT colaborativo e apresenta o título CIoTA: *Collaborative IoT Anomaly Detection via Blockchain* (2018) (GOLOMB; MIRSKY; ELOVICI, 2018). O CIoTA, como é mostrado na Figura 7, é um modelo de detecção de anomalias baseado em Blockchain, onde modelos locais são treinados em cada dispositivo inteligente, no caso simulado com cerca de 48 Raspberry Pis como é mencionado no trabalho. Essas raspberrys agem como infiltrados e recolhem informações de inteligência sobre os dados colhidos, produzindo relatórios de inteligência. Quando algo é encontrado, um bloco pode ser criado e publicado, compartilhando a informação com os outros dispositivos através da rede Blockchain. Dessa forma os algoritmos locais de cada IoT podem ser atualizados com mais dados tanto benignos como malignos (anomalias).

Figura 6 – Arquitetura do modelo proposto e denominado CBSigIDS Blockchain.



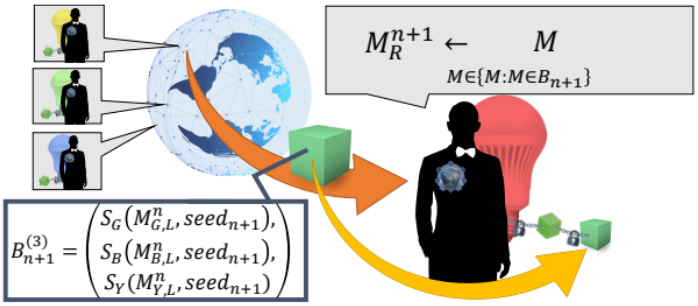
Fonte: (Tug; Meng; Wang, 2018).

De modo geral o autor determina esses dispositivos inteligentes com o algoritmo baseado em Cadeias de Markov para avalias as situações e instruções da rede. A missão de cada agente é continuamente detectar atos maliciosos e reunir a Inteligência sobre o que está acontecendo e compartilhar periodicamente uma coleção da Inteligência (como rumores) com outros agentes. Como um agente está no território do inimigo, ele pode receber a Inteligência falsa, que introduz ruído nos rumores transmitidos aos outros agentes. Portanto, um agente confiará apenas em um boato (alguns Intel) se ele souber que pelo menos outros agentes ouviram o mesmo boato. Finalmente, um agente aceita o conjunto mais recente de rumores confiáveis.

Após a publicação do Bloco, os agentes avaliam a informação para obter consenso e validar o bloco com informações de inteligência. Outros agentes só aceitarão esse bloco parcial se ele for maior que o bloco parcial e se puderem atestar que é seguro (verificando-o em relação ao seu próprio modelo local). Assim, o bloco parcial só aumenta se a maioria dos agentes tiver verificado que ele contém um modelo seguro. Depois que o bloco parcial tiver relatórios L, ele será fechado como um bloco concluído. Portanto, um agente recebe as informações mais recentes de seus colegas agentes, substituindo ML pelo modelo combinado contido no mais novo bloco fechado.

Em (HODO *et al.*, 2016), foi realizado uma análise de ameaças em rede IoT utilizando uma rede neural para a detecção de intrusão. Apesar do Procedimento não utilizar Blockchain, é um trabalho muito interessante, que faz a detecção de intrusões sem utilizar a rede Blockchain como base, mas avalia um ambiente com grande volume de dados. Um detalhe interessante é a definição de intrusão e as técnicas de detecção de intrusão citadas que são de extrema importância neste trabalho, pois concentrou-se em etapas como análise estatística com a comparação de tendências e estabelecendo uma base para a detecção. Algoritmo evolutivo, pois o modelo

Figura 7 – Arquitetura do modelo proposto e denominado CIoTA.



Fonte: (GOLOMB; MIRSKY; ELOVICI, 2018).

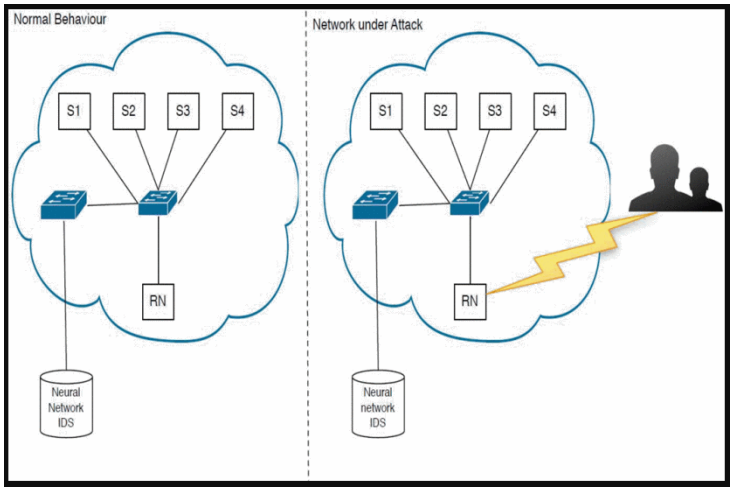
permite que o algoritmo seja treinado com novos valores e com informações de outros modelos dado pela comunicação que foi denominado de fofoca. A fofoca seria o compartilhamento que os modelos fazem dada a sua base local de informação que é compartilhada e ajuda aprimorar o modelo.

Esse trabalho utiliza a técnica de RNA, com Redes Neurais Artificiais para gerar um modelo que detecte as intrusões. Em seu cenário, como é mostrado na Figura 8, possui uma rede IoT que é composta por 5 nós sensores. Quatro dos nós estão atuando como cliente e um está atuando como um nó de retransmissão do servidor para fins de análise de dados. O tráfego é capturado por meio de um toque na rede, evitando a modificação do tráfego ao vivo. O nó do servidor reconhece os dados enviados pelos nós sensores e responde com os dados com base nos dados recebidos. O invasor externo é o ponto de foco para o trabalho citado. É avaliado o comportamento do nó servidor, que recebe e é entrada dos dados. À medida que o nó do servidor fica sem resposta, os nós dos sensores não conseguem adaptar seu comportamento, levando a uma falha no sistema monitorado. A detecção do ataque é, portanto, crucial, permitindo que a equipe de resposta evite interrupções na rede de sensores e garanta a estabilidade da rede. No trabalho é avaliado ataques como DoS e DDoS, tendo uma precisão geral de 94%.

3.2 Comparação dos Trabalhos Relacionados

Cada arquitetura possui aspectos semelhantes e diferenças ao que foi apresentado neste trabalho. A primeira diferença encontrada é o nível que cada arquitetura trata as intrusões ou anomalias. De modo geral esses termos tratam de tentativas externas de ataques ao sistema e verificam ataques como Dos e DDos, em que o ataque consegue fazer com que a aplicação negue o serviço. Nos casos mais específicos como neste trabalho, as intrusões ou anomalias estão ligadas a problemas internos a aplicação, falhas de segurança e tentativas de usuários internos de burlar o sistema, ou utilizar algum defeito de programação e infligir o fluxo correto que a aplicação deveria seguir. Cada arquitetura foi projetada para mitigar esse problema em seu contexto e pode ser ampliada para outros problemas. Para discutir melhor, reunimos algumas

Figura 8 – Arquitetura do modelo proposto e denominado *Threat analysis of IoT networks using artificial neural network intrusion detection system*.



Fonte: (HODO *et al.*, 2016).

questões para focar em alguns pontos e relacionar esses adjetivos de cada arquitetura e o que cada arquitetura provê ao sistema final.

1. Possui uma Arquitetura Distribuída e Descentralizada ?
2. Apresenta um gerenciamento dos Modelos distribuídos ?
3. Realiza o processo de atualização do Modelo Localmente nos nós?
4. Realiza o processo de atualização do Modelo com base em informações de outros nós?
5. Detecta Intrusões?
6. Detecta anomalias?
7. Realiza a detecção de Anomalias e/ou Intrusões avaliando os clientes internos da organização?
8. Utiliza dados do Blockchain e dados das funções exercidas pelos nós para encontrar anomalias e funções adversas ao apresentado diariamente na rede?
9. Como foi avaliado o modelo ? (real, simulado)
10. Utilizou dados sintéticos ou reais? Qual a precisão final?

Dentre as perguntas realizadas e com objetivo de avaliar e criar uma arquitetura que possui um aspecto que a diferencie, foi escolhida a detecção de anomalias (ameaças internas de usuários, tentativas de inserir dados inválidos em transações e tentativas de burlar o fluxo de transações). A arquitetura não trata explicitamente a detecção de intrusões externas, pois

Tabela 2 – Questões avaliativas para identificar semelhanças e diferenças nas arquiteturas propostas.

Artigos	**Perguntas**									
	1	2	3	4	5	6	7	8	9	10
Putra *et al.* (2020)	✓	✓	✓	✓	✓				R(NC)	NC
Anthi *et al.* (2019)		NC	NC		✓		✓		S	sintético 96%
Tug, Meng e Wang (2018)	✓	✓	✓	✓	✓		✓		SR	S 67%
Golomb, Mirsky e Elovici (2018)	✓	✓	✓	✓		✓	✓		R	NC
Hodo *et al.* (2016)			✓		✓				S	S 99.4%
Modelo Proposto	✓	✓	✓	✓		✓	✓	✓	S	S 75%

Legenda: P = Pacial, NC = Não Comentado, SR = Simulado e Real, S= Simulado e R = Real

com a arquitetura baseada em blockchain integra as questões de autenticidade, imutabilidade e distribuição dos dados. O interesse e ponto foco é verificar a informação e detectar anomalias baseando-se em históricos de transações na rede blockchain e atividades dos usuários. Com isso, o trabalho detecta possíveis anomalias antes que cheguem à rede blockchain como uma espécie de filtro, que invalida transações suspeitas e evidência às organizações e ao usuário o estado da transação para que medidas sejam tomadas na verificação do ocorrido.

Em comparação às outras arquiteturas, o intuito é completar e investigar um problema que pode ocorrer em redes blockchains e também em outras redes com a característica de serem distribuídas, para mitigar não apenas ataques externos como apresenta as outras arquiteturas, mas investigar tentativas internas de usuários burlarem as regras de negócios inseridas na rede.

Um aspecto interessante, e uma diferença nos trabalhos relacionados, está ligado ao armazenamento e gerenciamento dos modelos de detecção. Alguns trabalhos utilizam o blockchain como um gerenciador dos modelos que fazem detecção de intrusão em redes descentralizadas. Assim, o algoritmo está registrado na rede e novas alterações devem passar por uma votação para serem incluídas, na tentativa de eliminar modelos que contradizem o atual. Em outros casos existem um modelo central que possui um algoritmo mais significativo e modelos locais que atualizam e se comunicam com outros modelos de detecção. Por fim, o modelo proposto utiliza os contratos inteligentes e insere o modelo de detecção neste contrato, dessa forma, está implicitamente utilizando a imutabilidade que se faz ao criar um contrato que não será modificado. Quando houver atualizações um novo contrato pode ser emitido ao invés de atualizar o contrato atual. Além disso, o contrato checa, se comunica e atualiza dentro da camada de blockchain implementada, e não com cada usuário, na rede, a camada de aplicação, para evitar tentativas de corromper o modelo de detecção. A comunicação e desenvolvimento interno do algoritmo também auxiliam em deixar os modelos mais confiáveis, exceto pelo fato de um modelo poder ter sido treinado incorretamente e por isso, o mecanismo de votação verifica modelos inseguros.

CAPÍTULO

4

MADCS

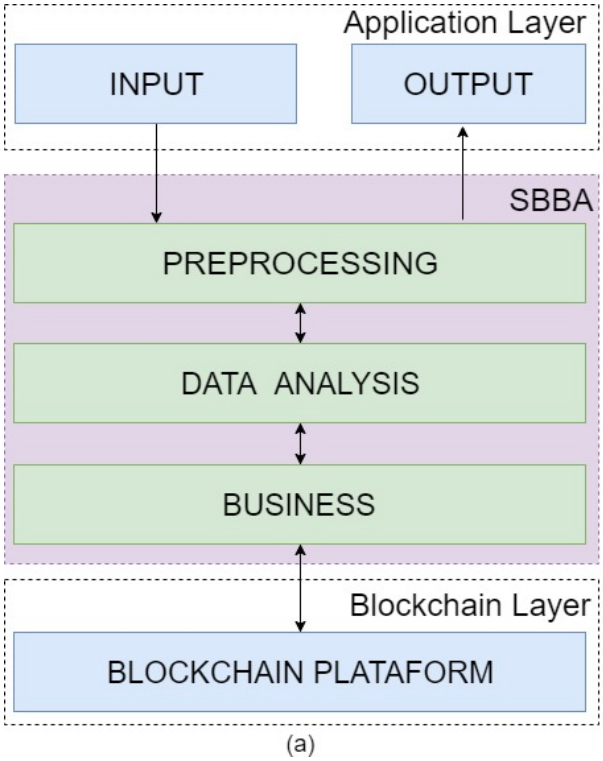
Nesta seção, apresenta-se o *middleware* desenvolvido para sistemas baseados em blockchain. A arquitetura do *middleware* e da rede desenvolvida é apresentada, bem como os detalhes de cada camada.

4.1 Introdução ao MADCS

Como mencionado no referencial teórico, a rede blockchain possui aspectos de segurança e privacidade, além da imutabilidade dos blocos, que permitem a troca de transações com segurança e sem mostrar precedentes de seus usuários. No entanto, imagine que a rede blockchain, mesmo tendo o algoritmo de consenso, permita que informações inválidas sejam inseridas na rede pelas organizações internas associadas que estão geograficamente distribuídas, para que as informações sejam registradas influenciando transações futuras e erros de propagação. As informações que saem da rede podem ser consideradas seguras e imutáveis, mas as informações que chegam à rede dos usuários (pessoas ou dispositivos inteligentes) podem conter erros ou anomalias que a própria rede pode não verificar. Portanto, um mecanismo que verifica o histórico da transação, analisa a transação atual e avalia sintática e semanticamente se faz necessário para garantir a confiabilidade antes de ser entregue à rede blockchain. Essa camada também avalia se o dispositivo tem permissão para executar a ação exigida no contrato em que os dados serão repassados, evitando tentativas de executar contratos sem autorização. Portanto, esta seção tem como objetivo descrever o *middleware* que estará entre a entrada de dados da organização e o livro razão da rede blockchain para manipular as informações de entrada.

Como pode ser visto na Figura 9, o *middleware* MADCS foi estruturado em seis camadas, começando com a camada de entrada, saída, pré-processamento, análise de dados, a camada de negócios que é o próprio contrato e a camada da rede blockchain. Na camada de entrada e saída, estão as ações de criptografar e descriptografar os dados para que a análise avalie os dados enviados na transação. Na camada de pré-processamento, filtrou-se os dados para remover

Figura 9 – Middleware MADCS proposto.



Fonte: Próprio autor.

antecipadamente informações nulas e complexas, transformando os dados numericamente; no entanto, essa transformação depende do contexto em que os dados precisam ser tratados. Após a transformação, a camada de análise de dados recebe os dados para extrair a semelhança do *cluster* criado anteriormente, sendo possível atualizar o *cluster* com os novos dados. Assim, em um caso positivo, a transação pode passar para o contrato e executar normalmente, e em um caso negativo, a camada de saída envia um *feedback* para a organização sobre a anomalia para que possa verificar e se situar no problema, se isso acontecer. Ao mesmo tempo, a camada de saída emite essa detecção para outros *clusters* como uma fofoca, para que outras tentativas sejam identificadas diretamente e todos os gerenciadores de *cluster* sejam atualizados.

4.2 Camada de Entrada

Fornece o serviço de recebimento de dados. Os dados são recebidos de organizações que possuem DApps para seus *stakeholders* para realizar transações com o objeto sensível em negociação. Dois tipos de dados podem ser recebidos: 1) **Dados de informação:** necessariamente dados recebidos de dispositivos ou sensores inteligentes, que de alguma forma influenciarão a execução do contrato preestabelecido na camada de referência da blockchain e na tomada de decisões. **2) Dados no formato de transação do cliente externo:** esse tipo de dado já passou pelo aplicativo na camada de cliente externo,foi autorizado pela organização e está no formato

de transação pronto para validação e ser executado. Antes de mover a transação para a camada de pré-processamento, são executadas as seguintes etapas:

Validação de transação: Como explicado na estrutura, quando um usuário assina uma transação, ele usa sua chave privada. Portanto, quando o remetente recebe a transação e deseja saber se ela ainda está intacta, ele pode usar a chave pública do emissor que é complementar à chave usada na assinatura e provar a integridade da transação.

Decodificação de transação: Após a transação ter sido verificada, o usuário pode descriptografar a mensagem, sabendo que nada foi alterado. A descriptografia do conteúdo recebido pode ser feita usando a chave privada, que é a chave pública usada para criptografar a mensagem. Dessa forma, apenas o remetente, que terá a chave privada correspondente, pode descriptografar a mensagem.

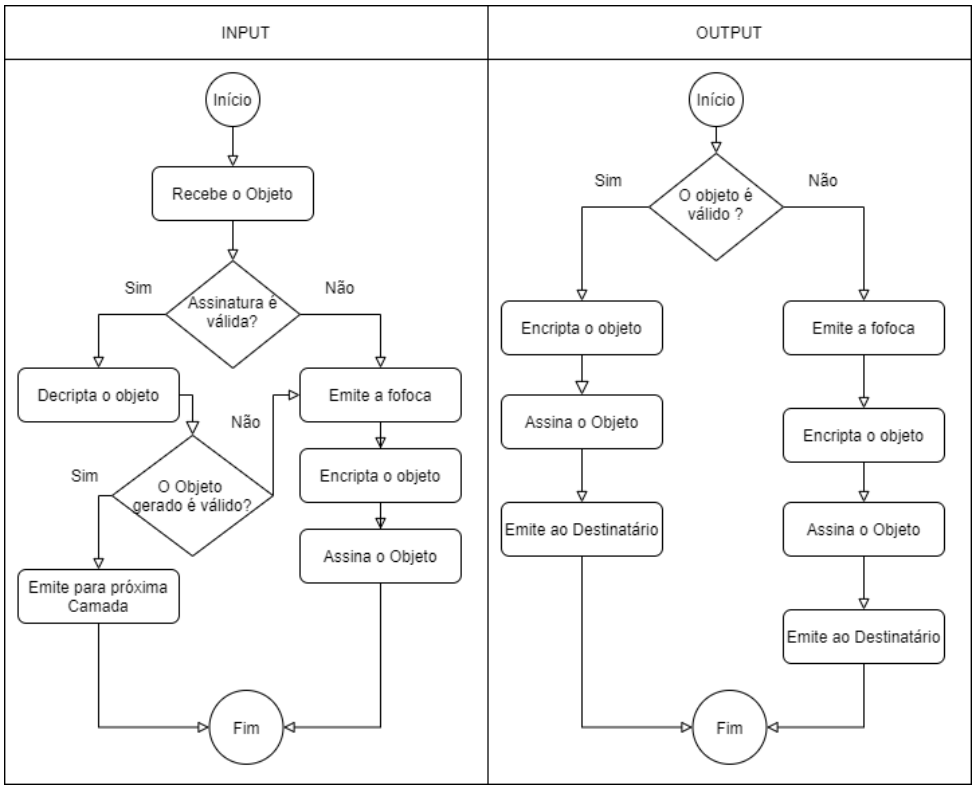
O Diagrama 10 mostra o fluxo de comunicação no *middleware* proposto. A validação de objeto (transações) é realizada diretamente na camada de entrada para que o processo termine se algum problema de criptografia ou assinatura for encontrado. É importante lembrar que, para seguir essa sequência de métodos de assinatura, criptografia e descriptografia, são recomendados métodos de criptografia de chave assimétrica, os quais têm duas chaves (uma pública e uma privada) complementares entre si, que fornecem maior segurança no processo de transmissão de mensagens. O método de chave simétrica também pode ser usado para executar esse processo de transmissão de dados, mas sua única vantagem é que o tempo de criptografia e descriptografia é reduzido no processo, mas a segurança é comprometida porque uma única chave é usada e compartilhada por usuários que precisam ver a mensagem.

4.3 Camada de Saída

A camada de saída tem duas funções principais: 1) Enviar *feedback* para a organização e 2) Mostrar status da transação na rede. Um servidor rest é usado para recuperar os eventos inseridos nos contratos inteligentes da camada de negócios. Esses eventos informam quando uma transação ocorreu no sistema e quando foi validada ou devolvida. As transações retornadas, conforme apresentadas na camada de entrada, devem ter erros de criptografia ou assinatura e podem ser refeitas ou ignoradas. O sistema pode fornecer uma visão geral da rede do status dessas transações para o usuário.

A segunda função executa o envio para os outros *clusters* de gerenciamento de rede, sobre o status das transações que estão recebendo. Como fofoca do bairro, as más notícias tendem a se espalhar pela rede, inibindo as tentativas de inserir transações com problemas identificados.

Figura 10 – Diagrama para o processo de negociação do usuário no *middleware* proposto. O processo é executado diretamente entre o remetente e o destinatário sem interferência de terceiros confiáveis. O fluxo do diagrama indica quais decisões podem ser tomadas, dependendo do estado do objeto.



Fonte: Próprio autor.

4.4 Camada de Pré-processamento

A camada de análise é responsável pelo pré-processamento dos dados após a validação e verificação da assinatura. As camadas anteriores executam esses processos de validação e verificação e inspecionam se os dados não foram alterados até atingir o destinatário. No entanto, isso não assegura a execução de contratos inteligentes que garantam sucesso com os dados recebidos ou se os dados recebidos são consistentes com o retorno de uma resposta válida.

No estágio de pré-processamento, os dados precisam ser transformados nas informações que o contrato pode receber. Como os testes são executados na camada anterior para verificar se os dados correspondem ao escopo da entrada da função do contrato, essa camada verifica se os dados representam um valor consistente com base nos dados anteriores. Para ser mais específico, foi realizado uma análise sintática e semântica para verificar se os dados representam o valor de um sensor, mas não vêm com essas informações inteiramente puras e no formato desejado (Inteiro ou Flutuante), e assim, eles precisarão ser manipulados antes da execução do contrato.

Na etapa de filtragem, os dados inválidos (não pode ser transformado ou apenas nulo) são bloqueados e o status pode ser enviado ao remetente para avisar sobre o evento inesperado.

Tabela 3 – Estrutura de cabeçalho de uma transação. A linha 1 demonstra os atributos necessários para receber uma transação ou criar uma. As outras linhas são exemplos de recebimentos de transação.

ID	**Tipo de Ativo**	**Emissor**	**Id do Contrato**	**Assinatura**	**Checked**	**Aceitação das Partes**	**Área de Dados**
1	Computer	Operational center	CON 0001	Yes	Yes	Yes	not shown
18	Smart phone	sales clerk	CON 0098	No	No	No	not shown
13	Medical Record	clinical patient	CON 0111	Yes	Yes	Yes	not shown

Fonte: Próprio autor.

Lembrando que a rede Blockchain não exclui registros de sua cadeia, e as informações validadas no contrato permanecerão na rede. Portanto, os dados devem atingir as camadas mencionadas de forma clara e válida, para que esses tipos de problemas não sejam criados. Além disso, dados inválidos na camada do contrato podem validar funções incorretas ou efetuar pagamentos e multas indevidos.

A camada de pré-processamento permite filtrar os objetos do sistema que ativarão o contrato. Nesta camada, os dados que estão dentro da transação são verificados para encaminhar a camada de análise de dados. Como visto na camada anterior, o *middleware* recebe dados de duas maneiras diferentes, na forma de transação ou como dados de monitoramento. Descrevemos cada uma dessas maneiras abaixo:

Dados de monitoramento: Os dados de monitoramento estão relacionados às informações do dispositivo do aplicativo real. Essas informações são essenciais para a tomada de decisões inteligentes sobre contratos. O monitoramento não está apenas vinculado a dispositivos inteligentes, mas também a entradas de dados de aplicativos monitorados, como revender ou atualizar o status de um componente. É essencial especificar como os dados são inseridos na rede para padronizar o tipo de informação que o *middleware* precisa para receber e associar as informações ao contrato. Ao especificar o contrato inteligente, conforme explicado na Tabela 3, cada contrato e ativo possui dados específicos que o tornam único na rede. Esses dados devem chegar de maneira clara e precisa para que o contrato seja cumprido.

Portanto, os dados de rastreamento seguem o seguinte cabeçalho:

ID: cada informação deve conter um índice direto que possa ser referenciado e facilmente acessível através da camada de “coisas”. Esse ID é o que identifica o componente, os dados do dispositivo ou as informações inseridas pelo usuário. O ID faz parte da transação e é um código de referência direta para rastrear o dispositivo ou usuário.

Tipo de ativo: a camada do contrato trata as informações como um ativo. Portanto, os

dados precisam especificar de qual componente a informação vem, para que permitisse identificar o tipo de ativo que está enviando a informação e se pode enviá-lo para o contrato especificado.

Participante emissor: o tipo de participante pode ser um dispositivo IoT, um usuário com acesso e autorização para enviar dados. A caracterização do participante também é necessária para o contrato, para que ele possa identificar se na etapa em que o contrato está sendo executado (criação, validação, inspeção).

ID do contrato: o ID é o índice direto que se refere ao contrato que possui um link para sua entrada de dados. A entrada de dados vincula um ou mais contratos e, dentro desse contrato, pode-se ativar uma função. Cada contrato tem seu ID específico para que possa ser acionado diretamente.

Assinatura: a assinatura é o escopo de entrada da chamada de função. Na camada de entrada, são observadas se ocorreram alterações no envio de dados, mas somente nessa camada é verificada se os dados podem ativar a função de contrato solicitada. A assinatura permite verificar qual contrato deve ser ativado e se ele não foi alterado quando a negociação estava sendo concluída.

Verificado: Este campo é a confirmação da camada anterior sobre o estado dos dados recebidos. Após a checagem realizada por assinatura este campo é preenchido com o aceite/bloqueio da transação.

Acordo entre as partes: Este campo determina o estado do objeto. Pode ser definido como Sim ou Não. No estado NÃO significa que uma das partes no contrato ainda não validou a transação. No estado SIM, a transação é válida e pode passar para a próxima camada para análise de dados após a validação dessa camada.

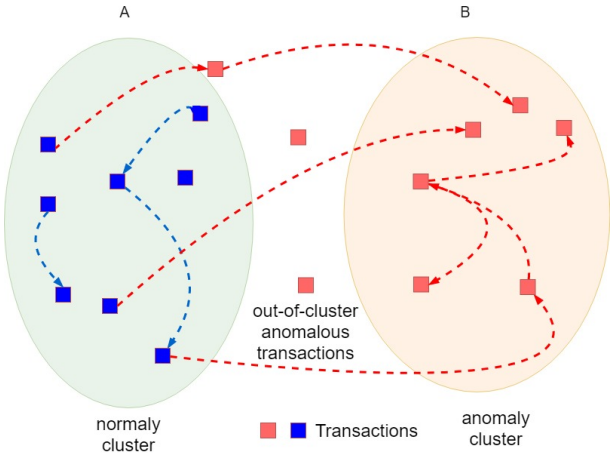
Área de dados: contém conteúdo a ser processado pelas camadas de contrato e blockchain. Os testes de ativação de contrato são executados nessa camada para verificar se está em conformidade com o especificado.

Dados de transação: Dados de transação são informações que já estão sendo negociadas e foram validadas anteriormente por uma instância do *middleware*. Quando uma transação chega com a formatação correta, verificada por outra instância do *middleware*, os dados são enviados para a camada de análise e passam pela detecção de anomalias e verificação de dados antes de fazer uma chamada de função do contrato.

4.5 Camada de Análise de Dados

A camada de análise é responsável por detectar anomalias nos dados anteriores. As camadas anteriores executam os processos de validação e verificação e inspecionam se os dados não foram alterados até atingir o destinatário, no caso a rede blockchain. No entanto, isso não assegura a execução de contratos inteligentes que garantam sucesso com os dados recebidos ou

Figura 11 – Representação dos dados para a hipótese gerada. O grupo denominado *nomaly* representa as transações válidas, usadas e criadas. O grupo *anomaly* contém as transações inválidas.



Fonte: Próprio autor

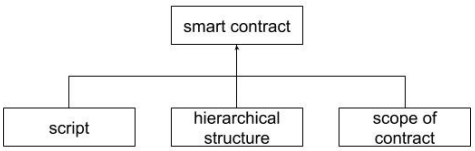
se os dados recebidos são consistentes com o retorno de uma resposta válida.

Na fase de análise histórica, o comportamento da transação e o histórico de transações deste contrato são avaliados. Cada contrato tem um comportamento que pode ser assimilado a partir dos insumos que recebe. Essas entradas, consideradas válidas, pertencem a um grupo de transações que já estão na rede Blockchain. Como pode ser visto na Figura 11, cada nova transação deve pertencer ao mesmo grupo que as transações avaliadas como corretas. Essa avaliação pode bloquear a transação, para que o administrador da rede possa avaliar a situação específica e, assim, determinar se a transação tem alguma anomalia. Para realizar essa análise, usou-se o algoritmo de agrupamento K-means e criou-se quatro grupos. O primeiro grupo, são as transações criadas. O segundo grupo está relacionado às transações válidas, o terceiro grupo às transações reusadas e o último grupo às transações inválidas. A classe com transações inválidas ou anômalas encontra possíveis transações com problemas reais. Esses problemas estão relacionados ao valor inserido ou a uma mudança repentina no comportamento das leituras do sensor, por exemplo.

Outros algoritmos foram usados no estudo de caso, mas o algoritmo baseado em *cluster* foi o mais promissor e classificado com mais certeza. No entanto, o algoritmo nessa camada dependerá amplamente do que o usuário considera uma anomalia. Considerou-se aqui uma anomalia como: 1) Uma transação que possivelmente foi alterada e passou por etapas de verificação ou; 2) Uma transação que não possui dados consistentes e, por motivos de problemas físicos, está enviando dados incorretos para a rede.

Após validar os dados nesta camada, a camada de negócios recebe os dados e executa o contrato.

Figura 12 – Representação da estrutura do contrato inteligente.



Fonte: Adaptado de (COMPOSER, 2019).

4.6 Camada de Negócios

A camada de negócios é responsável por armazenar e executar contratos inteligentes preestabelecidos de acordo com o padrão das organizações de rede (usuários) que compartilham informações. Dentro deste contrato, todos os acordos, prazos e penalidades aplicáveis em sua execução devem ser estabelecidos. Após a criação de um contrato inteligente, ele é validado pela rede e se torna imutável como transações validadas subsequentes.

A ideia do contrato inteligente surgiu em 1994 com Nick Szabo, que definiu o contrato inteligente como um *"protocolo de transação eletrônica que impõe cláusulas em um contrato"* (SZABO, 1994). Os contratos são definidos computacionalmente, respeitando todos os termos estabelecidos e são executados de acordo com a demanda de dados, o recebimento e a emissão temporal de datas e termos.

Os contratos inteligentes seguem uma estrutura IF-ELSE, mas sua implementação pode variar dependendo do aplicativo. Para implementar um contrato inteligente, você deve seguir uma estrutura básica que respeite a criação e execução de contratos. Muitas redes de blockchain que implementam contratos têm sua linguagem de programação e como criar o esqueleto do contrato. Para implementar um contrato inteligente, um esquema e uma estrutura de contrato são propostos abaixo.

Os contratos inteligentes têm um ciclo de vida diferente do blockchain. Por ter a característica de ser imutável, o contrato inteligente deve ser bem estruturado e implementado corretamente, especialmente com alguma forma de fechamento quando o contrato expirar. Além do contrato inteligente, essa camada implementa estruturas que definem ativos, participantes e transações e eventos de contratos. Essa camada define como os ativos são estruturados, a definição de participantes e sua hierarquia de acesso e a estrutura de contratos inteligentes (transações e eventos). Como você pode ver na Figura 12, o contrato inteligente pode consistir em três arquivos principais: script, hierarquia de acesso e escopo do contrato.

- Escopo do contrato: a primeira estrutura contém todas as definições da camada de contrato. O ativo é definido, assim como seus atributos armazenados na sequência imutável. Na rede blockchain, os atributos a serem armazenados precisam ser considerados sensíveis, isto é, eles devem ser muito cuidadosos para que as alterações não sejam feitas ao longo do

tempo. Após a definição do ativo, a estrutura precisa conhecer o escopo das transações que compõem o contrato inteligente e deve-se definir os atributos que acionam a execução do contrato. Por exemplo, em uma locadora de vídeo *streaming*, para alocar um filme, a locadora deve inserir o nome e o ano do filme para ativar o contrato inteligente e efetuar o pagamento automático. Esses dois dados devem estar corretos para acionar o serviço de concessão de contrato. Assim, quando o usuário digita esses dados e chama a função, o contrato executa o pagamento e ativa o serviço de *streaming*. Por fim, a estrutura pode definir alguns eventos que alertam os participantes do contrato sobre determinadas situações previstas no contrato.

- Script: A segunda estrutura do contrato contém todas as funções descritas como uma programação. Geralmente, essa programação traz em termos práticos as cláusulas estabelecidas no contrato. O idioma a ser utilizado depende apenas da plataforma escolhida para implementar os contratos (*Hyperledger, Ethereum*) ou de uma plataforma privada desenvolvida pelos usuários da rede.

- Hierarquia de acesso: a terceira e a última estrutura é a hierarquia de acesso ao contrato inteligente. Como na primeira estrutura, os participantes do contrato foram definidos e, para esses participantes, é necessário definir os níveis de acesso e quais funções podem ser acessadas por cada um. Se esse arquivo não estiver bem definido, o contrato poderá estar sujeito a execuções de usuários não autorizados ou mesmo à execução de funções que não foram planejadas para serem acionadas por usuários (funções internas do contrato).

Em blockchains públicas, o acesso é gratuito para todos os usuários. Todos podem validar um contrato ou participar de um contrato. Além disso, esse contrato é visível para todos na rede blockchain que foi armazenada. No consórcio blockchain, um grupo de validadores autoriza o armazenamento de contratos e gerencia as transações executadas nesses contratos posteriormente. Pode-se também inserir uma hierarquia de acesso e criar pequenas redes 'privadas'. No blockchain privado, um gerente autoriza a criação de contratos e verifica a execução de cada função. Os usuários têm permissões bem definidas e não podem participar do processo de validação da rede.

O *Hyperledger* implementa essas três estruturas e o administrador que cria um contrato e deve definir todas as restrições de acesso quando elas existirem. Um grupo de organizações predefinidas valida o contrato e o armazena em um bloco na rede. Quando esse contrato é criado, os participantes deste contrato recebem uma referência às funções que podem ser executadas para ativar o contrato. Assim, o processo de execução do contrato pode começar normalmente.

No *Ethereum*, os usuários podem definir contratos em várias linguagens de programação, que usam o sistema "Prova de trabalho". Isso permite que a rede chegue a um acordo sobre o estado de todas as informações registradas no *Ethereum* e evite certos tipos de ataques

econômicos. No entanto, a validação está sendo alterada nos testes de apostas para reduzir o tempo de validação.

4.7 Camada de Rede Blockchain

Na seção 2.3.2, define-se a estrutura básica de uma rede blockchain. Nesta seção do *middleware* MADCS, descreveu-se como diferentes tipos de blockchain podem ser usados, sempre dependendo de qual aplicativo está sendo implementado. Seguindo o diagrama 10, para cada tipo de aplicativo, uma rede blockchain pode ser mais útil. O fator **Performance** vs **Safety** é um ponto importante a ser considerado nesta camada.

A rede pública de blockchain se tornou uma das redes mais seguras do mundo para negociar moedas virtuais como o Bitcoin. No entanto, o recurso dessa rede distribuída é que ela é pública para qualquer usuário que queira participar do processo de mineração e do processo de compra e transferência de dinheiro. Os usuários têm sua identidade confidencial e trocam informações, geralmente sem estar cientes dos outros.

A blockchain do consórcio pode ser considerada um compromisso entre a blockchain pública e a blockchain privada. Seu processo de consenso é realizado por um grupo de organizações específicas e previamente selecionadas. No entanto, a participação na rede pode ser pública ou privada. Pequenos grupos podem criar contratos inteligentes ou estabelecer ativos e incorporar privacidade a essa rede. Podemos considerá-los como blockchains reconfiguráveis.

O blockchain privado, por outro lado, tende a ser central no processo de validação de transações e contratos inteligentes, além de fornecer permissões mais robustas e específicas. O processo de consenso é exclusivo de uma organização, o que melhora o desempenho, mas o processo de auditoria não é aberto aos usuários da rede.

Independentemente de qual blockchain for escolhida, os registros de contratos e transações são registrados no razão. Antes do registro, os mineradores realizam o chamado processo de consenso. Esse processo de consenso da camada depende do blockchain escolhido. A blockchain de primeira geração, como a blockchain Bitcoin e Ether, usa o protocolo de consenso à prova de trabalho. Este consenso de 51 % da rede de acordo com o bloco a ser inserido na rede para concluir o processo. Esse processo pode levar a problemas de desempenho e atrasos na validação de transações na rede porque o tamanho do bloco é limitado. Nas blockchains 3.0 que implementam contratos inteligentes, o processo de consenso para a validação de transações e contratos é diferente e apresenta melhor desempenho. No entanto, os projetistas de contratos precisam estabelecer políticas e regras de privacidade mais robustas.

Blockchain ou blockchain 3.0 tem limitações com o problema de armazenamento devido à sua interconexão e armazenamento em bloco, conforme explicado na seção sobre tipos de blockchain. Uma solução é avaliar o tamanho dos dados confidenciais que precisam ser armaze-

nados e, se forem muito grandes, como um conjunto de imagens e documentos, o blockchain pode armazenar uma referência para onde o documento está armazenado. Dessa forma, um banco de dados separado é usado para armazenar as informações localmente, e a rede blockchain armazena a referência e todas as transações feitas nesse arquivo. Esse conceito é conhecido como fora da cadeia. Assim, o blockchain é usado como um log de eventos para gerar auditabilidade dos arquivos compartilhados por esses bancos de dados.

Esse *middleware* requer uma rede Blockchain que implemente a camada de contrato inteligente definida anteriormente para executar todas as etapas e registrar as transações adequadamente. O usuário recebe *feedback* das transações na camada de saída para verificar se o processo foi executado com sucesso.

4.8 Conclusão

Após a formalização de cada camada, temos os seguintes aspectos qualitativos para melhor explicar como essa arquitetura em camadas irá auxiliar na utilização em diferentes contextos e tipos de redes blockchain. Um dos pontos importantes é a modularização. Este modelo permite a mudança de tecnologias em todas as camadas desde que o formato de comunicação não se altere entre as ligações que conecta uma camada a outra. Dessa forma, caso a camada de filtro e análise tenham que ser alterados os algoritmos ou a forma de análise da transações, dado o contexto em que está sendo inserida, o MADCS não precisará de alterações nas demais camadas. Outro ponto importante é que todas as camadas que formam o MADCS tendem a ficar próximas ou entre a camada de armazenamento de dados do blockchain e os sistemas de comunicação dos usuários. Assim, o sistema de detecção não ficará nas aplicações da borda, essencialmente as aplicações diretamente ligadas a interações dos usuários, mas sim próximo a rede blockchain e os *peers* que recebem, processam e criam os blocos. Para exemplificar a utilização do *middleware* em diferentes contexto, utilizou-se um estudo de caso fora do contexto de criptomoedas. Nesse estudo de caso foi utilizado informações de usuários de uma rede de saúde, que possui informações sensíveis fora do contexto econômico.

CAPÍTULO

5

ESTUDO PILOTO

Este estudo piloto trata-se de um gerenciamento com prescrições médicas. Toda a implementação do modelo proposto pode ser encontrado no repositório (SILVA, 2020). A prescrição ou receita médica é um documento que habilita a retirada de alguns medicamentos nas farmácias que precisam da indicação do profissional para que o paciente possa realizar o tratamento. Na Figura 14, é apresentado o diagrama de casos de uso para instanciar o modelo para o estudo piloto.

Figura 13 – Diagrama de casos de uso do estudo piloto proposto.

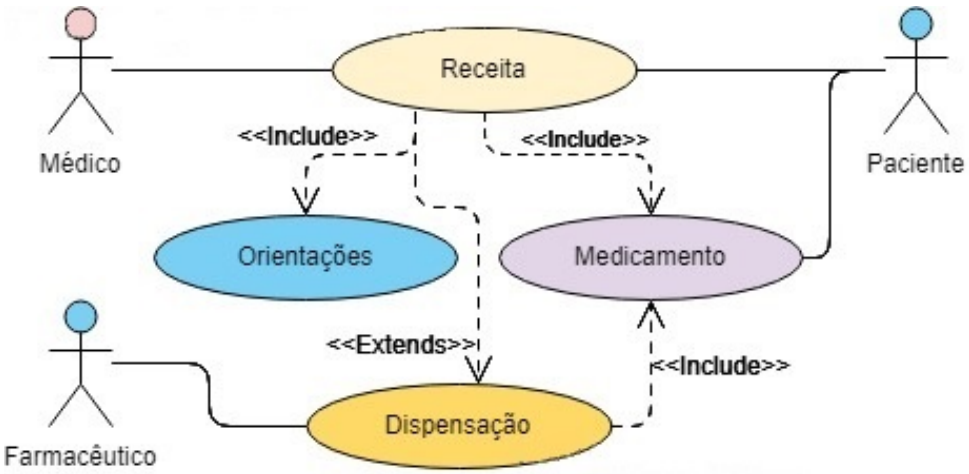


Figura 14 – Fonte: Próprio autor.

5.1 Detalhes de Implementação

Para validar o *middleware*, selecionou-se uma instituição pública que servirá como um caso de uso para implementar e validar o *middlewar* proposto. Novas leis sobre a proteção e a privacidade dos dados dos clientes foram implementadas em todo o mundo para determinar

a responsabilidade das empresas pelas informações confidenciais dos clientes. Em um caso específico, as prescrições médicas em alguns países se tornaram válidas em todo o território nacional, o que pode dificultar a validação das prescrições quando um cliente está indo à farmácia. Embora facilite muito o atendimento ao paciente e a administração de medicamentos, há uma dificuldade em verificar se o documento é válido, pois em cada estado pode haver particularidade no documento que não é visto em outro lugar.

Para este estudo piloto, temos o seguinte cenário: Uma instituição deseja substituir sua prescrição médica em papel por uma prescrição eletrônica. Essa prescrição contém dados confidenciais do usuário, como exemplos de identificação que podem ser associados a registros médicos e revelam diagnósticos do paciente. A instituição possui um grupo de hospitais que desejam gerenciar suas prescrições, mas precisam manter o compartilhamento entre eles, garantindo todos os problemas de privacidade dos usuários. Embora o grupo de hospitais seja conhecido um do outro e supostamente confiável, as empresas que fornecem medicamentos prescritos não são e, portanto, os registros de prescrição precisam ser gerenciados para que eventos indevidos não ocorram. Esses eventos podem ser possíveis fraudes, apropriação indébita e uso de prescrições por médicos não autorizados.

Analisando a situação descrita acima, implementou-se cada camada do *middleware* para atender às especificações necessárias. Para implementar as camadas de negócios e a rede blockchain, escolhemos a geração de blockchains 3.0 que implementam contratos inteligentes. A partir desses blockchain 3.0, chegou-se à conclusão de que o *Ethereum* e o *HyperLedger* são mais maduros em suas plataformas e fornecem flexibilidade para a criação de contratos inteligentes. Embora haja alguns problemas com a execução do contrato, a camada de análise verifica cada etapa para garantir que a rede esteja se comportando da melhor maneira possível. Para este estudo piloto, usou-se o *HyperLedger*, pois é um consórcio blockchain. Então, pensou-se em criar um ambiente gerenciável para hospitais, que precisa verificar e auditar os processos de distribuição de medicamentos. Assim, o processo de consenso está ciente das organizações, neste caso, hospitais, mas o acesso à comunidade não é privado. Isso torna o processo mais rápido em comparação com as blockchains tradicionais e cada instituição pode implementar seu contrato para validar a prescrição médica que gera. Este contrato será armazenado de maneira distribuída e facilmente acessível.

5.2 Camada de Entrada

Nessa camada, os dados do paciente e do médico são fornecidos por prescrição. A prescrição contém dados de orientação do farmacêutico para o paciente sobre a medicação, quantidade, uso e informações gerais do médico e do paciente. Inicialmente, concentrou-se apenas na prescrição médica como um ponto de estudo para avaliar a detecção de fraude, mas o modelo pode abranger mais funções como cadeia de suprimento, registros médicos e serviços

hospitalares prestados. Também registrou-se a data e a hora do conteúdo criado, que servirá como uma verificação da vida da receita. Cada prescrição terá uma data de validade, uma data de validade e um limite máximo de uso, o que significa que o usuário não precisará sempre ir ao médico somente se a prescrição tiver sido amplamente usada. Todas as regras de informação e expiração e a duração da prescrição são gerenciadas pelo contrato inteligente criado.

Um registro de data e hora foi integrado no ativo, para verificar a vida útil da prescrição. Cada prescrição terá uma data de vencimento, uma data de validade e um limite máximo de uso, para facilitar ao usuário que precisa buscar periodicamente os remédios, sem precisar ser consultado pelo médico novamente. De acordo com a Figura 16, cada prescrição instanciada no blockchain terá um Time To Live que será decrementado para cada evento de dispensação até atingir zero. O campo de data de vencimento será usado pelo contrato inteligente para garantir que a prescrição expire se exceder o prazo estabelecido.

Figura 15 – Modelo de prescrição médica para criação do ativo.

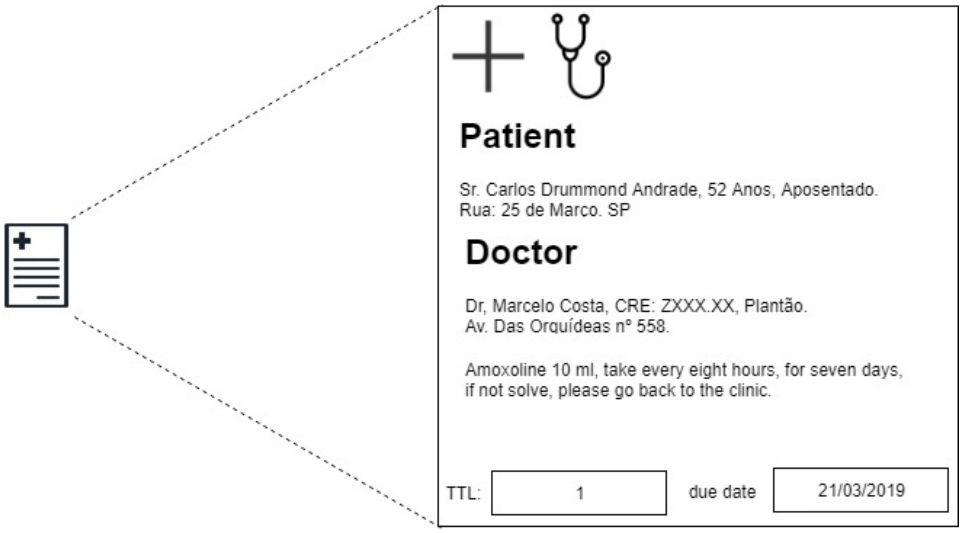


Figura 16 – Fonte: Próprio autor.

5.3 Camada de Pré-processamento

Depois de instanciar todos os objetos da camada de usuário, temos o *middleware* de negócios pronto para ser instanciado no *HyperLedger*, que será o *host* para implementar a rede blockchain e os *Smart Business Contracts*.

O *Hyperledger* possui quatro tipos de arquivos para criar o modelo: Modelo, Arquivo de *Script*, Controle de Acesso e Consulta (COMPOSER, 2019). Com o modelo (.ctl) pode-se criar todos os ativos e configurar uma cadeia de blocos de rede. Dados de ativos e contratos inteligentes devem ser descritos nesta etapa para gerar o *middleware* que será usado para instanciar os componentes do sistema, por exemplo, um participante (médico ou paciente), um ativo no caso de prescrição e os contratos que estabelecem as regras . O arquivo de *script* (.js)

é onde configurou-se todas as ações do sistema e implementou-se como o contrato deve se comportar antes de cada transação. Assim, todas as regras são descritas no formato de idioma e depois executadas. Na ACL, podemos controlar a permissão de cada usuário no sistema, mas, como trabalhou-se com prescrições médicas, o paciente ficou com a maioria das ações prescritas e, nesse ponto específico, a camada de análise lidou-se com a validade da transação e se nada foi alterado no documento. Finalmente, a consulta (.qry) é onde podemos capturar os dados da rede. Um servidor de descanso é criado e podemos consumir dados de blockchain e realizar mineração e análise de dados.

5.4 Camada de Negócios

A camada de negócios trata-se do ambiente onde o contrato inteligente é configurado e desenvolvido para atuar nas modificações e certificações das novas transações que chegam no modelo de negócio. Conforme a criação dos contratos, diferentes organizações são vinculadas conforme o interesse, e o contrato tem a função de executar os critérios preestabelecidos para aplicar as penalidades e execuções de cláusulas. O contrato inteligente está dentro da camada de Blockchain e vem após a camada de Pré-processamento. Foi estabelecido que o produto capaz de avaliar e detectar anomalias na camada de pré-processamento fique junto com o contrato e conforme o contrato muda o modelo também será atualizado, mantendo a base histórica como base para encontrar os pesos necessários. Dessa forma, garantimos que o modelo fique dentro da Blockchain e não com cada usuário e o modelo antecipadamente antes da execução de um contrato. Isso permite avaliar a transação antes de ser executada e antevir a ações maliciosas; Então, após a validação a transação deverá ser executada no contrato conforme as clausulas pré-estabelecidas entre as partes interessadas e prosseguir com o andamento das transações. Logo abaixo é mostrado o algoritmo para realizar a dispensação farmacêutica 1 das prescrições médicas.

No cenário como um todo, o médico irá atender ao paciente e poderá prescrever os medicamentos. Caso seja necessário a prescrição é criada pelo médico e vinculado ao usuário. A partir desse momento o usuário tem em sua *wallet* o vinculo com a prescrição farmacêutica que fica em sua posse. Posteriormente, o usuário poderá realizar a dispensação conforme o número de vezes que foi possibilizada e também pela data máxima de validade da prescrição preestabelecida no contrato. Para a dispensação, o farmacêutico deverá receber a autorização do usuário para ver a prescrição médica em seu sistema e, dessa forma, concluir a dispensação após o *middleware* e o contrato inteligente aplicar a ação positiva na prescrição. O Farmacêutico não poderá aplicar dispensações para usuários sem prescrição médica, a não ser que o medicamento seja permitido aplicar ou vender sem autorização. Após a dispensação e conforme as regras do contrato, o usuário poderá reutilizar a prescrição sem retornar novamente ao médico. Com isso o processo agiliza as prescrições de medicamentos e o usuário não irá parar de utilizá-las, como por exemplo pessoas com diabetes, epilepsia e outras doenças que não têm uma cura mas podem

ser periodicamente controladas por medicamento.

Algoritmo 1 – Algoritmo de Dispensação

```
1: procedimento PAYOUT(DispencingReceived)                ▷ prescrição para dispensação
2:     presc ← DispencingReceived.prescription
3:     cont ← DispencingReceived.contract
4:     payOut ← 0
5:     se ((presc.issueDate.getDate() ≤ cont.minDate.getDate())and
   (presc.dueDate.getDate() ≥ cont.maxDate.getDate())or
   (cont.maxUsePrescription − presc.numberOfUses) ≤ 0) então
6:         presc.status ← INVALID
7:         payOut ← payOut − cont.minPenaltyFactor × presc.numberOfUses
8:     senão
9:         se ((presc.dueDate.getDate() ≥ DispensingReceived.timestamp.getDate()))
   então
10:            presc.status ← INVALID
11:            payOut ← payOut − cont.minPenaltyFactor × presc.numberOfUses
12:        senão
13:            se (presc.status == VALID  or  presc.status == INVALID) então
14:                se (presc.status == VALID) então
15:                    presc.status ← USED
16:                fim se
17:                se (presc.status == INVALID or ((cont.maxUsePrescription −
   presc.numberOfUses) ≤ 0)) então
18:                    payOut ← payOut − cont.maxPenaltyFactor × presc.numberOfUses
19:                fim se
20:            senão
21:                se (presc.status ≠ USED) então
22:                    presc.status ← VALID
23:                fim se
24:            fim se
25:        fim se
26:    fim se
27:    presc.numberOfUses ← presc.numberOfUses + 1
28:    se ((cont.maxUsePrescription − presc.numberOfUses) ≥ 0) então
29:        payOut ← payOut + (cont.maxUsePrescription − prescr.numberOfUses) ×
   cont.minPenaltyFactor
30:    fim se
31:    DispensingReceived.patient.accountBalance+ = payOut
32: fim procedimento
```

5.5 Camada Blockchain

A partir da organização feita acima, implementou-se cada arquivo da estrutura necessária para gerar nossa rede de negócios. Nesta etapa, usou-se o *composer* para configurar a rede e realizar os testes para gerar os dados e treinar o *cluster* para a camada de análise. Nesta etapa,

temos a interface da web criada para registrar as transações, os contratos de instância e os participantes da rede de negócios. Usou-se o *HyperLedger composer* e também o servidor de *rest* que pode ser gerado a partir do aplicativo e, portanto, consome os dados da rede blockchain.

5.6 Camada de Análise de Dados

Para análise dos dados gerados na camada blockchain, explorou-se um algoritmo baseado em *cluster*, seguindo o cenário e a hipótese:

- Dado o conjunto de transações que seguem uma **normalidade** entre elas, ou seja, transações que passam por todo o processo de criação, compartilhamento, validação e finalização e que não mudam ao longo da vida útil da rede blockchain, é representado por um domínio conhecido como transações regulares.

- Transações que foram alteradas ou modificadas ao longo do tempo, talvez em torno dos *clusters*, com a possibilidade de ter um *cluster* que represente os estados das transações e aponte aqueles que são anômalos.

- Regras do contrato: incorporou-se as regras dos contratos inteligentes de acordo com o documento que especifica as prescrições médicas. Infelizmente, cada país e até cada estado de um país contém sua própria especificação de como o documento para a prescrição de medicamentos deve ser seguido. Assim, focou-se no design básico do contrato, mais comum entre esses documentos, para tornar a especificação mais geral, pensando em criar um projeto que possa ser estendido a diferentes tipos de prescrições. Nestas regras básicas, verificou-se a data de validade das prescrições, dados de médicos e usuários, diretrizes, medicamentos prescritos e número de usos que a prescrição pode exercer. O administrador pode estabelecer regras para cada prescrição ou grupo de prescrições. Essas prescrições podem permanecer na carteira do cliente, para que ele se sinta mais à vontade, com seus dados. No entanto, o *middleware* será responsável por verificar se as alterações não foram feitas nesse período.

A Hipótese é que o algoritmo de clusterização pode detectar essas transações que foram modificadas por algum usuário (externo ou proprietário), através de uma análise das informações internas da transação e como ela varia no tempo em que ocorreu a próxima transação. Isso gera um sistema de alerta, que avisa o usuário de qualquer tipo de fraude sobre o conteúdo que está na transação e se esse conteúdo sofreu modificações anteriormente.

K-means foi o algoritmo escolhido para este primeiro teste. Algoritmos de clusterização são bem utilizados para identificar a similaridade entre informações, como no mercado financeiro, identificar quais perfis de usuário compram uma determinada marca e assim por diante.

Como a rede produz dados multivariados, tem-se a perspectiva de que com o algoritmo de clusterização K-means, pode-se avaliar e detectar a anomalia e dessa foram, confirmar a hipótese dado o contexto de uma rede blockchain. Considerou-se que os algoritmos de clusterização são usados para estruturas subjacentes (para análise de dados, detecção de fraudes ou identificação de recursos), classificação natural (identificar similaridade) ou para compactação.

O método de Elbow para encontrar o número apropriado de *clusters* no conjunto de dados de teste. Este método examina a variação percentual explicada como uma função do número de *clusters*. A escolha do número de *clusters* é importante para visualizar a variação dos dados e reduzir a busca por componentes, eliminando aqueles que não apresentam alterações ao longo do tempo. A curva ou o cotovelo, como é chamado nesse método, informa quando um número de *clusters* atinge a quantidade máxima de informações que eles podem adicionar ao método.

5.7 Camada de Saída

Nesse nível, o médico, o farmacêutico e o paciente podem acessar os serviços de monitoramento, visibilidade de transações, contratos inteligentes e criação de transações.

- **Criação de participantes**: Cada participante terá acesso ao que é permitido e descrito no arquivo de permissões. Para executar esse gerenciamento, a ferramenta *Hyperledger Composer*, usada na camada blockchain, conforme descrito na camada de negócios. Isso é feito através de cartões de identificação, que são uma coleção de arquivos contendo todas as informações necessárias para permitir que um participante se conecte a uma rede comercial (COMPOSER, 2019).

- **Prescrição**: O médico inserirá os dados da prescrição: ID do pedido, tipo (medicamento), status, quantidade, diretrizes a serem seguidas, data de validade e o contrato ao qual será associado. O contrato definirá o tempo máximo de uso da receita médica.

- **Distribuição Farmacêutica**: O farmacêutico terá acesso à visualização "Prescrição", onde pode acessar as informações da prescrição, acessar a data de validade e o número de vezes que foi usada. Cada dispensação será monitorada e registrada diminuindo o tempo de vida da prescrição. Além disso, o status da prescrição indicará sua situação atual.

5.8 Conclusão

Toda a estruturação da implementação do modelo proposto teve uma influência do tipo de blockchain escolhido para compor a camada de Blockchain. A rede utilizada possui código aberto para gerar, editar e excluir componentes em toda a parte, além de permitir total configuração dos peers e do algoritmo de roteamento e escolha do líder da rede para distribuir as transações. A grande dificuldade está em entender todo o processo bem como as tecnologias a linguagens

utilizadas, o que dá um nível técnico maior para trazer a rede a execução completa. Porém, isso permitiu colocar o MADCS entre a validação e criação dos blocos e a comunicação do usuário com a rede. Dessa forma, pode-se interceptar as transações, analisar e filtrar os dados antes de executar o contrato e enviar os resultados para inserção do bloco. Isso contribuiu para a redução de dados errôneos dentro da rede, evitando assim, uma propagação desse erro para os dados futuros que influenciaria em novas execuções de contratos com dados inválidos.

Dois fatores importantes foram a camada de entrada adaptada para prescrições médicas e a autoridade dos dados para os usuários. A implementação trouxe um fator importante que é deixar a responsabilidade do dado para o usuário, ou seja, que ele fosse o detentor da sua informação. Dessa forma, para empresas externas que quisessem ter acesso as informações desse usuário, teriam que ter autorização para visualizar esses dados. Com isso, a camada de análise teve que estar preparada para checar os dados com o teste de conhecimento zero, para saber se a informação foi modificada ao longo do tempo. Ainda assim, dado a possibilidade de o usuário consegui enganar essa prova, a camada de análise e filtro tiveram o papel essencial de verificar a informação com base nos dados históricos e encontrar possíveis anomalias dessa transação. Com isso, foi possível reduzir os erros e gerenciar o estudo piloto proposto com base nos seguintes experimentos que serão apresentados no capítulo a seguir.

CAPÍTULO

6

EXPERIMENTOS

Para avaliar o modelo proposto, foi utilizada a rede *Hyperledger*, com a criação de nós e a atualização dos blocos da cadeia, além da especificação do contrato inteligente. Para avaliar a rede, implementou-se a camada de *middleware* responsável por capturar dados históricos e por enviar e impedir que transações incorretas entrem no processo de verificação da rede.

6.1 Configuração

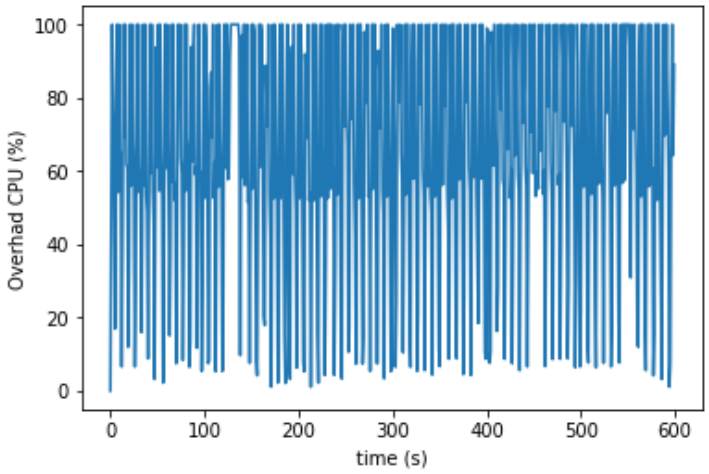
Para realizar esses experimentos, foram utilizadas 4 instâncias do modelo e a infraestrutura do laboratório Intermídia do Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo. A configuração dos computadores é: 16 GB de memória RAM, HD 1T, Processador: Intel core i7 sétima geração, placa de vídeo de 4 GB e o sistema operacional Ubuntu 17.4. O *middleaware* foi desenvolvido no Python 3.x e consome os dados de um API Rest criado para cada instância. As outras camadas de entrada e saída foram criadas com o próprio compositor do *HyperLedger* e alteradas conforme necessário. As outras camadas do Blockchain e Contratos também foram feitas com o compositor.

6.2 Avaliação de Performance

Como pode ser visto na Figura 17, a utilização da CPU nos pares foi usada em 100 % na maioria dos picos. Esse pré-processamento é esperado, no entanto, para ajudar nisso, não atualizou-se o *cluster* para validar a transação e realizar o consenso, pois esse processo pode ser feito em segundo plano no momento em que a CPU estiver mais ociosa. Com a grande quantidade de requisições e transações criadas a rede blockchain acumulou mais transações do que estava criando blocos, o que trouxe mais picos de processamento nos peers para resolver os blocos e avaliar as demais transações. Ao mesmo tempo que um bloco é criado, o aceite é feito

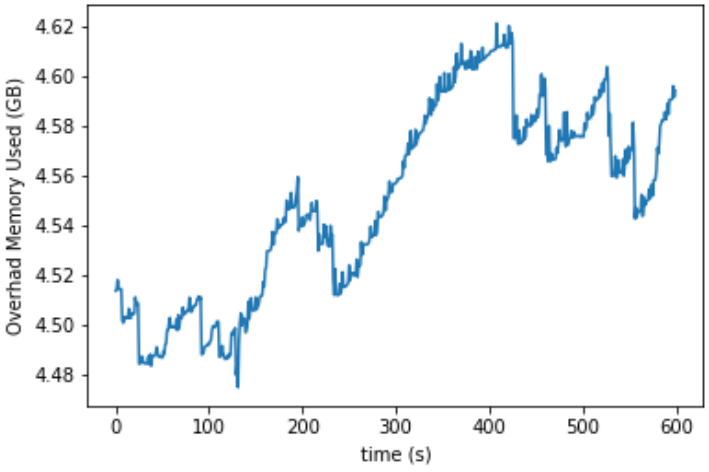
por toda a rede e ocorre a sincronização dos dados do banco nos demais peers o que gera mais uma carga de CPU.

Figura 17 – Análise média da sobrecarga da CPU dos pares.



Como mostra a Figura 18, o consumo de memória não atingiu o limite, mas houve um aumento com a introdução de clusters, o que não mudou significativamente o desenvolvimento da rede. Apesar do acumulo de transações citadas acima, não impactou no uso de memória que o blockchain utiliza. Após a resolução e criação do bloco, as requisições que salvam os dados no banco e distribui para os outros peers são executados rapidamente liberando o espaço em memória do sistema.

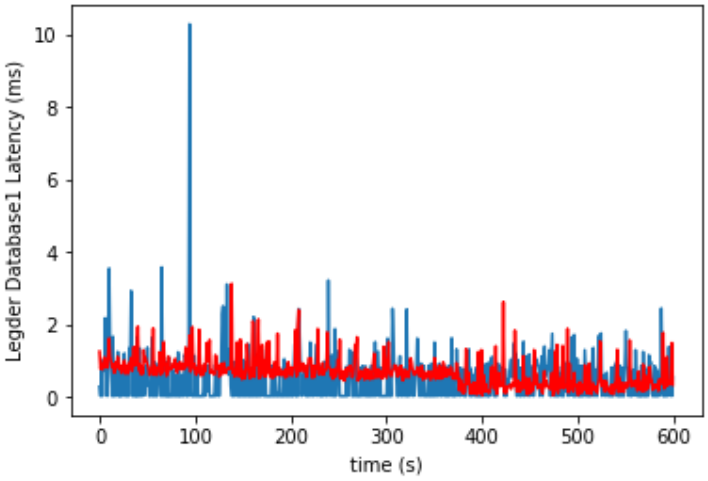
Figura 18 – Análise de sobrecarga de memória média dos pares.



Ao comparar a latência com e sem o *middleware* implementado na rede, mostrado na Figura 19, podemos observar um aumento no tempo de resposta, pois as camadas intermediárias no processo não permitem que o fluxo ocorra anteriormente. No entanto, o aumento da latência não afeta a rede como um todo, se levarmos em conta a redução de dados e as tentativas de contornar a rede. O aumento da latência está diretamente ligado a dois subprocessos adicionados

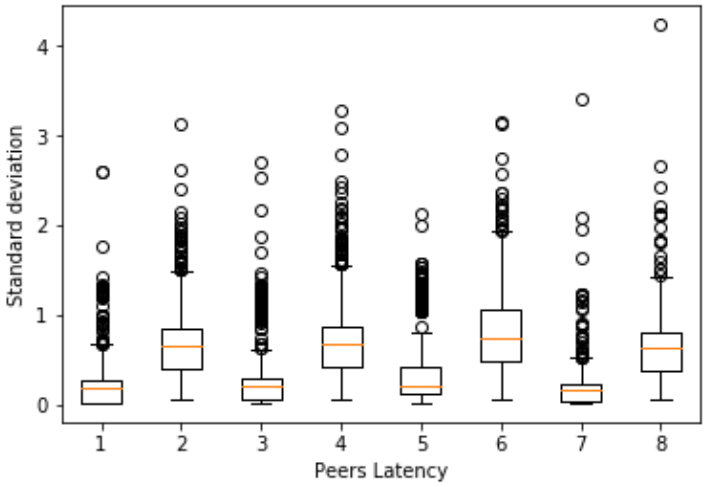
na rede *blockchain*: I) pré-processamento e II) detecção da anomalia. Na primeira etapa, os dados são transformados para a base numérica que é requisitada pelo modelo para realizar a detecção de anomalia. Na etapa de detecção, O modelo está em memória pronto para responder o estado da transação, com base na entrada enviada. Ao ter um *feedback* negativo é preciso emitir um alerta de mensagem *broadcast* para os outros *peers*, de modo a encaminhar o problema para a rede e impedir novos ataques. Com isso, a quantidade de mensagens enviadas, somado as duas nova camadas agregam um valor a mais de latência sobre a rede *blockchain*, o que na maioria dos casos em que a transação está permitida para prosseguir não ocorre.

Figura 19 – Comparação da latência com e sem *middleware* em um indivíduo semelhante.



Para avaliar de uma forma geral, a Figura 20 demonstra os resultados obtidos na análise de tempo realizada nos pares. No eixo X, um conjunto de medianas são apresentadas com o tempo de todos os pares sem o MADCS (ímpares) e com o MADCS executado (pares). Como era esperado, o repentino aumento é dado pela avaliação das camadas antes de chegar ao contrato inteligente. Porém, esse tempo é compensado pela quantidade de transações inválidas que puderam ser encontradas antes de enviar a camada de execução do contrato. Dado que o treinamento do algoritmo de detecção não será feito a cada detecção, o mecanismo para atualizar os *clusters* não interferirá na rede como um todo. Este gráfico também mostra a latência geral dada um exame de carga realizado com as transações criadas em simulação. Um ponto importante que pode ser observado é que a arquitetura proposta não traz instabilidade no sistema. Apesar do aumento da latência, se permaneceu estável em todos os peers, sendo um tempo médio fixo adicionado em relação a rede tradicional.

Figura 20 – Comparação de latência com e sem *middleware* para todos os pares.



CAPÍTULO

7

RESULTADOS E DISCUSSÃO

Nesta seção são apresentados os resultados e uma discussão geral do *middleware* desenvolvido. Como foi observado, o modelo de transações distribuído, o Blockchain, possui diversos pontos positivos e negativos, sendo um deles a imutabilidade. Este aspecto permite ter confiança na cadeia que armazena as transações e permite um controle de forma distribuída desta cadeia, mesmo tendo várias cópias, é possível saber qual é a correta cadeia a ser seguida. Porém, se a entrada de dados for suficientemente esperta para mascarar um erro, que passa pelo consenso, o problema pode ser armazenado e desencadear erros futuros, que ficarão registrados na cadeia. Um desses problemas por exemplo é o gasto duplo, em que usuários tentam enviar mais de uma vez sua transação e duplicar o valor que possui para repassar a diferentes usuários, aproveitando-se do tempo que a rede precisa para validar as transações e da distribuição da rede. Muitas soluções foram pensadas para gerenciar e resolver este problema, mas uma camada responsável por verificar problemas entre a camada de entrada e o Blockchain não foi proposta. Assim, o *middleware* foi desenvolvido com a intenção de controlar de uma forma intrínseca e sem a opinião de usuários externos, sobre o estado das transações que ativam funções do Blockchain, contratos e desencadeiam ações na rede.

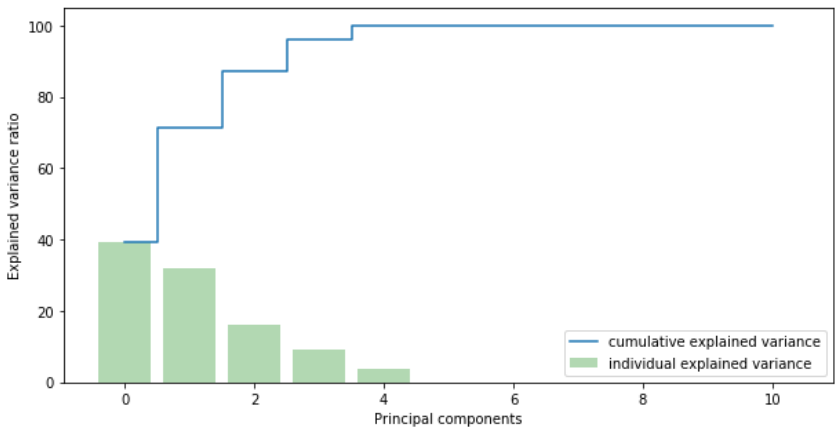
Primeiramente concentrou-se em testar a camada de análise para avaliar o comportamento do algoritmo perante o formato das transações de entrada. Como o estudo de caso é sobre prescrições médicas e dispensação farmacêutica, avalou-se as transações que são necessárias para realizar essa atividade dentro de um sistema distribuído como o blockchain. Para isso, foi realizado um estudo empírico simulando dados de prescrição obtidos de um banco de dados Kaggle (ANALYTICS, 2018). Esses dados foram utilizados para criarmos um cenário de simulação para criar prescrições, validar e transacioná-las pela rede. Essa simulação incluiu dez participantes, dez médicos e dez farmacêuticos. Criou-se oito tipos de contratos inteligentes com regras padrão e regras diferentes, para que cada empresa tenha seus regulamentos definidos para a data de validade e a duração das prescrições. Transformou-se os dados no formato que é lido na estrutura do blockchain, com os verificadores de estado sendo modificados por transações. Por

fim, os pedidos foram criados e registrados, conectando o paciente à prescrição e prescrição do respectivo contrato estabelecido por um médico. A princípio obteve-se um total de 80 prescrições médicas para iniciar a simulação. A simulação obteve um total de 2500 transações, cada pedido sendo executado pelo menos oito vezes cada prescrição, testando as condições do contrato.

Na simulação, os pacientes passam pelos farmacêuticos para retirar o medicamento e usar a prescrição uma vez. Nesse estágio, dos oitenta pedidos, 40 estavam incorretos ou com informações inválidas daqueles estabelecidos no contrato e obtiveram status inválido. Na segunda instância, as prescrições inválidas foram alteradas para tentar violar a regra do contrato ao tentar usá-la em outra farmácia. Como o pedido pertence ao usuário, o risco de ser alterado ou duplicado para tentar burlar as regras do contrato é alto. Embora o contrato detecte informações que não sejam de regra, se essas informações foram alteradas antes da transação ser aprovada, não há como ver esse evento e acabar validando os negócios com as informações que eles possuem. Após essa simulação, obtemos as 2.500 transações geradas em quatro momentos diferentes. Mantemos esse tamanho para avaliar a detecção repentina de alterações por um curto período, verificando se o algoritmo notaria a diferença e indicaria o erro. Cada prescrição tem um intervalo de 8 a 10 comutadores e solicitações inválidas são detectadas assim que as alterações são percebidas no *middleware*.

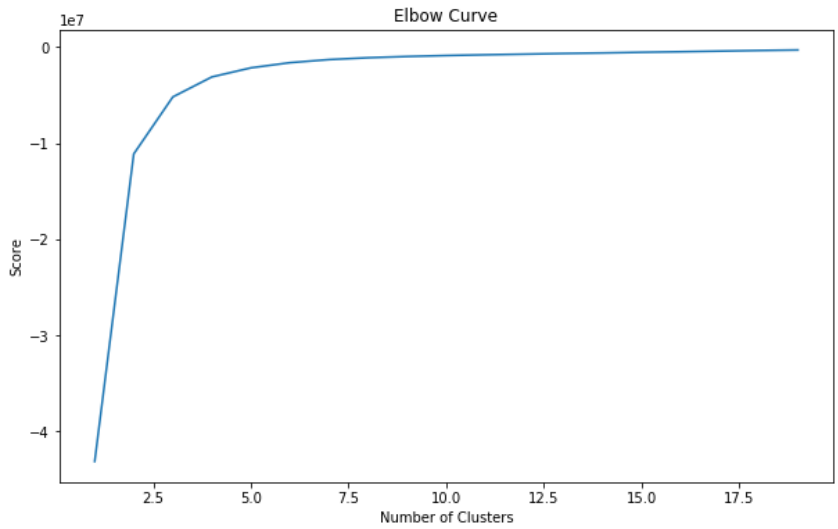
Antes de passar pelo algoritmo, categorizou-se os dados em formato numérico. Após a categorização, verificou-se a variação dos dados, como visto em 21, para confirmar quais informações foram alteradas de alguma forma, removendo as informações que permaneceram intactas ao longo do tempo e reduzindo as dimensões passadas para o algoritmo. Com os dados transformados, realizou-se os testes do algoritmo K-means para encontrar o número de *clusters* necessários. Para isso, usou-se o método (HOTCHKISS; WEILAND, 1987; BHOLOWALIA; KUMAR, 2014) e obtemos a curva 22, que aponta para o número de *clusters* necessários para o teste. Foi utilizado o valor de 4 grupos, sendo este valor escolhido após os testes realizados e verificando a eficiência com esse número de *clusters*.

Figura 21 – Variação de componentes individuais e cumulativos.



A partir da primeira perspectiva, usou-se o número de *cluster* igual a dois, porém,

Figura 22 – método de elbow para escolher o número de *clusters*.



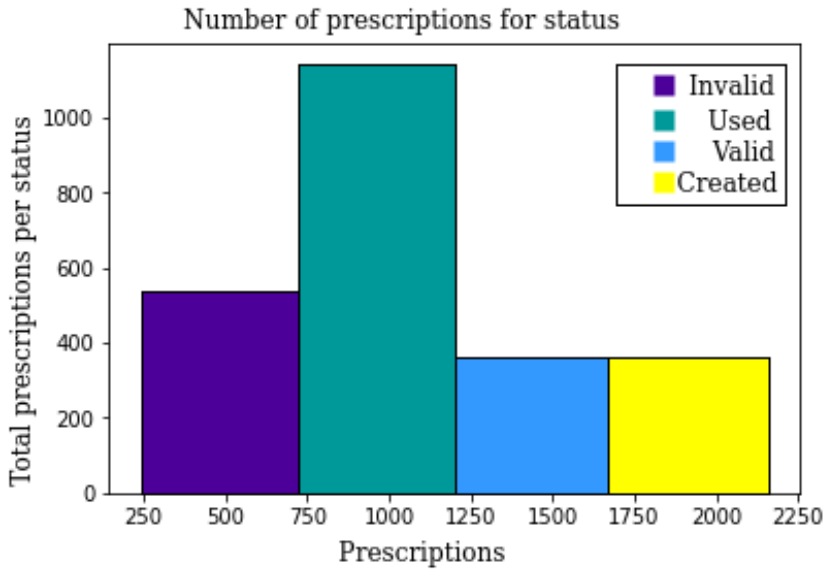
muitos falsos positivos apareceram na detecção, o que demonstrava que a separação não estava permitindo detectar de forma correta as transações. Portanto, após uma análise empírica e avaliando o comportamento dos *cluster* e a precisão obtida chegou-se a conclusão que, com o número de *clusters* igual a 4, conseguimos cobrir uma boa parte das transações e identificá-las de maneira correta.

Embora houvesse o registro quais transações eram falsas e quais não eram, não foi passado ao algoritmo para permitir que se determine sem nenhuma influência. Como pode ser visto na figura 23, tivemos um total de 500 transações inválidas 380 transações válidas, 380 transações criadas e 1240 operações usadas. Esses valores se referem ao número de operações necessárias para interagir com as 80 prescrições criadas. Em cada interação, o estado das transações foi coletado e analisado. Na Figura 24, mostrou-se as séries temporais formadas pelas operações 2500, cada linha representando os dados de uma transação. Como pretendemos encontrar grupos que representam transações válidas e inválidas e os grupos intermediários usados e criados, avaliou-se as negociações como um todo e, portanto, quando um acordo tem a possibilidade de anomalia, podemos analisá-lo de uma maneira mais específica, realizando um corte de transações por ID de prescrição e vinculando todas as transações que foram executadas em uma transação específica.

Para realizar os testes, separou-se os dados em treinamento e análise, que é de 80 % para treinamento e 20 % para teste, respeitando a distribuição dos dados com anomalia e sem anomalia. Como resultado da análise, pode-se observar que as transações foram separadas em três clusters, dois clusters com transações válidas e um cluster com transações inválidas.

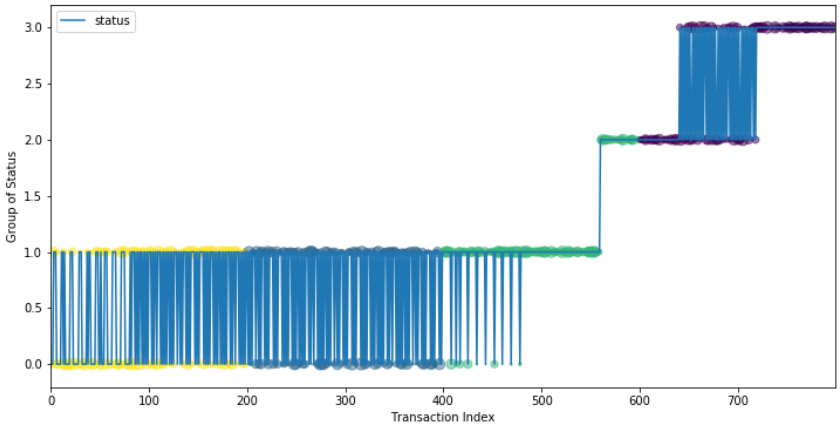
A figura 24 mostra os pontos nos quais as transações, neste caso, as prescrições, foram identificadas nos quatro grupos (criação, usado, válido e inválido), transações que foram alteradas, embora inicialmente estivessem em uma. O cluster correto, a partir de um processo no meio do caminho em que foram alterados, levou a uma mudança de comportamento na linha do tempo,

Figura 23 – Número de transações por estado.



passando para um cluster que os denotava como anômalo. Observou-se que o tipo de função usada entre a criação e o compartilhamento de uma prescrição pode indicar que as alterações foram intencionalmente feitas para contornar os termos do contrato.

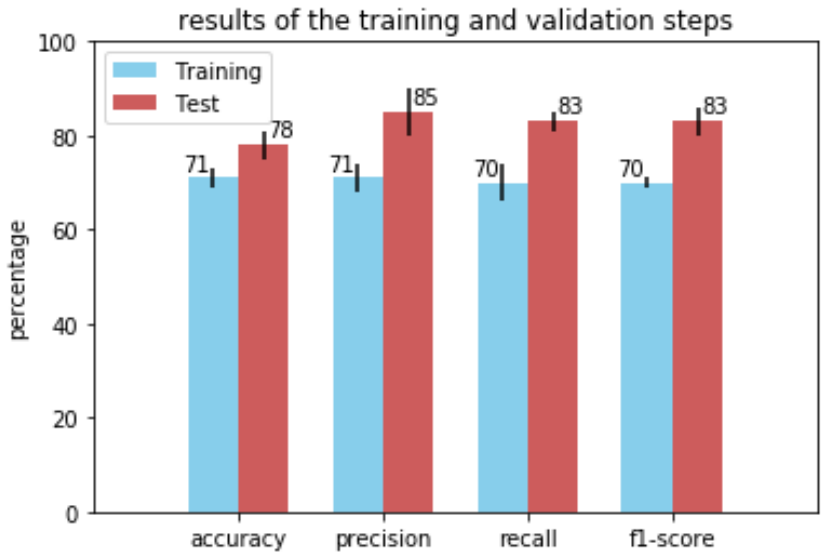
Figura 24 – Classificação com 4 grupos no teste.



Na fase de treinamento, como pode ser visto na figura 25, obtivemos uma precisão de 70 % e permanecemos na questão de precisão, recordação e pontuação f1. Na etapa de teste, embora o percentual tenha sido um pouco maior, com precisão de 78 % e precisão de 85 %, permaneceu estatisticamente conciso e próximo à etapa de treinamento. Em comparação com outras técnicas que poderiam ser usadas, o clustering foi aquele com menor número de falsos positivos, mas tem-se a possibilidade de usar algoritmos para detectar discrepâncias e auxiliar na detecção de anomalias.

Com base nos resultados de desempenho apresentados, a utilização do modelo para detecção de anomalias não irá trazer problemas significativamente ao desempenho da rede. Conforme é discutido e analisado em (NUNES, 2016) a complexidade do algoritmo k-means

Figura 25 – Resultado do algoritmo kmeans.



está na ordem de $O(nkd)$ para o treinamento, sendo n distâncias dos pontos aos k centroides, k o número de *clusters* e d a dimensão do ponto. E $O(tnkd)$ para a atualização dos *clusters* com base na média dos pontos. A complexidade do algoritmo para o consenso baseado no Problema de Tolerância a Falhas Bizantinas, na sigla em inglês Practical Byzantine Fault Tolerance (PBFT) é relativamente maior quando comparado a do algoritmo para detecção de anomalias. Dada a análise mostrada em (GABRIEL, 2019), o PBFT está na ordem $O(n^2)$. Como o treinamento dos *clusters* não precisa ser realizado a cada transação, temos apenas que garantir a sua periodicidade e avaliar como os *cluster* estão se comportando para que a detecção não diminua sua precisão.

CAPÍTULO

8

CONCLUSÃO

8.1 Conclusão

Este trabalho teve como objetivo criar um mecanismo para detecção de anomalias em uma rede descentralizada, utilizando a tecnologia Blockchain com contratos inteligentes. Isso foi possível com a criação do *middleware* MADCS. O *middleware* MADCS é uma arquitetura de seis camadas que introduz o detector de anomalias dentro da arquitetura do Blockchain. Essa arquitetura permite utilizar os dados históricos das transações para criar dados, que são utilizados para gerar os grupos com o algoritmo k-means. Dessa forma, um modelo é gerado para detectar as transações que inválidas, dado a avaliação das transações e a concepção de um administrador para avaliar corretamente e confirmar o grupo inválido. Para avaliar a camada de detecção, foram utilizados dados empíricos gerados a partir de um conjunto de dados de prescrições farmacêuticas. Para utilizar esses dados, foi desenvolvimento uma estrutura de contrato inteligente para criação e dispensação farmacêutica, no qual as transações foram geradas a partir da criação e execução dos contratos. A partir das transações foi possível treinar o modelo e detectar os dados de teste e conforme os resultados obtidos, cerca de 75% de acurácia e 85% de precisão. Após a avaliação do algoritmo, foi verificado o desempenho da rede com a adição do MADCS. Para avaliar o desempenho foi verificado a latência da rede, o *overhead* de CPU para cada par da rede e no final foi realizado uma média dos tempos de execução dos pares da rede com o MADCS e sem o MADCS e foi observado que houve um aumento do tempo devido a avaliação do *middleware* para que não interfere de forma significativa na condução das transações na rede. Para que o *middleware* possa ser executado o modelo e os dados bem como a estrutura da rede pode ser encontrado no repositório MADCS (SILVA, 2020). O *middleware* cumpriu seu objetivo na rede e como trabalhos futuros, pretende-se aprimorar a atualização dos *clusters* e testar outros algoritmos para realizar a detecção de anomalias, além de introduzir outras estruturas do cenário médico para análise e desenvolvimento da rede. Também pretende-se avaliar outros cenários como créditos bancários e ambientes IoT.

8.2 Trabalhos Futuros

Garantir que todos os pares da rede estão transmitidos anomalias corretamente é um ponto importante que precisa ser melhorado. Cada avaliador deve verificar sua informação local e repassar aos pares para que avaliem e atualizem seus grupos. Como trabalho futuro, o algoritmo para avaliação de dados de outros pares deve ser refeito, de modo que possa identificar grupos que tentam alterar os grupos com informações incorretas. A configuração e hierarquia de acesso das organizações é um ponto para refletir e organizar. Dado que uma organização pode exercer um poder maior sobre as outras, seria necessário definir diferentes papéis na rede, conforme a hierarquia de cada grupo. Este projeto, apesar de ter um estudo de caso focado em prescrições médicas, pode se estender por todo o sistema médico, incluindo prontuários médicos e outras funções do sistema e como trabalho futuro, pretende-se explorar como as organizações de saúde gerenciam seus recursos e as demais informações sensíveis que são compartilhadas entre as organizações.

O protocolo de Fofoca está no estágio inicial e pode ser melhorado. Este protocolo se comunica com todos os peers quando uma anomalia é encontrada. Dessa forma, todos os peers da rede estarão cientes de qualquer problema com uma transação e poderá treinar seu modelo local para prever determinado incidente. A comunicação feita como *broadcast* pode ser melhorada para reduzir os envios para a rede.

8.3 Limitações

Uma das limitações da rede está ligado a camada de Análise e detecção de anomalias. Como foi possível observar, a rede opera e treina os modelo para detecção através de dados históricos da rede. Em uma rede totalmente inicial e sem dados para popular, seria complexo criar um modelo artificial para detectar a anomalia nos dados. Porém, no caso de uma portabilidade, poderia repassar os dados na rede, ou fazer um treinamento com dados que a organização já utiliza e ter um modelo para avaliar os dados até que a rede obtenha transações necessárias para ter um modelo com base nos próprios dados. Outro pronto são os contratos inteligentes ligados aos modelos de detecção. Para garantir a imutabilidade inserimos cada modelo ligado ao contrato. Conforme o contrato seja feito um novo *deploy*, o modelo poderá ser atualizado com ele. Em rede onde os custos para cada criação de contrato seja elevado, seria necessário ter a camada de análise e detecção fora do *blockchain*, o que leva a ter mais mecanismo para garantir a confiabilidade do modelo.

REFERÊNCIAS

ALECRIM, E. **O que é GDPR e que diferença isso faz para quem é brasileiro**. <https://tecnoblog.net/245101/gdpr-privacidade-protecao-dados/>, 2020. Citado na página 100.

Alkurdi, F.; Elgendi, I.; Munasinghe, K. S.; Sharma, D.; Jamalipour, A. Blockchain in IoT Security: A Survey. In: **2018 28th International Telecommunication Networks and Applications Conference (ITNAC)**. [S.l.]: IEEE, Sydney, NSW, Austrália, 2018. p. 1–4. ISSN 2474-154X. Citado nas páginas 44 e 45.

ANALYTICS, R. **Prescription-based prediction, Predicting doctor attributes from prescription behavior**. <https://www.kaggle.com/roamresearch/prescriptionbasedprediction#roam_prescription_based_prediction.jsonl>, 2018. Citado nas páginas 27 e 87.

Anthi, E.; Williams, L.; Słowińska, M.; Theodorakopoulos, G.; Burnap, P. A supervised intrusion detection system for smart home iot devices. **IEEE Internet of Things Journal**, v. 6, n. 5, p. 9042–9053, Oct 2019. ISSN 2372-2541. Citado nas páginas 56, 57 e 61.

ANTONOPOULOS, A. M. **Mastering Bitcoin: unlocking digital cryptocurrencies**. [S.l.]: "O'Reilly Media, Inc.", 2014. Citado na página 26.

AZURE, M. **O que é middleware?** <https://azure.microsoft.com/pt-br/overview/what-is-middleware/>, 2020. Citado na página 100.

BHOLOWALIA, P.; KUMAR, A. Ebk-means: A clustering technique based on elbow method and k-means in wsn. **International Journal of Computer Applications**, Citeseer, v. 105, n. 9, p. 1 – 8, 2014. Citado na página 88.

Bhowmik, D.; Feng, T. The multimedia blockchain: A distributed and tamper-proof media transaction framework. In: **2017 22nd International Conference on Digital Signal Processing (DSP)**. <https://signalprocessingsociety.org/blog/dsp-2017-2017-22nd-international-conference-digital-signal-processing>: IEEE Signal Processing, London, UK, 2017. p. 1–5. ISSN 2165-3577. Citado na página 26.

CACHIN, C. Architecture of the hyperledger blockchain fabric. In: **Workshop on distributed cryptocurrencies and consensus ledgers**. [S.l.]: Proceedings of the 2017 on Cloud Computing Security Workshop, Dallas, Texas, USA, 2016. v. 310, p. 1–4. Citado na página 49.

CHENG, L.; LIU, J.; SU, C.; LIANG, K.; XU, G.; WANG, W. Polynomial-based modifiable blockchain structure for removing fraud transactions. **Future Generation Computer Systems**, v. 99, p. 154–163, 2019. Citado na página 26.

COMPOSER, H. **Welcome to Hyperledger Composer**. <https://hyperledger.github.io/composer/v0.19/introduction/introduction>, 2019. Citado nas páginas 27, 49, 50, 51, 70, 77 e 81.

COULOURIS, G.; DOLLIMORE, J.; KINDBERG, T.; BLAIR, G. **Sistemas Distribuídos-: Conceitos e Projeto**. [S.l.]: Bookman Editora, 2013. Citado nas páginas 33, 34, 35, 36, 37, 38 e 39.

FEIGE, U.; FIAT, A.; SHAMIR, A. Zero-knowledge proofs of identity. **Journal of Cryptology**, v. 1, n. 2, p. 77–94, Jun 1988. ISSN 1432-1378. Disponível em: <https://doi.org/10.1007/BF02351717>. Citado na página 52.

FENG, Q.; HE, D.; ZEADALLY, S.; KHAN, M. K.; KUMAR, N. A survey on privacy protection in blockchain system. **Journal of Network and Computer Applications**, v. 126, p. 45 – 58, 2019. ISSN 1084-8045. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1084804518303485>. Citado na página 27.

Fotiou, N.; Polyzos, G. C. Decentralized name-based security for content distribution using blockchains. In: **2016 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)**. <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7562112>: San Francisco, CA, USA, 2016. p. 415–420. Citado na página 26.

GABRIEL, J. **Um olhar técnico sobre o algoritmo do Consenso da Libra**. <https://gazetalibertaria.news/joaogabriel/olhar-tecnico-consenso-libra/>, 2019. Citado na página 91.

GOLOMB, T.; MIRSKY, Y.; ELOVICI, Y. Ciota: Collaborative iot anomaly detection via blockchain. **arXiv preprint arXiv:1803.03807**, 2018. Citado nas páginas 57, 59 e 61.

GOMES, P. C. T. **O que é e como funciona um cluster?** <https://www.opservices.com.br/o-que-e-um-cluster/>, 2015. Citado na página 99.

HODO, E.; BELLEKENS, X.; HAMILTON, A.; DUBOUILH, P.-L.; IORKYASE, E.; TACHTATZIS, C.; ATKINSON, R. Threat analysis of iot networks using artificial neural network intrusion detection system. In: IEEE. **2016 International Symposium on Networks, Computers and Communications (ISNCC)**. [S.l.], 2016. p. 1–6. Citado nas páginas 58, 60 e 61.

HÖLBL, M.; KOMPARA, M.; KAMIŠALIĆ, A.; ZLATOLAS, L. N. A systematic review of the use of blockchain in healthcare. **Symmetry**, Multidisciplinary Digital Publishing Institute, v. 10, n. 10, p. 470, 2018. Citado na página 26.

HOTCHKISS, R. N.; WEILAND, A. J. Valgus stability of the elbow. **Journal of Orthopaedic Research**, Wiley Only Library, <https://onlinelibrary.wiley.com/doi/abs/10.1002/jor.1100050309>, v. 5, n. 3, p. 372–377, 1987. Citado na página 88.

ICHIKAWA, D.; KASHIYAMA, M.; UENO, T. Tamper-resistant mobile health using blockchain technology. **JMIR mHealth and uHealth**, JMIR Publications Inc., Toronto, Canada, v. 5, n. 7, p. e111, 2017. Citado na página 26.

JAIN, A. K. Data clustering: 50 years beyond k-means. **Pattern recognition letters**, Elsevier, v. 31, n. 8, p. 651–666, 2010. Citado na página 51.

KRISHNA, K.; MURTY, M. N. Genetic k-means algorithm. **IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)**, IEEE, v. 29, n. 3, p. 433–439, 1999. Citado nas páginas 27 e 51.

Leka, E.; Selimi, B.; Lamani, L. Systematic literature review of blockchain applications: Smart contracts. In: **2019 International Conference on Information Technologies (InfoTech)**. [S.l.: s.n.], 2019. p. 1–3. Citado na página 47.

MANOHAR, A.; BRIGGS, J. Identity management in the age of blockchain 3.0. Association for Computing Machinery, 2018. Citado na página 99.

Mettler, M. Blockchain technology in healthcare: The revolution starts here. In: **2016 IEEE 18th International Conference on e-Health Networking, Applications and Services (Healthcom)**. https://ieeexplore.ieee.org/abstract/document/7749510: Munich, Germany, 2016. p. 1–3. Citado na página 26.

NAKAMOTO, S. Bitcoin: A peer-to-peer electronic cash system. **Amazon Document**, Working Paper, <https://s3.amazonaws.com/academia.edu.documents/54517945/Bitcoin_paper_Original_2.pdf>, v. 1, p. 1–9, 2008. Citado nas páginas 26 e 41.

NASIR, M.; HUYNH, T.; NGUYEN, S.; DUONG, D. Forecasting cryptocurrency returns and volume using search engines. **Financial Innovation**, v. 5, n. 1, 2019. Citado na página 26.

Nath, I. Data Exchange Platform to Fight Insurance Fraud on Blockchain. In: **2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)**. [S.l.]: Barcelona, Spain, 2016. p. 821–825. ISSN 2375-9259. Citado na página 41.

Niya, S. R.; Schiller, E.; Cepilov, I.; Maddaloni, F.; Aydinli, K.; Surbeck, T.; Bocek, T.; Stiller, B. Adaptation of proof-of-stake-based blockchains for iot data streams. In: **2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)**. [S.l.: s.n.], 2019. p. 15–16. Citado na página 47.

NUNES, D. H. F. **Um breve estudo sobre o algoritmo K-means**. Dissertação (Mestrado) — Departamento de Matemática, Faculdade de ciência e Tecnologia, Universidade de Coimbra, 2016. Citado na página 90.

PINNA, A.; RUTTENBERG, W. Distributed ledger technologies in securities post-trading revolution or evolution? **ECB Occasional Paper**, n. 172, 2016. Citado na página 40.

PORTUGUÊS, D. O. de. **Anomalia**. <https://www.dicio.com.br/anomalia/>, 2020. Citado na página 99.

PUTRA, G. D.; DEDEOGLU, V.; KANHERE, S. S.; JURDAK, R. Towards scalable and trustworthy decentralized collaborative intrusion detection system for iot. **arXiv preprint arXiv:2002.07512**, 2020. Citado nas páginas 55, 56 e 61.

RANDALL, D.; GOEL, P.; ABUJAMRA, R. Blockchain applications and use cases in health information technology. **J Health Med Informat**, v. 8, n. 276, p. 2, 2017. Citado na página 26.

REDAÇÃO. **O que é DoS e DDoS?** <https://canaltech.com.br/produtos/o-que-e-dos-e-ddos/>, 2020. Citado na página 99.

SHARMA, P. K.; MOON, S. Y.; PARK, J. H. Block-vn: A distributed blockchain based vehicular network architecture in smart city. **JIPS**, v. 13, n. 1, p. 184–195, 2017. Citado na página 26.

SILVA, A. **MADCS, implementação do blockchain para criação e utilização do middleware**. <https://github.com/alef123vinicius/MADCS>, 2020. Citado nas páginas 75 e 93.

SONDA. **Afinal, o que é o middleware?** <https://blog.sonda.com/o-que-e-middleware/>, 2020. Citado na página 100.

Sukhwani, H.; Martínez, J. M.; Chang, X.; Trivedi, K. S.; Rindos, A. Performance modeling of pbft consensus process for permissioned blockchain network (hyperledger fabric). In: **2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS)**. [S.l.: s.n.], 2017. p. 253–255. Citado na página 46.

SWAN, M. Connected car: quantified self becomes quantified car. **Journal of Sensor and Actuator Networks**, Multidisciplinary Digital Publishing Institute, v. 4, n. 1, p. 2–29, 2015. Citado na página 26.

SZABO, N. **Smart Contracts**. <shorturl.at/dvO13>, 1994. Citado na página 70.

TANKARD, C. What the gdpr means for businesses. **Network Security**, v. 2016, n. 6, p. 5 – 8, 2016. ISSN 1353-4858. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1353485816300563>. Citado na página 25.

Tug, S.; Meng, W.; Wang, Y. Cbsigids: Towards collaborative blockchained signature-based intrusion detection. In: **2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)**. [S.l.: s.n.], 2018. p. 1228–1235. ISSN null. Citado nas páginas 57, 58 e 61.

VOIGT, P.; BUSSCHE, A. Von dem. The eu general data protection regulation (gdpr). **A Practical Guide, 1st Ed., Cham: Springer International Publishing**, Springer, 2017. Citado nas páginas 25 e 100.

WACHTER, S.; MITTELSTADT, B.; RUSSELL, C. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. **Harv. JL & Tech.**, HeinOnline, v. 31, p. 841, 2017. Citado na página 25.

Wang, K.; Zhang, Z.; Kim, H. S. Reviewchain: Smart contract based review system with multi-blockchain gateway. In: **2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)**. [S.l.: s.n.], 2018. p. 1521–1526. Citado na página 47.

WOOD, G. *et al.* Ethereum: A secure decentralised generalised transaction ledger. **Ethereum project yellow paper**, v. 151, p. 1–32, 2014. Citado na página 48.

Wu, S.; Chen, Y.; Wang, Q.; Li, M.; Wang, C.; Luo, X. Cream: A Smart Contract Enabled Collusion-Resistant e-Auction. **IEEE Transactions on Information Forensics and Security**, v. 14, n. 7, p. 1687–1701, July 2019. ISSN 1556-6013. Citado na página 26.

Yang, X.; Chen, Y.; Chen, X. Effective scheme against 51% attack on proof-of-work blockchain with history weighted information. In: **2019 IEEE International Conference on Blockchain (Blockchain)**. [S.l.: s.n.], 2019. p. 261–265. Citado na página 46.

Yuan, Y.; Wang, F. Towards blockchain-based intelligent transportation systems. In: **2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)**. <https://ieeexplore.ieee.org/abstract/document/7795984>: Rio de Janeiro, Brazil, 2016. p. 2663–2668. ISSN 2153-0017. Citado na página 26.

Zheng, Z.; Xie, S.; Dai, H.; Chen, X.; Wang, H. An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends. In: **2017 IEEE International Congress on Big Data (BigData Congress)**. [S.l.]: IEEE, Honolulu, HI, USA, 2017. p. 557–564. Citado nas páginas 44 e 45.

GLOSSÁRIO

Anomalia: Termo que designa uma característica atípica ou uma irregularidade de um processo, um produto, uma matéria, um fenômeno natural ou qualquer fato fora do comum, dentro do escopo observado (PORTUGUÊS, 2020). No contexto desse trabalho, anomalia são as alterações que podem ocorrer nos dados dos usuários antes de serem executados na Rede Blockchain.

Blockchain 3.0: Terceira geração da rede que implementa a estrutura da Rede em áreas não financeiras como este trabalho por exemplo. Outros exemplos seriam: sistemas cívicos, sistemas de saúde em geral, gerenciamento de identidades, arte, entre outros (MANOHAR; BRIGGS, 2018).

Cluster: Cluster pode ser traduzido como grupos ou aglomerações que pode ser aplicado em vários contextos. No caso da computação, tem-se o termo para definir arquiteturas baseadas em grupos de processadores capazes trabalhar em conjunto e em paralelo ou pode denominar um grupo de computadores combinados para trabalhar juntos e compartilhar funções (GOMES, 2015). Para analise de dados seriam grupos ou aglomerações desses dados no espaço e que denotam uma ou mais características em comum.

DApp: Termo utilizado para a camada de aplicações em redes Blockchain. São porta de entrada para a comunicação dos usuários com a rede, que podem interagir utilizando chaves criptográficas ou a própria interface estabelecida, dado que o sistema está além da utilização como criptomoeda, não sendo somente uma Wallet(carteira digital).

DDoS: A sigla Distributed Denial of Service (DDoS) significa *Distributed Denial of Service* também conhecido como ataque de negação de serviço distribuído é um tipo de ataque DoS, mas o computador mestre pode ter vários computadores escravos e realizar um ataque em paralelo de vários lugares diferentes. Dessa forma, o servidor pode atingir o limite de acesso que pode atender e o serviço fica inacessível para outros usuários (REDAÇÃO, 2020).

DoS: A sigla Denial of Service (DoS) significa (Denial of Service) É um tipo de ataque que pode ser realizado por um único computador. Sua característica principal é dada pela tentativa de fazer com que aconteça uma sobrecarga em um servidor ou computador comum para que recursos do sistema fiquem indisponíveis para seus utilizadores (REDAÇÃO, 2020).

GDPR: General Data Protection Regulation é um projeto de lei para proteção de dados e identidade de usuários. Apesar da lei ser de origem na União Europeia, muitas empresas estão revendo suas políticas de proteção aos dados sensíveis dos usuários, como por exemplos RG, nome, exames médicos e tudo que pode identificar ou relacionar a uma pessoa (VOIGT; BUSSCHE, 2017; ALECRIM, 2020).

IDS: Termo comumente utilizado para Sistemas de Detecção de Intrusão necessários para o gerenciamento de uma infraestrutura de rede e sua segurança.

Ledger: Termo que se refere ao livro razão onde são registrados transações e todo o processo financeiro de entrada e saída de dinheiro. Atualmente é um termo também utilizado para registros eletrônicos em sistemas de criptomoedas.

Middleware: Segundo a Microsoft Azure (AZURE, 2020) o middleware é o software que se encontra entre o sistema operacional e os aplicativos nele executados. Funcionando de forma essencial como uma camada oculta de tradução permitindo a comunicação, gerenciamento e transmissão de dados para aplicativos distribuídos. Alguns exemplos comuns são middlewares para banco de dados, servidores de aplicativos, orientado a mensagens e monitores de processamento de transações. Em (SONDA, 2020) é evidenciado que com o middleware, as informações provenientes de diversas fontes passam a ser filtradas e refinadas, o que reduz (e muito) o volume do processamento de dados, aumentando a qualidade das informações fornecidas.

Raspberry PI: É um micro PC programável, que possui recursos de memória e cpu mais limitados que um CPU, porém, possui grande capacidade quando comparado aos microcontroladores. Normalmente é utilizados em projetos de programação, IoT, dentre outros projetos eletrônicos.

token: O Token é um dispositivo eletrônico ou virtual que pode gerar senhas periodicamente ou baseado em eventos. Esse dispositivo é muito importante para a segurança dos usuários, para proteger seus dados pessoais. Possui algumas versões físicas onde fica armazenado em um dispositivo USB e pode ser utilizado conectando-o ao computador.


ICMC
USP
SÃO CARLOS