

**UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE INFORMÁTICA
TECNOLOGIA EM ANÁLISE DE DESENVOLVIMENTO DE SISTEMAS**

**FELIPE AUGUSTO BARCELOS
JOÃO VICTOR DE LIMA MARTINS**

**APLICABILIDADE DA *BLOCKCHAIN* NO SISTEMA DE ELEIÇÃO
DEPARTAMENTAL DA UTFPR CAMPUS PONTA GROSSA**

TRABALHO DE CONCLUSÃO DE CURSO

**PONTA GROSSA
2020**

**FELIPE AUGUSTO BARCELOS
JOÃO VICTOR DE LIMA MARTINS**

**APLICABILIDADE DA *BLOCKCHAIN* NO SISTEMA DE ELEIÇÃO
DEPARTAMENTAL DA UTFPR CAMPUS PONTA GROSSA**

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas, do Departamento Acadêmico de Informática, da Universidade Tecnológica Federal do Paraná.

Orientadora: Prof. Dra. Simone de Almeida

PONTA GROSSA

2020



Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Ponta Grossa



Diretoria de Graduação e Educação Profissional
Departamento Acadêmico de Informática
Tecnologia em Análise e Desenvolvimento de Sistema

TERMO DE APROVAÇÃO

APLICABILIDADE DA *BLOCKCHAIN* NO SISTEMA DE ELEIÇÃO DEPARTAMENTAL DA UTFPR CAMPUS PONTA GROSSA

por

FELIPE AUGUSTO BARCELOS
JOÃO VICTOR DE LIMA MARTINS

Este Trabalho de Conclusão de Curso (TCC) foi apresentado em 21 de setembro de 2020 como requisito parcial para a obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas. Os candidatos foram arguidos pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

Prof.^a Dra Simone de Almeida
Orientadora

Prof. Dr Augusto Foronda
Membro titular

Prof. MSc Rogério Ranthum
Membro titular

Prof. MSc Geraldo Ranthum
Responsável pelo Trabalho de Conclusão de Curso

Prof. Dr André Pinz Borges
Coordenador do curso

Dedico este trabalho à minha família,
especialmente a minha filha Lara,
pelos momentos de ausência e a
cada pequeno passo seu que não
pude presenciar.

AGRADECIMENTOS

Agradecemos primeiramente a nossa orientadora Simone de Almeida, que sempre demonstrou disponibilidade e atenção ao desenvolvimento do trabalho e durante o curso de nossa vida acadêmica, pudemos perceber como uma professora comprometida com o aprendizado e bem-estar dos alunos.

Felipe: “Gostaria de agradecer também aos meus pais, em especial ao meu pai que partiu deste mundo no decorrer de minha graduação, mas sempre esteve preocupado comigo e presente nos momentos que eu mais precisei, mesmo com toda minha insubordinação às palavras dele”.

João: “Gostaria de agradecer minha mãe, que desde sempre batalhou para que minha educação nunca fosse afetada apesar das dificuldades que passamos, e ao meu pai que pelo tempo que esteve ao meu lado sempre me guiou para o caminho do bem e da honestidade”.

Por fim, queremos agradecer a todos os outros profissionais, professores e colegas verdadeiramente envolvidos e comprometidos com a qualidade do curso e que tornaram possível chegarmos onde estamos agora.

Bitcoin é uma conquista criptográfica notável. A capacidade de criar algo que não é duplicável no mundo digital tem um valor enorme. Muitas pessoas construirão negócios em cima disso (ERIC EMERSON SCHMIDT, 2014)

RESUMO

BARCELOS, Felipe Augusto; MARTINS, João Victor De Lima. **Aplicabilidade Da Blockchain no Sistema de Eleição Departamental da UTFPR Campus Ponta Grossa**. 2020. 59 f. Trabalho de Conclusão de Curso de Tecnologia em Análise e Desenvolvimento de Sistemas - Universidade Tecnológica Federal do Paraná. Ponta Grossa, 2020.

O presente trabalho se destina a utilizar um sistema de votação baseado em *blockchain* para verificar a viabilidade de sua utilização no processo eleitoral de escolha de chefe de departamento da UTFPR campus Ponta Grossa. Espera-se que a possibilidade de uma eleição segura online possa trazer mais eleitores onde a votação não é obrigatória. Foram utilizadas interface *web* e uma *blockchain* privada. Verificou-se a possibilidades de manter o voto secreto em uma votação online, sem ser possível comprovar tanto para o eleitor quanto a mesa organizadora. Analisou-se os quesitos segurança e desempenho do sistema proposto. Concluiu-se que apesar de viável a utilização de um sistema eleitoral online utilizando a *blockchain*, é impossível manter o sigilo do voto se a votação não for feita em um recinto controlado.

Palavras-chave: Sistema eleitoral descentralizado. *Blockchain*.

ABSTRACT

BARCELOS, Felipe Augusto; MARTINS, João Victor De Lima. **Blockchain applicability in the UTFPR Campus Ponta Grossa Departmental Election System**. 2020. 59 p. Work of Conclusion Course Graduation in Análise e Desenvolvimento de Sistemas - Federal Technology University - Paraná. Ponta Grossa, 2020.

This work aims to use a blockchain-based voting system to verify the feasibility of its use in the electoral process of choosing the department head of the UTFPR campus Ponta Grossa. It is hoped that the possibility of a secure online election could bring in more voters where voting is not mandatory. A web interface and a private blockchain were used. The possibility of keeping the vote secret in an online poll were verified, without being possible to prove to both the voter and the organizing table. The security and performance requirements of the proposed system were analyzed. It was concluded that although the use of an online electoral system using the blockchain is feasible, it is impossible to maintain the secrecy of the vote if voting is not done in a controlled venue.

Keywords: Decentralized electoral system. Blockchain.

LISTA DE ILUSTRAÇÕES

Figura 1 - Função <i>hash</i> SHA-1.....	25
Figura 2 - Criptografia Simétrica.....	29
Figura 3 - Criptografia Assimétrica.....	30
Figura 4 - Assinatura Digital.....	31
Figura 5 - Composição do bloco.....	33
Figura 6 - Tela de Login.....	36
Figura 7 - Tela Inicial.....	36
Figura 8 - Tela de Candidatos.....	37
Figura 9 - Tela de Usuários.....	37
Figura 10 - Tela para Registro de Nós.....	38
Figura 11 - Tela de Resultados.....	38
Figura 12 - Criação do Bloco Gênese.....	39
Figura 13 - Inicialização do Código.....	39
Figura 14 - Novas Transações e Novos Blocos.....	40
Figura 15 - PoW.....	41
Figura 16 - Registro de Nós.....	42
Figura 17 - Consenso Entre Nós.....	43
Figura 18 - Rotas de Requisições HTTP	45

LISTA DE QUADROS

Quadro 1 - Tabela de Candidatos.....	34
Quadro 2 - Tabela de Usuários.....	35

LISTA DE ABREVIATURAS, SIGLAS E ACRÔNIMOS

LISTA DE SIGLAS

API	<i>Application Programming Interface</i>
HTTP	<i>Hypertext Transfer Protocol</i>
UTFPR	Universidade Tecnológica Federal do Paraná
SQL	<i>Structured Query Language</i>

LISTA DE ACRÔNIMOS

PoS	<i>Proof-of-Stake</i>
PoW	<i>Proof-of-Work</i>

SUMÁRIO

1 INTRODUÇÃO	14
1.1 OBJETIVOS.....	15
1.1.1 Objetivo Geral.....	15
1.1.2 Objetivos Específicos.....	16
1.2 JUSTIFICATIVA.....	16
1.3 MÉTODO UTILIZADO.....	17
1.4 ORGANIZAÇÃO DO TRABALHO.....	17
2 LEGISLAÇÃO QUE REGULAMENTA O SERVIÇO PÚBLICO	18
2.1 SUSTENTAÇÃO CONSTITUCIONAL.....	18
2.2 LEI DE ACESSO À INFORMAÇÃO.....	19
2.3 LEGISLAÇÃO VIGENTE NA UTFPR.....	21
2.4 CONSIDERAÇÕES DO CAPÍTULO.....	22
3.TECNOLOGIAS APLICADAS À BLOCKCHAIN	24
3.1 FUNÇÕES <i>HASH</i>	24
3.2 ALGORITMOS DE CONSENSO.....	25
3.2.1 <i>Proof-of-Work</i> (PoW).....	26
3.2.1 <i>Proof-of-Stake</i> (PoS).....	27
3.3 CRIPTOGRAFIA.....	28
3.3.1 Criptografia Simétrica.....	29
3.3.2 Criptografia Assimétrica.....	30
3.3.3 Assinatura Digital.....	31
3.4 ARQUITETURA DA <i>BLOCKCHAIN</i>	31
3.5 CONSIDERAÇÕES DO CAPÍTULO.....	33
4 DESENVOLVIMENTO	34
4.1 INTERFACE WEB.....	34
4.1.1 Tela de Login.....	35
4.1.2 Tela Inicial.....	36
4.1.3 Candidatos.....	36
4.1.4 Usuários.....	37
4.1.5 Registro de nós.....	37
4.1.6 Resultado.....	38
4.2 <i>BLOCKCHAIN</i>	38
4.2.1 Criação do bloco gênese.....	39
4.2.2 Criação de novas transações e novos blocos.....	40
4.2.3 Mineração, PoW e verificação de <i>hash</i>	41
4.2.4 Registro de nós.....	42
4.2.5 Consenso entre os nós.....	43
4.2.6 Rotas de requisições HTTP.....	45
4.3 CONSIDERAÇÕES DO CAPÍTULO.....	48

5 RESULTADOS	50
5.1 ENDEREÇOS PRIVADOS.....	50
5.2 ANONIMATO.....	50
5.3 SEGURANÇA E DESEMPENHO.....	51
5.4 CONSIDERAÇÕES DO CAPÍTULO.....	52
6 CONSIDERAÇÕES FINAIS	53
6.1 CONCLUSÃO.....	53
6.2 TRABALHOS FUTUROS.....	54
ANEXO A - O Problema Dos Generais Bizantinos	57

1 INTRODUÇÃO

Em 2008, o pseudônimo Satoshi Nakamoto publicou “*Bitcoin: A Peer-to-peer Electronic Cash System*” que trata de um sistema eletrônico ponto a ponto para a transferência de dinheiro. Este sistema utiliza criptografia assimétrica, normalmente utilizada em ambientes não confiáveis e que serve para validar a autenticação de transações do sistema.

Bitcoin é uma rede *peer-to-peer* e uma moeda digital que opera, essencialmente, como dinheiro *online*. Seu funcionamento consiste na verificação das transações, e o gasto duplo é prevenido, por meio de um uso inteligente de criptografia de chave pública. Esse mecanismo exige que a cada usuário sejam atribuídas duas chaves, uma privada, que é mantida em segredo, como uma senha, e outra pública, que pode ser compartilhada.

Quando um usuário “A” decide transferir Bitcoin para o usuário “B”, cria-se uma mensagem de transação, que contém a chave pública do “B”, assinando com sua chave privada. A transação, e, portanto, uma transferência de propriedade dos Bitcoins é registrada, carimbada com data e hora e exposto em um bloco chamado de *blockchain*. A criptografia de chave pública garante que todos os computadores da rede tenham um registro constantemente atualizado e verificado de todas as transações dentro da rede Bitcoin, o que impede o gasto duplo e qualquer tipo de fraude (ULRICH, 2017).

Embora o Bitcoin seja uma das mais famosas aplicações da *blockchain*, ela pode ser utilizada em outras aplicações, muito além das criptomoedas, como contratos inteligentes, serviços públicos, *internet* das coisas, sistemas de reputação e serviços de segurança. Isto deve-se ao fato que as principais características da tecnologia *blockchain* são a descentralização, persistência, anonimato e auditabilidade (ZHENG et al, 2016).

A *blockchain* é uma sequência de blocos de dados que armazena registros públicos das operações na rede. Cada bloco aponta para o bloco imediatamente anterior (chamado de bloco pai) por meio de uma referência que essencialmente é um valor de *hash*, sendo que o primeiro bloco é chamado de bloco gênese e não possui bloco pai (ZHENG et al, 2016).

É um desafio para os nós da rede obterem consenso em um ambiente não confiável, onde podem existir nós maliciosos. Para resolver o problema vários

algoritmos de consenso podem ser implementados, como *proof-of-work* (utilizado pelo Bitcoin) e *proof-of-stake*.

Para dar clareza ao escopo do trabalho proposto é preciso fazer uma diferenciação entre *blockchain* pública e *blockchain* privada. Massessi (2018) distingue estas duas classificações de *blockchain*, sendo que a *blockchain* pública é uma *blockchain* sem permissão, ou seja, qualquer um pode entrar na rede, o que significa que eles podem ler, escrever ou participar da *blockchain* pública. Nas *blockchains* públicas ninguém tem controle sobre a rede, porém elas são *blockchains* seguras, pois os dados não podem ser alterados depois de validados.

A *blockchain* privada é uma *blockchain* autorizada. As redes autorizadas impõem restrições sobre quem pode participar da rede e em quais transações. Basicamente, tudo começa com o gerenciamento de identidade e, em uma *blockchain* privada, é possível saber quem são os participantes desde o início.

Desta forma, a proposta deste trabalho é usar uma *blockchain privada*, para criar uma rede de votação confiável dentro do departamento acadêmico de informática da UTFPR campus Ponta Grossa, onde vários setores, departamentos, diretórios e/ou indivíduos pertencentes à instituição poderão atuar como nós dentro do sistema, garantindo maior lisura ao processo.

1.1 OBJETIVOS

O objetivo geral do trabalho é apresentado na seção 1.1.1 e na seção 1.1.2 são apresentados os objetivos específicos.

1.1.1 Objetivo Geral

Demonstrar a aplicabilidade da *blockchain* na simulação de sistema de votação para chefe de departamento na UTFPR campus Ponta Grossa, mais especificamente no departamento acadêmico de informática, visando a transparência, confiabilidade e eficiência do processo.

1.1.2 Objetivos Específicos

- Atribuir chaves criptográficas aos eleitores e candidatos envolvidos no processo eleitoral;
- Garantir o anonimato da eleição, o voto do eleitor não deve ser conhecido nem por si mesmo nem por terceiros;
- Discutir os fatores de segurança e desempenho do sistema.

1.2 JUSTIFICATIVA

Observe-se que:

- 1) A facilidade de realização de votações por meio de dispositivos ligados em rede, serve de chamariz para atrair possíveis eleitores dentro de uma organização onde o voto não é obrigatório;
- 2) No serviço público, a transparência é compreendida como um pressuposto básico servindo como princípio fundamental para assegurar a probidade de uma ação;
- 3) A estrutura da *blockchain* dificulta a fraude por agentes maliciosos, uma vez que é necessário que o *hash* de blocos posteriores a uma transação também seja alterado para modificar uma transação pertencente a um bloco anterior.

Além disso, é perceptível o interesse ao redor do mundo de empregar a *blockchain* a um sistema de votação ao observar as constatações de (OSGOOD, 2016, p. 16):

O primeiro uso da tecnologia de votação da *blockchain* aconteceu na Dinamarca quando o a Danish Liberal Alliance usou um sistema de votação *blockchain* para realizar uma votação interna em abril de 2014 durante sua reunião anual. (...). Em fevereiro de 2016, a Ucrânia e os Estados Unidos assinaram um memorando para desenvolver um sistema de votação baseado no Ethereum (Ethereum é uma plataforma que utiliza a *blockchain*). A Rússia também anunciou em abril de 2016 que havia desenvolvido e testado com sucesso um sistema de votação por procuração baseado em *blockchain*. Finalmente, o pequeno partido australiano Flux defendeu pelo uso do voto *blockchain* nas eleições australianas (OSGOOD, 2016, p. 16).

Tendo isto em vista, este trabalho visa a utilização de um sistema seguro, transparente e participativo por meio da *blockchain*, tecnologia que combina criptografia e sistemas distribuídos. Com todos os avanços tecnológicos que vem

ocorrendo ao redor do mundo, os sistemas de votação não devem ficar defasados e para garantir a total legitimidade do processo, a *blockchain* talvez possa garantir a confiabilidade e a segurança almejadas.

1.3 MÉTODO UTILIZADO

O trabalho tem como base um código aberto em python que pode ser encontrado em <https://github.com/dvf/blockchain> e possui licença MIT. A *blockchain* foi alterada para comportar algumas mudanças propostas no trabalho, como por exemplo a retirada da recompensa pela mineração, o ajuste de sua dificuldade, o registro de nós confiáveis por um administrador do sistema.

Quanto a votação propriamente dita, primeiramente é preciso definir quem é elegível para votar dentro da estrutura organizacional da instituição e isto deve ocorrer fora da *blockchain*, garantindo segurança ao procedimento. No caso da UTFPR, certos eleitores podem ser considerados como elegíveis para votar e uma vez definido que tal indivíduo é elegível, este será cadastrado no sistema e permitirá que ele vote apenas uma vez.

Os votos serão registrados em blocos de dados e estes são gradativamente vinculados à *blockchain* pelos nós definidos como confiáveis. Então a quantidade de votos recebida por algum candidato dentro da rede poderá ser conferida ao fim da votação.

1.4 ORGANIZAÇÃO DO TRABALHO

Este trabalho está organizado em seis capítulos. O Capítulo 2 dispõe sobre a legislação que serviu como motivação ao desenvolvimento do trabalho quanto à legislação pertinente à eleição departamental da UTFPR. O Capítulo 3 detalha o funcionamento das principais tecnologias utilizadas pela *blockchain*, estrutura imprescindível ao sistema proposto no trabalho.

O Capítulo 4 expõe os métodos utilizados, descrevendo as etapas executadas demonstrando seu progresso e a explicação do código para possibilitar a análise do sistema. Prosseguindo o Capítulo 5 apresenta os resultados obtidos com o sistema eleitoral adotado. Por fim, o Capítulo 6 trata das conclusões e contribuições deste trabalho, além de sugestões de trabalhos futuros.

2 LEGISLAÇÃO QUE REGULAMENTA O SERVIÇO PÚBLICO

Este Capítulo apresenta as leis e regulamentos do serviço público, em especial detalha o processo de eleição de chefe de departamento na Universidade Tecnológica Federal do Paraná campus Ponta Grossa, o qual se encontra dividido em quatro seções sendo que a seção 2.1 relata, com base na Constituição Federal Brasileira, os princípios que devem ser seguidos pela gestão pública. A seção 2.2 apresenta a Lei de Acesso à Informação, nº 12.527, de 2011. A seção 2.3 traz o atual funcionamento do processo eleitoral para chefe de departamento da UTFPR. Finalizando, tem-se a seção 2.4 que apresenta as considerações do Capítulo.

2.1 SUSTENTAÇÃO CONSTITUCIONAL

A Constituição Federal Brasileira torna explícitos os princípios pertinentes à gestão pública em seu Art. 37: “A administração pública direta e indireta de qualquer dos Poderes da União, dos Estados, do Distrito Federal e dos Municípios obedecerá aos princípios de legalidade, impessoalidade, moralidade, publicidade e eficiência (...)”. Serão apresentados apenas os princípios que estão relacionados ao trabalho.

- Princípio Da Legalidade

O princípio da legalidade nada mais é que a consequência de um estado democrático, visto que o estado brasileiro é regido por leis, as mesmas tem que afirmar o desejo da liberdade de expressão, por parte da população. Isso é assegurado no seu Art. 5º, inciso II, “ninguém será obrigado a fazer ou deixar de fazer alguma coisa senão em virtude de lei” (BRASIL, 1988).

Sendo assim, pode-se dizer que esse princípio nada mais é que uma garantia constitucional, protegendo os indivíduos contra arbítrios cometidos pelo Estado. Portanto, a sociedade tem ampla liberdade para fazer o que quiser, desde que não seja um ato, um comportamento ou uma atividade proibida por lei (PEREIRA, 2012).

- Princípio Da Publicidade

O princípio da publicidade visa tornar públicos os atos da administração pública, garantindo conhecimento e transparência ao cidadão. O princípio da publicidade é uma extensão da cidadania, pois permite o controle social do Poder Público nas mãos dos cidadãos. Esse princípio exige que uma verdadeira República,

que visa o interesse público e de agentes públicos, deve ser absolutamente transparente e controlável.

Segundo Culau e Fortis (2006) a transparência exerce a função de aproximar o Estado da sociedade, à medida que amplia o nível de acesso do cidadão às informações da gestão pública, além de se constituir como um dos requisitos fundamentais da boa governança.

- Princípio Da Eficiência

Pela definição léxica a palavra eficiência trata-se da capacidade de realizar tarefas ou trabalhos de modo eficaz e com o mínimo de desperdício. Meirelles (2013, p. 102) define este princípio na esfera da administração pública da seguinte maneira:

O Princípio da eficiência exige que a atividade administrativa seja exercida com presteza, perfeição e rendimento funcional. É o mais moderno princípio da função administrativa, que já não se contenta em se desempenhar apenas com uma legalidade, exigindo resultados positivos para o serviço público e satisfatório atendimento às necessidades da comunidade e de seus membros. (MEIRELLES, 2013, p. 102).

2.2 LEI DE ACESSO À INFORMAÇÃO

A Lei de Acesso à Informação nº 12.527, de 18 de novembro de 2011, define informação como “dados, processados ou não, que podem ser utilizados para produção e transmissão de conhecimento, contidos em qualquer meio, suporte ou formato”. O direito à informação está previsto na Constituição Federal de 1988 desde sua promulgação, no inciso XXXIII do Art. 5º:

XXXIII - todos têm direito a receber dos órgãos públicos informações de seu interesse particular, ou de interesse coletivo ou geral, que serão prestadas no prazo da lei, sob pena de responsabilidade, ressalvadas aquelas cujo sigilo seja imprescindível à segurança da sociedade e do Estado;

No inciso II do § 3º do Art. 37:

§ 3º A lei disciplinará as formas de participação do usuário na administração pública direta e indireta, regulando especialmente: (Redação dada pela Emenda Constitucional nº 19, de 1998)

(...)

II - o acesso dos usuários a registros administrativos e a informações sobre atos de governo, observado o disposto no Art. 5º, X e XXXIII; (Incluído pela Emenda Constitucional nº 19, de 1998) (Vide Lei nº 12.527, de 2011)

Apesar de ter suporte constitucional ainda carecia de um instrumento legislativo que regulasse este direito. Neste contexto, a Lei de Acesso à Informação nº 12.527, de 18 de novembro de 2011, revela-se significativa pois tem o objetivo de tornar menos obscuro o conhecimento da informação por parte dos cidadãos, em relação à forma como os recursos públicos são administrados (FIGUEIREDO e SANTOS, 2013).

Medeiros, Magalhães e Pereira (2014) declaram que o acesso à informação potencializa as políticas públicas implementadas pelos governos, desde seu planejamento até sua execução e avaliação, pela participação dos cidadãos. O Art. 6 define o que deve ser assegurado pelo poder público quanto às informações:

Art. 6º Cabe aos órgãos e entidades do poder público, observadas as normas e procedimentos específicos aplicáveis, assegurar a:

- I - gestão transparente da informação, propiciando amplo acesso a ela e sua divulgação;
- II - proteção da informação, garantindo-se sua disponibilidade, autenticidade e integridade; e
- III - proteção da informação sigilosa e da informação pessoal, observada a sua disponibilidade, autenticidade, integridade e eventual restrição de acesso.

Para esclarecer os termos utilizados pelo Art. 6, a lei define em seu Art. 4:

Art. 4º Para os efeitos desta Lei, considera-se:

(...)

- VI - disponibilidade: qualidade da informação que pode ser conhecida e utilizada por indivíduos, equipamentos ou sistemas autorizados;
- VII - autenticidade: qualidade da informação que tenha sido produzida, expedida, recebida ou modificada por determinado indivíduo, equipamento ou sistema;
- VIII - integridade: qualidade da informação não modificada, inclusive quanto à origem, trânsito e destino;

O § 2 do Art. 8 prevê a obrigatoriedade e o uso da *internet* na divulgação das informações:

Art. 8º É dever dos órgãos e entidades públicas promover, independentemente de requerimentos, a divulgação em local de fácil acesso, no âmbito de suas competências, de informações de interesse coletivo ou geral por eles produzidas ou custodiadas.

(...)

§ 2º Para cumprimento do disposto no caput, os órgãos e entidades públicas deverão utilizar todos os meios e instrumentos legítimos de que dispuserem, sendo obrigatória a divulgação em sítios oficiais da rede mundial de computadores (*internet*).

Além do aspecto do uso da tecnologia da informação e comunicação como apoio ao acesso às informações públicas, a lei também estabelece prazos para a obtenção de informações junto aos órgãos públicos.

2.3 LEGISLAÇÃO VIGENTE NA UTFPR

O sistema de eleição de chefes de departamento da UTFPR é definido pelo Regulamento Para Eleição De Chefes De Departamentos Acadêmicos Da UTFPR, aprovado pela resolução nº. 037/13 do Conselho de Graduação e Educação Profissional da instituição. Este regulamento serve de parâmetro para as regras que devem ser respeitadas e implementadas no *software*. O capítulo ressalta os artigos mais expressivos deste regulamento:

O Art. 3 trata dos eleitores da escolha do chefe de departamento:

Art.3º - Poderão votar, para a escolha do Chefe de Departamento Acadêmico, os professores efetivos e os técnicos administrativos nele lotados, pertencentes ao Quadro de Pessoal da UTFPR.

Parágrafo Único - Terão igual direito a voto os professores e técnicos-administrativos efetivos afastados, total ou parcialmente, de suas atividades.

O Art. 6 trata da organização da comissão responsável pela realização da eleição:

Art.6º- A eleição para Chefe de Departamento será organizada e conduzida, no campus, por uma Comissão de Eleições designada pelo Diretor Geral do Campus e, se necessário, por subcomissões designadas pelo presidente da Comissão de Eleições.

Parágrafo único -A Comissão de Eleições poderá ser composta por um professor representante de cada um dos Departamentos Acadêmicos existentes no campus.

O Art. 9 trata como a votação é feita atualmente pela legislação vigente na UTFPR: “Art. 9º - A votação será feita por escrutínio secreto e por meio de cédula única, fornecida pela Mesa que preside a eleição, rubricada pelo Presidente da Comissão de Eleições ou do presidente da subcomissão, se houver.”. Este artigo traz o fator sigilo do voto à discussão.

O Art. 10 trata como o eleitor deve se apresentar à votação atualmente vigente na UTFPR: “Art.10 - Ao apresentar-se no local de votação, o eleitor identificar-se-á mediante a apresentação de um documento de identidade ou carteira funcional e

assinará a Lista Nominal de Comparecimento ao Pleito. Parágrafo Único- Não serão aceitos votos por procuração.” Já o Art. 11 estabelece o que o eleitor só possui direito à um voto dentre os constantes da lista nominal de candidatos de seu departamento acadêmico.

O Art. 18 trata da definição do vencedor da eleição departamental:

Art.18 - Será considerado eleito e nomeado pelo Diretor Geral do Campus o candidato que obtiver o maior número de votos válidos, tendo sido assegurado o comparecimento de pelo menos 50% dos votantes do Departamento Acadêmico.

Parágrafo Único - No caso em que o quorum de votantes for inferior a 50% num Departamento Acadêmico, far-se-á uma nova eleição nesse Departamento Acadêmico em data a ser definida e divulgada pela Comissão de Eleição, respeitando-se o disposto no Art. 23 deste regulamento.

O Art. 21 trata das informações constantes na ata do processo eletivo:

Art.21 - Encerrada a apuração dos votos a Comissão de Eleições ou Subcomissão de Eleições lavrará uma Ata do processo eletivo, na qual constará: I. o número de eleitores presentes e ausentes; II. as irregularidades constatadas; III. o nome dos professores votados por Departamento Acadêmico e; IV. os sufrágios obtidos. V. A lista de presença.

Parágrafo Único- Poderá a Subcomissão fazer constar em ata além do que dispõem o “caput” deste artigo qualquer acontecimento ou informação que julgue importante, em relação ao pleito.

Por fim o Art. 22 define que os resultados das eleições deverão ser consolidados em relatório, pela Comissão de Eleições, e encaminhados para homologação do Diretor Geral do Campus.

2.4 CONSIDERAÇÕES DO CAPÍTULO

A disposição legislativa deste Capítulo é relevante ao trabalho por atuar como base e justificativa para o desenvolvimento do mesmo, isto se dá primeiramente pelo embasamento na própria constituição brasileira, revelando alguns dos requisitos necessários do *software* à medida em que se entra no regulamento específico da instituição. A escolha pelo uso da *blockchain*, que armazena dados de difícil adulteração, e de um sistema distribuído transparente é solidificada pela Lei de Acesso à Informação nº 12.527, de 2011, que dispõe sobre a transparência dos atos da gestão pública que assegura o amplo acesso à informação e a proteção da

informação, ao passo que garante a sua disponibilidade, autenticidade, integridade e, quando for o caso, seu sigilo.

O sistema proposto deve trazer mais confiabilidade ao processo eleitoral, uma vez que serve como instrumento para que os interessados possam fiscalizá-lo. Também traz o aspecto de eficiência sobre o processo, uma vez que os votos poderão ser mais facilmente processados e auditados que na votação atualmente feita em cédulas.

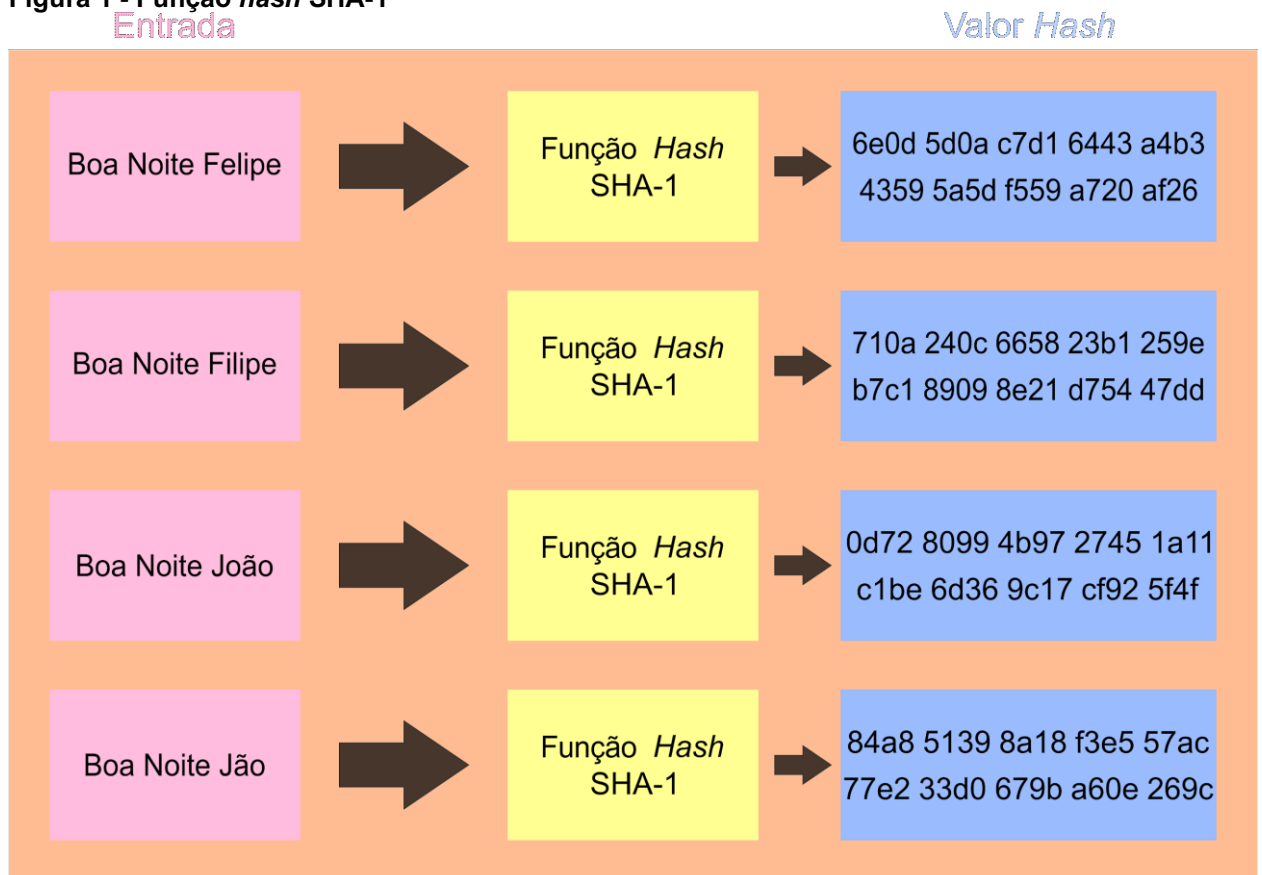
3.TECNOLOGIAS APLICADAS À *BLOCKCHAIN*

Este Capítulo apresenta as principais tecnologias que possibilitaram a criação da *blockchain*, cada seção detalha cada uma destas tecnologias, sendo que a seção 3.1 traz o funcionamento básico de funções *hash*. A seção 3.2 trata da problemática dos algoritmos de consenso relacionados a confiabilidade da informação circulante em sistemas distribuídos. A seção 3.3 informa os principais métodos de criptografia utilizados. A seção 3.4 fornece uma visão geral da arquitetura da *blockchain*. Conclui-se com a seção 3.5 que apresenta as considerações sobre o Capítulo.

3.1 FUNÇÕES *HASH*

A figura 1 ilustra o funcionamento de uma função *hash*:

Figura 1 - Função *hash* SHA-1



Fonte: Autoria própria (2019)

Uma função *hash* é uma função que transforma dados de comprimento variável para dados de comprimento fixo. Os valores retornados por uma função *hash* são

chamados de valores *hash*. Na criptografia, são necessários requisitos de segurança adicionais: se os dados de entrada são desconhecidos, deve ser difícil reconstruí-los apenas conhecendo o valor *hash* e deve ser difícil encontrar entradas colidindo, ou seja, duas entradas gerarem uma mesma saída (PRENEEL, 1998).

As colisões podem ser evitadas usando valores de *hash* maiores, usando várias funções de *hash* no processo ou ainda criando uma tabela de redirecionamento quando valores duplicados de *hash* são encontrados. Estas funções são usadas em diversos casos, por exemplo: criptografia, compressão, geração de *checksum* (tem a finalidade de validar a integridade de arquivos) e a criação de índices em dados (CHRISTENSSON, 2018). Exemplos comuns de funções *hash* criptográficas são MD5 e SHA-1.

3.2 ALGORITMOS DE CONSENSO

Em sistemas de computação distribuídos, um componente, como um servidor, pode aparecer tanto como falho quanto funcional a um sistema de detecção de falhas. Estes componentes podem apresentar diferentes sintomas a diferentes observadores, isto é conhecido como “culpa bizantina”. É difícil para os outros componentes declararem que o servidor falhou e o desligarem da rede, por que eles primeiramente precisam obter consenso a respeito de qual componente falhou, isto é conhecido como “falha bizantina” (DRISCOLL, 2004).

Este problema em computação é chamado de “Problema dos Generais Bizantinos”, apresentado no Anexo A do trabalho. O objetivo do “Problema Dos Generais Bizantinos” é ser capaz se defender contra falhas de componentes do sistema que impeça que outros componentes cheguem a um acordo entre si, para a operação correta do sistema (LAMPORT, 1982).

Na *blockchain*, como chegar a um consenso entre os nós não confiáveis é uma transformação deste problema já que se trata de uma rede distribuída. Nesta rede não há nó central que garanta que os registros das transações em nós distribuídos sejam todos iguais. Os nós precisam não confiar em outros. Assim, algumas abordagens são necessárias para garantir que os registros em nós diferentes sejam consistentes (ZHENG et al, 2016). Atualmente os algoritmos de consenso mais comuns para abordagem da temática são o *Proof-of-Work* (PoW) e *Proof-of-Stake* (PoS).

3.2.1 *Proof-of-Work (PoW)*

De acordo com (SOLVING, 2018), o Bitcoin, que é o pioneiro na implementação e o uso da *blockchain* implementa o PoW como um algoritmo de consenso. Nele há nós chamados de mineradores, responsáveis por verificarem as transações que ocorrem na rede. Os mineradores executam um programa que dá chance proporcional a seu poder de computação para realizar esta tarefa. Eles competem para encontrar uma resposta (um valor *hash*) para um bloco de dados. Este valor *hash* é uma saída que se baseia nas entradas do bloco, como as transações que ele contém.

Os mineiros buscam esse *hash* combinando as entradas do bloco com um número aleatório conhecido como *nonce*, até alguém encontrar a resposta correta. Então essa solução é transmitida, verificada por outros mineiros e quando confirmada, o bloco é adicionado à *blockchain* (cadeia de blocos). Os mineiros então usam este novo bloco como entrada para o *hash* necessário para o próximo bloco, formando desta forma um registro de todas as transações desde do início do Bitcoin.

Uma vez que os nós concordam em um acordo majoritário entre eles sem precisar qualquer autoridade central, apesar da rede não ser instantânea, e mesmo com a presença de partes potencialmente não confiáveis, e o PoW consegue resolver o “Problema Dos Generais Bizantinos”. Ela permite que os generais distribuídos e não coordenados cheguem a um acordo:

1. Os generais concordam que o primeiro plano recebido e verificado por todos os generais será aceito como o plano a ser seguido;
2. Um general resolve o problema PoW e transmite a solução para a rede para que todos os generais o recebam;
3. Verificando a solução, cada general trabalha na solução do próximo problema de PoW, incorporando a solução anterior ao novo problema.
4. Cada vez que um problema de PoW é resolvido, um bloco é gerado e adicionado à cadeia de blocos, e então ela continuamente vai crescendo. Qualquer general trabalhando em uma solução diferente com o tempo mudará para a cadeia mais longa, pois é a cadeia que a maioria dos generais está contribuindo e tem a maior chance de sucesso.

O PoW também impede a adulteração dos dados históricos da cadeia, uma vez que armazena a assinatura *hash* do bloco anterior em cada novo bloco. Portanto,

qualquer alteração em um bloco anterior exigiria que todos os blocos sucessivos também fossem alterados e isto demandaria uma quantidade excessiva de poder de computação, protegendo desta forma o registro de transações.

O PoW também resolve o problema em determinar a representação da maioria na tomada de decisão. Se a maioria fosse baseada onde um endereço IP valesse um voto, isto poderia ser subvertido por qualquer pessoa capaz de alocar muitos IPs. No PoW, essencialmente, um CPU equivale a um voto. (NAKAMOTO, 2008).

Porém há dois problemas proeminentes na implementação do PoW: o potencial de centralização - visto que com o desenvolver da rede um indivíduo comum dificilmente conseguiria minerar um bloco com seu poder computacional sozinho - e ao desperdício de energia gerado para proteger a rede - os cálculos não têm aplicação em outros campos. Devido essa ineficiência de recursos o emprego de muitas *blockchains* optam por utilizar o *Proof-of-Stake*, ao invés do *Proof-of-Work*.

3.2.1 *Proof-of-Stake (PoS)*

Conforme (SOLVING, 2018), o algoritmo de consenso PoS funciona pedindo aos usuários que bloqueiem seus ativos na forma de um depósito. Como seus ativos estão bloqueados, o usuário (aqui chamado como validador e não minerador) pode começar a validar novos blocos de transações. O algoritmo seleciona um validador para processar o próximo bloco através de um processo quase aleatório, mas quanto mais fundos o validador apostar, maior a chance de ele ser escolhido. Outros fatores podem ser levados em conta para randomizar o processo, como o período de tempo em que os fundos do validador foram usados. Um validador desonesto é punido com a perda de seus fundos.

Desta forma o PoS, também resolve o “Problema Dos Generais Bizantinos”:

1. Os generais tornam-se candidatos a validadores depositando e bloqueando seus ativos;
2. O algoritmo seleciona um general para se tornar um validador para o próximo bloco, o qual ele cria;
3. Posteriormente, outro general é escolhido para se tornar um validador e faz referências sobre o bloco anterior para formar uma cadeia crescente;

4. Após um período de tempo, é possível saber se um número suficiente de outros generais está trabalhando na mesma cadeia e então os outros generais podem migrar para ela.

Isso resolve o “Problema Dos Generais Bizantinos” e elimina o alto custos de energia e hardware associados ao PoW. O PoS dá a todos os usuários uma chance igual, proporcional aos seus fundos, de participar da validação dos blocos. Isto permite participações mais justas entre o validador com muitos fundos e outro com poucos fundos, levando a uma rede mais descentralizada que o PoW, porém ainda gerando potenciais problemas de centralização, já que quem tem mais fundos tem mais chances de participar como validador da rede. As criptomoedas, costumam recompensar o trabalho de minerador e validador com moedas na própria rede, gerando um ciclo vicioso no PoS: quem tem mais fundos, tem mais chances de validar as transações e ser recompensada com mais fundos por isto.

3.3 CRIPTOGRAFIA

A fim de assegurar a confiabilidade da informação é imprescindível garantir a segurança desta em sistemas e serviços computacionais. Neste cenário, surgem alguns fundamentos de segurança da informação (OLIVEIRA, 2012):

1. Disponibilidade: uma informação deve estar disponível para acesso no momento desejado;
2. Integridade: o conteúdo da mensagem não deve ser alterado.
3. Controle de acesso: o conteúdo da mensagem deve ser acessado somente por pessoas autorizadas;
4. Autenticidade: a identidade de quem está enviando a mensagem deve ser garantida;
5. Não-repudição: deve-se prevenir que terceiros neguem o envio e/ou recebimento de uma mensagem;
6. Privacidade: deve-se impedir que pessoas não autorizadas tenham acesso ao conteúdo da mensagem.

A criptografia, conjunto de princípios e técnicas empregadas para cifrar a escrita, com o intuito torná-la ininteligível a quem não tem acesso autorizado a ela é um dos maiores artifícios utilizados nesse âmbito. Nas subseções seguintes serão

detalhados alguns dos métodos mais comuns utilizados em criptografia, relevantes ao trabalho.

3.3.1 Criptografia Simétrica

A criptografia simétrica baseia-se em dois elementos principais: o algoritmo de cifragem e a chave criptográfica. A criptografia simétrica faz uso de uma única chave, que é compartilhada entre o emissor e o destinatário de um conteúdo.

Uma chave criptográfica é um valor que atua sobre um algoritmo de cifragem. Imagine que a chave criptográfica é a chave da porta de sua casa e sua fechadura é o algoritmo. Quando uma chave é colocada na fechadura, cada um dos pinos dentro dela é movido para uma posição específica e estas posições, postas pela chave, são as que a fechadura precisa para ser aberta, ela abre, caso contrário, não (ROMAGNOLO, 2017). Na Figura 2 é exemplificado o uso deste tipo de criptografia:

Figura 2 - Criptografia simétrica



Fonte: Autoria própria (2019)

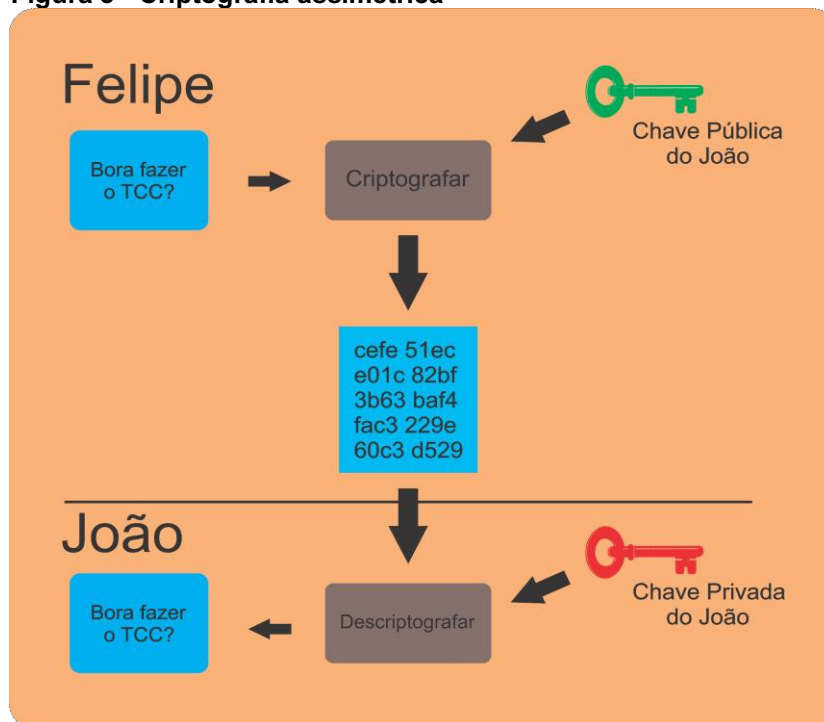
A principal vantagem é a simplicidade, apresentando facilidade de uso e rapidez para executar os processos criptográficos, porém isto traz um problema que deve-se ao fato que a chave de ciframento é a mesma utilizada para deciframento ou uma pode facilmente ser obtida da outra, ou seja, qualquer um que tenha acesso à

senha poderá descobrir o conteúdo secreto da mensagem. Além disso, a criptografia simétrica não garante os princípios de autenticidade e não-repudição (OLIVEIRA, 2012).

3.3.2 Criptografia Assimétrica

Neste tipo de criptografia são utilizadas duas chaves, uma pública e outra privada, diferentes e complementares. A chave pública, como o próprio nome insinua, pode estar acessível a qualquer pessoa que deseje se comunicar de modo seguro, porém a chave privada deve ficar em posse somente de cada titular. A chave privada é responsável por decodificar uma mensagem criptografada para ele com a sua respectiva chave pública. Desta maneira, é garantida a confiabilidade da mensagem, desde que a chave privada segura, posto que quem possuir acesso a esta chave terá acesso à mensagem (OLIVEIRA, 2012). A Figura 3 exemplifica esta técnica:

Figura 3 - Criptografia assimétrica



Fonte: Autoria própria (2019)

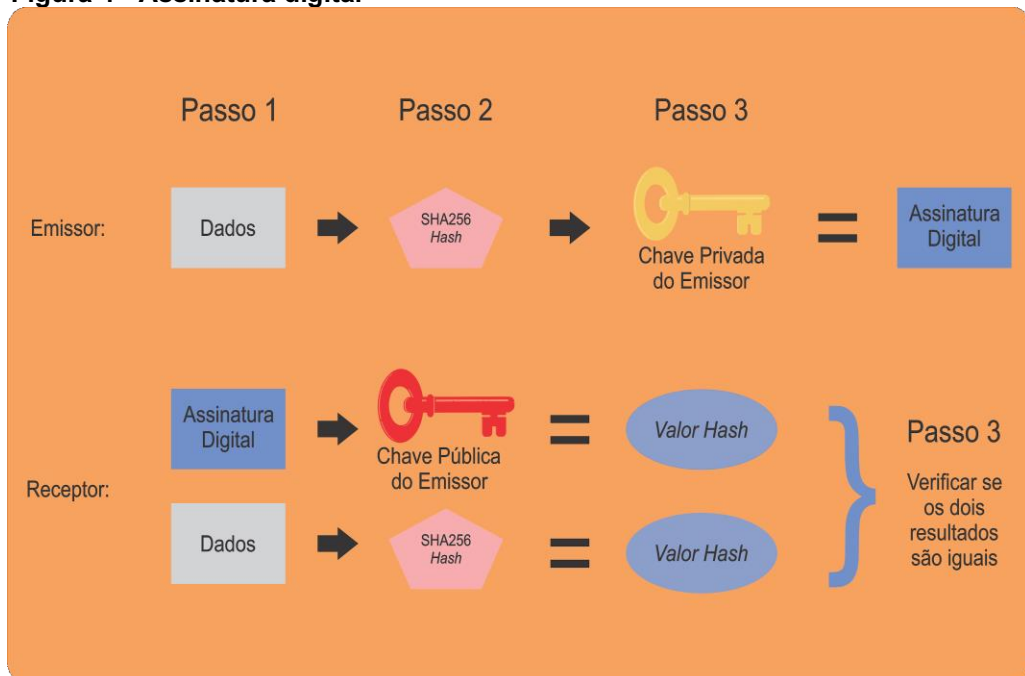
A vantagem deste método é a segurança, já que não é necessário e nem prudente compartilhar a chave privada. Por outro lado, a desvantagem é que o tempo de processamento de mensagens neste tipo criptografia é maior que na criptografia simétrica (OLIVEIRA, 2012).

3.3.3 Assinatura Digital

A assinatura digital consiste em um processo de inversão do sistema de criptografia assimétrica. O autor assina uma mensagem usando sua chave privada para cifrá-la e ela pode ser decifrada utilizando a chave pública do autor, confirmando sua identidade e reconhecendo que a mensagem não foi adulterada, uma vez que utiliza funções *hash* no processo. Este método assegura a autenticidade, integridade e não-repudição da mensagem, entretanto, não assegura sua confidencialidade, pois é decifrada utilizando a chave pública do emissor (OLIVEIRA, 2012).

No Bitcoin a chave privada é usada para assinar as transações que são espalhadas por toda a rede e, em seguida, são acessadas por chaves públicas, visíveis para todos na rede. O algoritmo de assinatura digital de curva elíptica (ECDSA) é frequentemente utilizado em *blockchains* (ZHENG et al, 2016). A Figura 4 elucida o procedimento:

Figura 4 - Assinatura digital



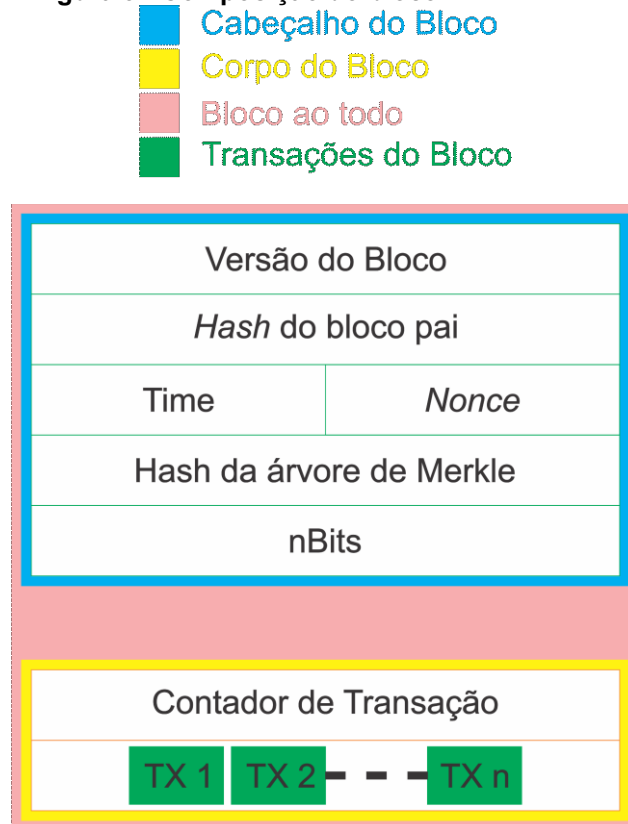
Fonte: Autoria própria (2019)

3.4 ARQUITETURA DA BLOCKCHAIN

A *blockchain* é uma sequência de blocos de dados que armazena registros públicos das operações na rede. Cada bloco aponta para o bloco imediatamente

anterior (chamado de bloco pai) por meio de uma referência que essencialmente é um valor de *hash*, sendo que o primeiro bloco é chamado de bloco gênese e não possui bloco pai (ZHENG et al, 2016). A Figura 5 ilustra a composição do bloco:

Figura 5 - Composição do bloco



Fonte: Autoria própria (2019)

No Bitcoin um bloco consiste de um cabeçalho e de um corpo. O cabeçalho inclui (BITCOIN, 2019):

- 1) Versão do bloco: indica quais regras de validação deverão ser seguidas;
- 2) *Hash* do bloco pai: um valor de *hash* de 256 bits (SHA256) que aponta para o bloco anterior. Isso garante que nenhum bloco anterior possa ser alterado sem alterar também o cabeçalho desse bloco;
- 3) *Hash* da árvore de Merkle: valor *hash* de todas as transações incluídas no bloco;
- 4) *Time*: Tempo de bloqueio é um período quando o minerador começou a fazer *hash* no cabeçalho do bloco (de acordo com o mineiro). Deve ser estritamente maior que o tempo médio dos 11 blocos anteriores. Nós completos (que possuem toda a *blockchain*) não aceitarão blocos com cabeçalhos com mais de duas horas passadas;

- 5) *nBits*: uma versão codificada do alvo que aparece no cabeçalho do bloco. O alvo é o limiar abaixo do qual um *hash* de cabeçalho deve estar para que o bloco seja válido. Quando os mineradores estão procurando por um bloco válido, eles criam uma grande quantidade de candidatos e blocos válidos devem ter um *hash* abaixo desse alvo.
- 6) *Nonce*: Um número arbitrário que os mineiros alteram para modificar o *hash* do cabeçalho para produzir um *hash* menor ou igual ao limite alvo.

Já o corpo do bloco é composto por transações e um contador de transações. O máximo número de transações que um bloco pode conter depende do tamanho do bloco e do tamanho cada transação (ZHENG et al, 2016).

3.5 CONSIDERAÇÕES DO CAPÍTULO

Este Capítulo traz as tecnologias que tornaram possível a criação da *blockchain*. Entender o processo de assinatura digital e algoritmos de consenso permitiu criar um banco de dados distribuído, rastreável, transparente e seguro.

O trabalho utiliza o algoritmo de consenso *Proof-Of-Work* na implementação do *software*, uma vez que o *Proof-Of-Stake* é dependente da quantidade de fundos dos validadores e neste trabalho isto é irrelevante, já que os validadores não possuem acúmulo de ativos, pois estes são votos.

4 DESENVOLVIMENTO

Este Capítulo apresenta as ferramentas utilizadas no desenvolvimento do trabalho começando pela seção 4.1 que descreve a interface web implementada. A seção 4.2 é tratada a estrutura da *blockchain* utilizada no trabalho. A seção 4.3 dispõe sobre os métodos utilizados para a coleta de resultados. Conclui-se com a seção 4.4 que apresenta as considerações sobre o Capítulo.

4.1 INTERFACE WEB

Para o desenvolvimento do trabalho, foi criada uma interface web, utilizando o *framework* PHP Laravel. A interface tem o objetivo de:

1. Centralizar as ações do administrador responsável por criar e iniciar a votação, cadastrando os candidatos e eleitores.
2. Fazer a comunicação com a *blockchain* registrando novos nós na rede e lançando transações para estes nós.
3. Servir como interface intuitiva para os eleitores, tornando o processo de votação e averiguação do resultado em si simples e claro.

Foi criado um banco de dados SQL, para gerenciar os eleitores e candidatos, estruturado como representado nos Quadro 1 e 2 respectivamente:

Quadro 1 - Tabela de Candidatos

Tabela: Candidatos		
Atributos	Tipagem	Descrição
id	bigint(20)	Chave primária da tabela
created_at	timestamp	Timestamp de criação
updated_at	timestamp	Timestamp de atualização
nome	varchar(255)	Armazena o nome do candidato

Fonte: Autoria própria (2020)

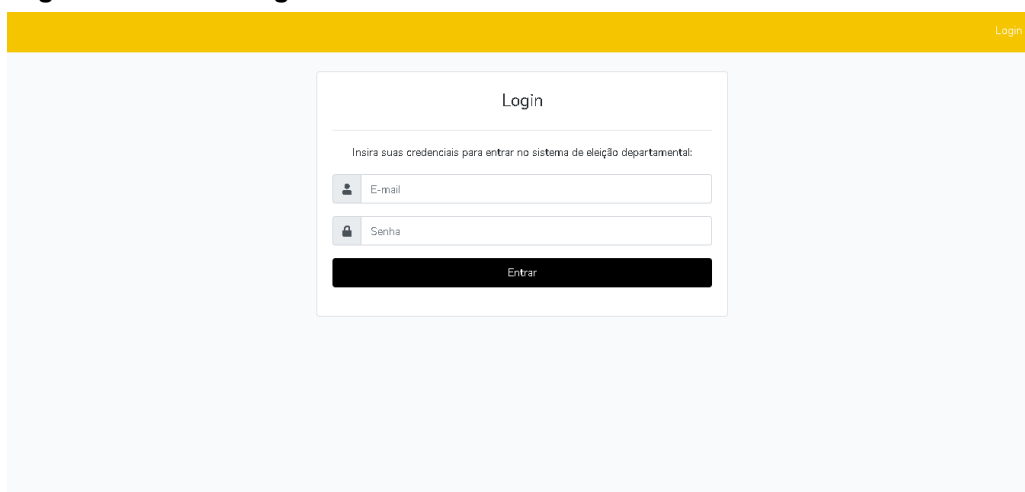
Quadro 2 - Tabela de Usuários

Tabela: Users		
Atributos	Tipagem	Descrição
id	bigint(20)	Chave primária da tabela
name	varchar(255)	Armazena o nome do candidato
email	varchar(255)	Email para login do usuário
created_at	timestamp	Timestamp de criação
updated_at	timestamp	Timestamp de atualização
email_verified_at	timestamp	Timestamp de verificação de e-mail
password	varchar(255)	Senha de login do usuário
remember_token	varchar(100)	Usado para validação de cookie
administrador	tinyint(1)	Verificação se o usuário tem função de administrador
eleitor	tinyint(1)	Verificação se o usuário tem função de eleitor
senha_redefinida	varchar(255)	Usado para redefinição de senha
jaVotou	tinyint(1)	Verificação se o eleitor já votou

Fonte: Autoria própria (2020)

4.1.1 Tela de Login

Interface onde o usuário realiza o acesso ao sistema. Feito o *login* o sistema verifica a permissão do usuário e habilita as funções referentes a mesma (Figura 6).

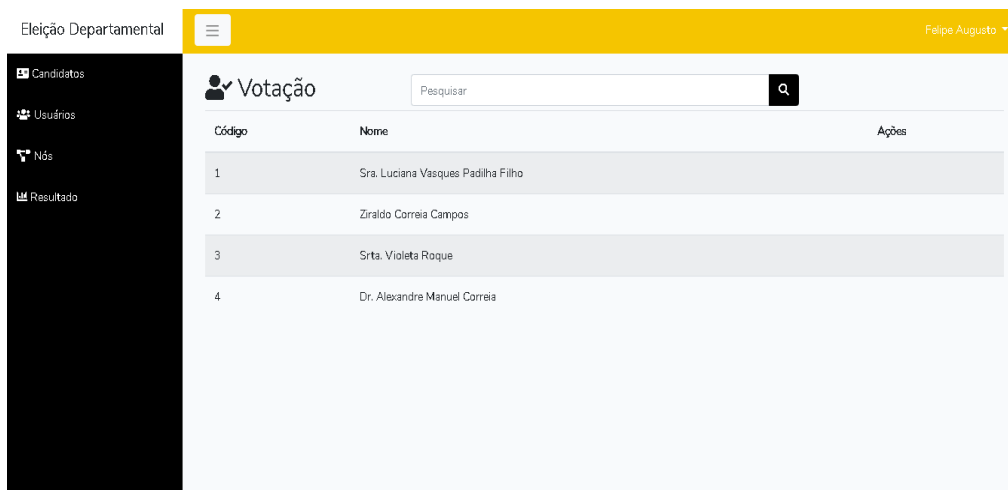
Figura 6 - Tela de Login

A tela de login apresenta um formulário centralizado com o título "Login". Abaixo do título, há uma instrução: "Insira suas credenciais para entrar no sistema de eleição departamental:". O formulário contém dois campos de entrada: "E-mail" com um ícone de usuário e "Senha" com um ícone de cadeado. Abaixo dos campos, há um botão "Entrar" em um fundo escuro.

Fonte: Autoria própria (2019)

4.1.2 Tela Inicial

Após realizar o *login* no sistema, é apresentada a interface dos candidatos disponíveis para a votação (Figura 7).

Figura 7 - Tela Inicial

A tela inicial do sistema de eleição departamental apresenta uma barra superior amarela com o texto "Eleição Departamental" e "Fevereiro Agosto". À esquerda, há um menu lateral com opções: "Candidatos", "Usuários", "Nós" e "Resultado". O conteúdo principal mostra a seção "Votação" com um campo de busca "Pesquisar". Abaixo, há uma tabela com os seguintes dados:

Código	Nome	Ações
1	Sra. Luciana Vasques Padilha Filho	
2	Zivaldo Correia Campos	
3	Srta. Violeta Roque	
4	Dr. Alexandre Manuel Correia	

Fonte: Autoria própria (2019)

4.1.3 Candidatos

Tela que disponibiliza as operações de criação, leitura, exclusão e atualização do cadastro dos candidatos (Figura 8). Somente administradores podem efetuar essas ações.

Figura 8 - Tela de Candidatos

Código	Nome	Total Votos	Ações
1	Sra. Luciana Vasques Padilha Filho	0	Visualizar Excluir
2	Zivaldo Correia Campos	0	Visualizar Excluir
3	Srta. Violeta Roque	0	Visualizar Excluir
4	Dr. Alexandre Manuel Correia	0	Visualizar Excluir

Fonte: Autoria própria (2019)

4.1.4 Usuários

Tela com o cadastro de usuários segue o mesmo padrão do de Candidatos, disponibilizando as operações, criação, leitura, exclusão e atualização cadastral (Figura 9). Somente usuários com cargo de administrador podem efetuar alterações no cadastro dos usuários.

Figura 9 - Tela de Usuários

Código	Nome	Ações
1	Felipe Augusto	Visualizar Excluir
2	João Victor	Visualizar Excluir

Fonte: Autoria própria (2019)

4.1.5 Registro de nós

Registro de nós que irão participar da *blockchain*, atuando como mineradores (Figura 10). Os nós não são todos os eleitores, mas sim toda entidade confiável para executar um mesmo nó. Por exemplo, cada campus da UTFPR poderia rodar ser um nó na rede.

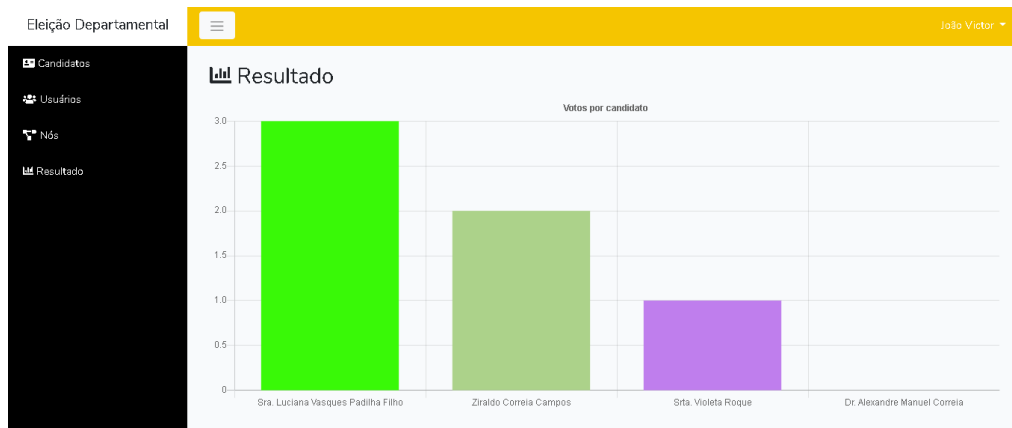
Figura 10 - Tela para Registro de Nós

Fonte: Autoria própria (2019)

4.1.6 Resultado

Tela para apresentar o resultado das eleições, essa tela está disponível a todos os usuários do sistema (Figura 11).

Figura 11 - Tela de Resultados



Fonte: Autoria própria (2019)

4.2 BLOCKCHAIN

A *blockchain* utilizada como base do projeto foi extraída do GitHub, plataforma de hospedagem de código-fonte e possui licença MIT.¹ A linguagem de programação utilizada no código é Python e nas próximas subseções serão descritos alguns métodos do código.

¹ Link: <https://github.com/dvf/blockchain>

4.2.1 Criação do bloco gênese

O bloco gênese é o primeiro bloco criado arbitrariamente para que os blocos subsequentes com dados possam ser vinculados à ele. Blocks.txt é a *blockchain* salva em arquivo, no nó que está participando da rede (Figura 12).

Figura 12 - Criação do Bloco Gênese

```
class Blockchain:
    def __init__(self):
        data = json.load(open('blocks.txt'))

        self.current_transactions = []
        self.chain = data
        self.nodes = set()

        # Create the genesis block
        self.new_block(previous_hash='1', proof=100)
    def save(self):
        import json
        with open('blocks.txt', 'w', encoding='utf8') as outfile:
            json.dump(self.chain, outfile, ensure_ascii=False)
```

Fonte: Autoria própria (2020)

O código já trata de algumas instâncias necessárias em sua execução, como demonstrado na Figura 13:

Figura 13 - Inicialização do Código

```
# Instantiate the Node
app = Flask(__name__)

# Generate a globally unique address for this node
node_identifier = str(uuid4()).replace('-', '')

# Instantiate the Blockchain
blockchain = Blockchain()
```

Fonte: Autoria própria (2020)

4.2.2 Criação de novas transações e novos blocos

Cada novo bloco contém o *hash* do bloco anterior, então se um agente malicioso conseguir alterar um bloco anterior da cadeia, todos os blocos subsequentes irão conter hashes incorretos. Cada nova transação é adicionada a uma lista de transações, que pertencerá ao próximo bloco minerado (Figura 14).

Figura 14 - Novas transações e novos blocos

```
def new_block(self, proof, previous_hash):
    """
    Create a new Block in the Blockchain

    :param proof: The proof given by the Proof of Work algorithm
    :param previous_hash: Hash of previous Block
    :return: New Block
    """

    block = {
        'index': len(self.chain) + 1,
        'timestamp': time(),
        'transactions': self.current_transactions,
        'proof': proof,
        'previous_hash': previous_hash or self.hash(self.chain[-1]),
    }

    # Reset the current list of transactions
    self.current_transactions = []

    self.chain.append(block)
    return block

def new_transaction(self, recipient, amount):
    """
    Creates a new transaction to go into the next mined Block

    :param recipient: Address of the Recipient
    :param amount: Amount
    :return: The index of the Block that will hold this transaction
    """
    self.current_transactions.append({
```



```

        'recipient': recipient,
        'amount': amount,
    })

    return self.last_block['index'] + 1

@property
def last_block(self):
    return self.chain[-1]

```

Fonte: Autoria própria (2020)

4.2.3 Mineração, PoW e verificação de *hash*

O PoW é o algoritmo que determina como novos blocos são criados ou extraídos da *blockchain*. Em nosso código, o objetivo do PoW é descobrir um número que, quando for passado o *hash* do novo bloco com o *hash* do bloco anterior, será produzido um *hash* que iniciará com quatro zeros, definido no atributo `guess_hash` do código. A dificuldade de encontrar esse número pode ser aumentada, adicionando mais números zeros necessários inicialmente e pode ser diminuída, subtraindo os mesmos. O que garante a 'imutabilidade' da *blockchain* é que computacionalmente este número deve ser difícil de encontrar, porém fácil de verificar se está o *hash* gerado por ele está correto (Figura 15).

Figura 15 - PoW

```

@staticmethod
def hash(block):
    """
    Creates a SHA-256 hash of a Block
    :param block: Block
    """

    # We must make sure that the Dictionary is Ordered, or we'll have inconsistent
    hashes

    block_string = json.dumps(block, sort_keys=True).encode()
    return hashlib.sha256(block_string).hexdigest()

def proof_of_work(self, last_block):
    """
    Simple Proof of Work Algorithm:

```

```

- Find a number p' such that hash(pp') contains leading 4 zeroes
- Where p is the previous proof, and p' is the new proof

:param last_block: <dict> last Block
:return: <int>
"""

last_proof = last_block['proof']
last_hash = self.hash(last_block)

proof = 0
while self.valid_proof(last_proof, proof, last_hash) is False:
    proof += 1

return proof

@staticmethod
def valid_proof(last_proof, proof, last_hash):
    """
    Validates the Proof
    :param last_proof: Previous Proof
    :param proof: Current Proof
    :return: True if correct, False if not.
    """
    guess = f'{last_proof}{proof}{last_hash}'.encode()
    guess_hash = hashlib.sha256(guess).hexdigest()
    return guess_hash[:4] == "0000"

```

Fonte: Autoria própria (2020)

4.2.4 Registro de nós

Por se tratar de uma *blockchain* privada todos os nós válidos na rede devem ser devidamente registrados pelo administrador do sistema, conforme demonstrado no código da Figura 16.

Figura 16 - Registro de Nós

```

def register_node(self, address):
    """
    Add a new node to the list of nodes

```

```

:param address: Address of node. Eg. 'http://192.168.0.5:5000'
"""

parsed_url = urlparse(address)
if parsed_url.netloc:
    self.nodes.add(parsed_url.netloc)
elif parsed_url.path:
    # Accepts an URL without scheme like '192.168.0.5:5000'.
    self.nodes.add(parsed_url.path)
else:
    raise ValueError('Invalid URL')

```

Fonte: Autoria própria (2020)

4.2.5 Consenso entre os nós

Para resolver o problema em que um nó possua uma cadeia diferente de outro nó, a regra adotada é que a cadeia válida será a mais longa na rede. Usando esse algoritmo, chegou-se ao consenso entre os nós da rede.

O método `valid_chain()` é utilizado para verificar se uma cadeia é válida percorrendo cada bloco e verificando o *hash* e se o número do PoW está correto. Já o método `resolve_conflicts()` percorre todos os nossos nós vizinhos, e verifica suas cadeias usando o método `valid_chain()`. Se uma cadeia válida for encontrada e seu comprimento for maior que o do nó atual, ela substituirá a cadeia do nó atual, o código dos métodos citados são apresentados na Figura 17.

Figura 17 - Consenso entre Nós

```

def resolve_conflicts(self):
    """
    This is our consensus algorithm, it resolves conflicts
    by replacing our chain with the longest one in the network.

    :return: True if our chain was replaced, False if not
    """

    neighbours = self.nodes
    new_chain = None

    # We're only looking for chains longer than ours

```

```

max_length = len(self.chain)

# Grab and verify the chains from all the nodes in our network
for node in neighbours:
    response = requests.get(f'http://{node}/chain')

    if response.status_code == 200:
        length = response.json()['length']
        chain = response.json()['chain']

        # Check if the length is longer and the chain is valid
        if length > max_length and self.valid_chain(chain):
            max_length = length
            new_chain = chain

# Replace our chain if we discovered a new, valid chain longer than
ours
if new_chain:
    self.chain = new_chain
    return True

return False

def valid_chain(self, chain):
    """
    Determine if a given blockchain is valid
    :param chain: A blockchain
    :return: True if valid, False if not
    """

    last_block = chain[0]
    current_index = 1

    while current_index < len(chain):
        block = chain[current_index]
        print(f'{last_block}')
        print(f'{block}')
        print("\n-----\n")
        # Check that the hash of the block is correct
        last_block_hash = self.hash(last_block)
        if block['previous_hash'] != last_block_hash:

```

```

        return False

    # Check that the Proof of Work is correct
    if not self.valid_proof(last_block['proof'], block['proof'],
last_block_hash):
        return False

    last_block = block
    current_index += 1

    return True

```

Fonte: Autoria própria (2020)

4.2.6 Rotas de requisições HTTP

Para a aplicação web se comunicar com os nós que estão executando a *blockchain* é utilizada uma API com as requisições HTTP. As rotas HTTP utilizadas no trabalho estão listadas a seguir (Figura 18):

1. `/transactions/new`: o nó cria uma nova transação em um bloco e a retorna como resposta;
2. `/mine`: o nó irá minerar um novo bloco e retorná-lo como resposta;
3. `/chain`: o nó retorna a sua *blockchain* completa;
4. `/save`: grava a *blockchain* em arquivo;
5. `/nodes/register`: registra um novo nó na lista de nós e devolve como resposta a lista de todos os nós registrados.
6. `/nodes/resolve`: aplica o consenso entre os nós e substitui a *blockchain* deste nó se houver outra válida e maior em outro nó registrado.

Figura 18 - Rotas de Requisições HTTP

```

@app.route('/save', methods=['GET'])
def saveBlockchain():
    blockchain.save()
    return jsonify({'success': True}), 200

@app.route('/mine', methods=['GET'])

```

```
def mine():
    # We run the proof of work algorithm to get the next proof...
    last_block = blockchain.last_block
    proof = blockchain.proof_of_work(last_block)

    # We must receive a reward for finding the proof.
    # The sender is "0" to signify that this node has mined a new coin.
    # blockchain.new_transaction(
    #     sender="0",
    #     recipient=node_identifier,
    #     amount=1,
    # )

    # Forge the new Block by adding it to the chain
    previous_hash = blockchain.hash(last_block)
    block = blockchain.new_block(proof, previous_hash)

    response = {
        'message': "New Block Forged",
        'index': block['index'],
        'transactions': block['transactions'],
        'proof': block['proof'],
        'previous_hash': block['previous_hash'],
    }
    return jsonify(response), 200

@app.route('/transactions/new', methods=['POST'])
def new_transaction():
    values = request.get_json()

    # Check that the required fields are in the POST'ed data
    required = ['recipient', 'amount']
    if not all(k in values for k in required):
        return 'Missing values', 400

    # Create a new Transaction
    index = blockchain.new_transaction(values['recipient'], values['amount'])

    response = {'message': f'Transaction will be added to Block {index}'}
    return jsonify(response), 201
```

```
@app.route('/chain', methods=['GET'])
def full_chain():
    response = {
        'chain': blockchain.chain,
        'length': len(blockchain.chain),
    }
    return jsonify(response), 200

@app.route('/nodes/register', methods=['POST'])
def register_nodes():
    values = request.get_json()

    nodes = values.get('nodes')
    if nodes is None:
        return "Error: Please supply a valid list of nodes", 400

    for node in nodes:
        blockchain.register_node(node)

    response = {
        'message': 'New nodes have been added',
        'total_nodes': list(blockchain.nodes),
    }
    return jsonify(response), 201

@app.route('/nodes/resolve', methods=['GET'])
def consensus():
    replaced = blockchain.resolve_conflicts()

    if replaced:
        response = {
            'message': 'Our chain was replaced',
            'new_chain': blockchain.chain
        }
    else:
        response = {
            'message': 'Our chain is authoritative',
```

```
        'chain': blockchain.chain
    }

    return jsonify(response), 200
```

Fonte: A autoria própria (2020)

4.3 CONSIDERAÇÕES DO CAPÍTULO

O código da aplicação encontra-se no *link* github.com/VictorLiima/TCCBlockchain. Também há um demonstrativo do sistema no endereço youtube.com/watch?v=s22CZk1eiRQ. Os requerimentos para executar o sistema podem ser encontrados na própria página do GitHub. Primeiramente todos os nós responsáveis pela *blockchain* devem estar ativos. Também o servidor web e SQL devem estar em funcionamento para a execução da interface web. As rotas das requisições HTTP, pelas quais o nó se comunica com o sistema, podem ser encontradas na seção 4.2.6.

Após concluir o login, o administrador do sistema deve cadastrar os candidatos, os eleitores e cadastrar os nós que irão participar da rede. Após serem tomadas estas providências a eleição está pronta para começar. Ao acessar a rota HTTP da corrente (*/chain*) podemos ver algumas informações do bloco, como seu índice, as transações pertencentes a ele, o *hash* do bloco anterior, o *timestamp* e o número *proof* utilizado na execução do *proof-of-work*. Através das rotas */mine* e */nodes/resolve* o nó irá minerar um bloco e realizar o consenso entre os nós.

Desta forma, este Capítulo apresentou como foi construído o projeto para aplicar a teoria exposta nos Capítulos 2 e 3. Quanto à legislação vigente ao processo de eleição UTFPR, a votação a ser feita por escrutínio secreto e por meio de cédula única não pode ser cumprida pois foi proposto outra forma de eleição, outras normas devem ser cumpridas fora do sistema proposto como a identificação e adição do eleitor, a definição da Comissão de Eleições e o relatório final para ser homologado pelo diretor do campus.

Quanto às técnicas utilizadas pela *blockchain*, foram utilizados o *hash*, algoritmos de consenso e uma arquitetura similar à utilizada pelo Bitcoin. No trabalho não foram necessárias a geração de endereços públicos e privados pelo próprio

software, uma vez que há uma comissão que definirá os candidatos e eleitores em um banco de dados SQL.

5 RESULTADOS

Baseando-se nos objetivos específicos identificados para esse trabalho, cada seção analisa os resultados obtidos para cada um deles. A seção 5.1 apresenta a atribuição de chaves para cada indivíduo dentro da *blockchain*. A seção 5.2 trata sobre o anonimato de cada eleitor. A seção 5.3 discorre sobre as características intrínsecas ao sistema: segurança e desempenho. Conclui-se com a seção 5.4 que apresenta as considerações sobre o Capítulo.

5.1 ENDEREÇOS PRIVADOS

No *Bitcoin* um par de chave pública e privada são geradas dentro da própria rede, para garantir independência do sistema, ou seja, qualquer indivíduo pode gerar estas chaves e utilizá-las como endereços próprios sem precisar de nenhuma outra identificação ou de intermediários para participar da rede. Já em um sistema eleitoral todos os participantes, candidatos, eleitores e administradores devem ser devidamente identificados por uma entidade centralizadora, portanto, a geração de um par de chaves foi vista como desnecessária em neste projeto e os dados dos participantes referidos foram inseridos no banco de dados SQL. É possível criptografar o nome dos eleitores para obter maior dificuldade em rastrear os votos. De cada

5.2 ANONIMATO

Tendo em vista que:

1. O eleitor não deve poder comprovar em quem votou, pois poderia utilizar deste artifício para vender o seu voto; e
2. O eleitor não pode ter seu voto reconhecido por outra entidade.

Em um primeiro momento, foi decidido atribuir uma identidade aleatória enviada para o eleitor no momento que ele confirmasse seu voto, para que qualquer indivíduo externo ao sistema não soubesse quem era o eleitor de um determinado voto, apenas o remetente. Porém isto gerou dois problemas, o primeiro é que esta identidade aleatória gerada pelo servidor centralizado poderia ser descoberta por agentes maliciosos do servidor web/SQL. O segundo é que a pessoa teria um comprovante de

seu voto, podendo verificar na *blockchain* a identidade atribuída a ele, revelando o voto de um determinado candidato.

Então decidiu-se remover o remetente da transação voto na *blockchain*. Apenas inserindo o candidato escolhido e a *timestamp* do voto. Desta forma seria possível descobrir apenas a identidade do eleitor se fossem monitoradas as alterações no banco de dados SQL do campo “jaVotou” correspondente, se o usuário já votou e associá-la ao *timestamp* do voto na *blockchain* de um determinado eleitor, hora em que este fosse alterado. Para mascarar isto poderia-se colocar um atraso aleatório no *timestamp*, tornando-se difícil fazer esta associação não interferindo no resultado da eleição.

Porém, mesmo se houvesse um sistema que resolvesse esses fatores, é iminente a chance de ser burlada externamente ao sistema, por exemplo, alguém filmar o seu processo de votação para comprovar e seu voto e vendê-lo. Por isso, a única solução para realmente garantir o anonimato é o isolamento na hora da votação e isto só pode ser alcançado se os eleitores comparecerem fisicamente a um recinto isolado para votarem.

5.3 SEGURANÇA E DESEMPENHO

O problema do ataque dos 51%, encontrado no *Bitcoin*, que diz que se um minerador possuir mais que 50% do poder de mineração ele poderia temporariamente reverter a *blockchain* e manipular transações na rede não se aplica, uma vez que, ao contrário do *Bitcoin*, os mineradores são selecionados por uma entidade centralizadora e, portanto, devem ser confiáveis. Além disso, o poder computacional requerido para validar uma transação pode ser ajustado de acordo com a quantidade de transações e mineradores.

O desempenho é inferior se comparado a um esquema clássico de banco de dados SQL, ou mesmo NoSQL, pois estes não dependem do algoritmo de consenso *Proof-of-work*. Porém, a diferença de performance não é substancial uma vez que a eleição tem prazo de término (em oposição ao *Bitcoin* que sempre precisa movimentar dados) e no caso de eleições pequenas, como a eleição departamental, há poucos agentes e transações (votos) no sistema.

5.4 CONSIDERAÇÕES DO CAPÍTULO

Este Capítulo refere-se ao cumprimento dos objetivos específicos propostos pelo trabalho. A geração de endereços privados não foi vista como necessária por tratar-se de uma *blockchain* privada. O anonimato pode ser alcançado dentro do sistema, porém é impossível garanti-lo se a votação não for feita em um local controlado. A integridade dos dados e segurança do sistema é aumentada em relação a um sistema SQL, proporcionalmente a quantidade de nós mineradores e a dificuldade de mineração. O desempenho é inferior à um sistema centralizado, porém não substancialmente a ponto de interferir na realização da eleição. Não foram encontrados métodos eficazes para medir segurança e o desempenho do sistema e assim compará-los objetivamente a um sistema centralizado com banco de dados SQL.

6 CONSIDERAÇÕES FINAIS

Este capítulo está dividido em duas seções que discorrem sobre as conclusões obtidas do trabalho realizado e indicadores de futuras aplicações complementando o presente estudo.

6.1 CONCLUSÃO

Neste trabalho foram detalhados os princípios da *blockchain* sendo proposto um sistema eleitoral *online* baseado na mesma, permitindo uma avaliação da possibilidade de substituição do método utilizado atualmente, com urna e voto secreto. Foi proposta uma interface web, um sistema descentralizado privado em que os nós eram escolhidos por um administrador e uma *blockchain* para armazenar os votos. O sistema *online* visava aumentar o interesse em votar de eleitores, aumentar a segurança, o desempenho e a facilidade em votar.

Técnicas clássicas de uma *blockchain* foram utilizadas, como o *hashing*, nós descentralizados e o algoritmo de consenso *proof-of-work*. Contudo, o maior desafio encontrado foi como manter o anonimato de um voto secreto feito *online*, em que nem o eleitor e nem a mesa organizadora da eleição poderiam comprovar em quem o eleitor votou. O trabalho não obteve êxito neste objetivo e concluiu-se que é impossível fazer uma votação *online* que consiga manter totalmente o anonimato, sendo necessário os eleitores estarem em um recinto fechado com fiscais a fim de garantir que o voto seja totalmente secreto.

Quanto a atribuição de endereços privados, ou a geração de pares de chaves públicas e privadas para cada indivíduo dentro do sistema foi constatado desnecessária sua aplicação, uma vez que os eleitores precisam ter um endereço conhecido pelo administrador do sistema. Por este motivo, não há razão para um eleitor gerar vários endereços a sua vontade.

A segurança do sistema proposto é superior ao sistema atual, uma vez que um indivíduo mal intencionado sozinho não pode facilmente modificar o resultado das eleições por conta própria e a segurança de modificar uma *blockchain* pode ser regulada pela dificuldade em resolver o *hash* proposto pelo algoritmo de consenso. Por outro lado, o desempenho é menor se comparado à um banco SQL comum, uma vez que os nós devem decidir entre si qual a maior cadeia de transações e se

sincroniza.

O trabalho conclui inviável a aplicação de um sistema eleitoral baseado em *blockchain* pelo fato que garantir o anonimato de um sistema *online* melhor do que remeter fisicamente seu voto em uma urna é impraticável. Nos outros quesitos se mostrou viável e a aplicação de um sistema próprio eleitoral.

6.2 TRABALHOS FUTUROS

Por tratar-se de uma tecnologia que se encontra em estágios inicial oferecendo muitas oportunidades a serem exploradas, existindo vários estudos que seriam possíveis de serem realizados, como a substituição da *blockchain* por um *directed acyclic graph*. Quanto ao trabalho em si, é pertinente a um sistema eleitoral conseguir eliminar a centralização dos dados de um servidor para à própria *blockchain*, mantendo dados sensíveis protegidos, descentralizando-os de um servidor central onde é mais suscetível a agentes maliciosos. Melhoras quanto ao anonimato do indivíduo também são passíveis de análise.

REFERÊNCIAS

- BITCOIN Developer Reference. **Bitcoin.org**. Disponível em: <<https://bitcoin.org/en/developer-reference>>. Acesso em 20 jul. 2019.
- BRASIL, Constituição. **Constituição da República Federativa do Brasil**. Brasília, DF: Senado Federal: Centro Gráfico, 1988.
- BRASIL. **Lei nº 12.527**, de 18 de novembro de 2011.
- CHRISTENSSON, Per. Hash Definition. **TechTerms**, 21 abril 2018. Disponível em: <<https://techterms.com/definition/hash>>. Acesso em: 21 jun. 2019.
- CULAU, Ariosto Antunes; FORTIS, Martin Francisco de Almeida. **Transparência e controle social na administração pública brasileira: avaliação das principais inovações introduzidas pela Lei de Responsabilidade Fiscal**. 2006.
- DA SILVA FIGUEIREDO, Vanuza; DOS SANTOS, Waldir Jorge Ladeira. **Transparência e controle social na administração pública**. Temas de Administração Pública, v. 8, n. 1, 2013.
- DRISCOLL, Kevin et al. **The real byzantine generals**. In: The 23rd Digital Avionics Systems Conference (IEEE Cat. No. 04CH37576). IEEE, 2004. p. 6. D. 4-61.
- LAMPORT, Leslie; SHOSTAK, Robert; PEASE, Marshall. **The Byzantine generals problem**. ACM Transactions on Programming Languages and Systems (TOPLAS), v. 4, n. 3, p. 382-401, 1982.
- LEE, KIBIN et al. Electronic voting service using block-chain. **Journal of Digital Forensics, Security and Law**, v. 11, n. 2, p. 8, 2016.
- MASSESSI, Demiro. **Public Vs Private Blockchain In A Nutshell**. 2018. Disponível em: <<https://medium.com/coinmonks/public-vs-private-blockchain-in-a-nutshell-c9fe284fa39f>>. Acesso em: 28 mar. 2019.
- PEREIRA, Luciana Freitas. **O princípio da legalidade na Constituição Federal: análise comparada dos princípios da reserva legal, legalidade ampla e legalidade**

estrita. 2012. Disponível em: <<https://www.direitonet.com.br/artigos/exibir/7125/O-principio-da-legalidade-na-Constituicao-Federal-analise-comparada-dos-principios-da-reserva-legal-legalidade-ampla-e-legalidade-estrita>>. Acesso em: 19 mai. 2019.

MEDEIROS, Simone Assis; MAGALHÃES, Roberto; PEREIRA, José Roberto. **Lei de acesso à informação**: em busca da transparência e do combate à corrupção. *Informação & informação*, v. 19, n. 1, p. 55-75, 2014.

MEIRELLES, Hely Lopes. **Direito Administrativo Brasileiro**. Ed. Malheiros, 2013.

NAKAMOTO, Satoshi. **Bitcoin: A Peer-to-Peer Electronic Cash System**. 2008.

OLIVEIRA, Ronielton Rezende. **Criptografia simétrica e assimétrica-os principais algoritmos de cifragem**. *Segurança Digital [Revista online]*, v. 31, p. 11-15, 2012.
OSGOOD, Ryan. **The future of democracy: Blockchain voting**. COMP116: Information Security, 2016.

PRENEEL, Bart. **The state of cryptographic hash functions**. School organized by the European Educational Forum. Springer, Berlin, Heidelberg, 1998. p. 158-182.

ROMAGNOLO, Cesar Augusto. O que é Criptografia? **Oficinadanet**, 24 agosto 2017. Disponível em: <https://www.oficinadanet.com.br/artigo/443/o_que_e_criptografia>. Acesso em: 21 jun. 2019.

SOLVING the Byzantine Generals Problem with Proof of Stake (PoS). 2018. **Radixdlt.com**. Disponível em: <<https://www.radixdlt.com/post/why-proof-of-stake-is-a-powerful-alternative>>. Acesso em 20 jul. 2019.

SOLVING the Byzantine Generals Problem with Proof of Work. 2018. **Radixdlt.com**. Disponível em: <<https://www.radixdlt.com/post/what-is-proof-of-work/>>. Acesso em 20 jul. 2019.

ULRICH, Fernando: **Bitcoin: a moeda da era digital**. Ed. LVM Editora, 2017.

ZHENG, Zibin et al. **Blockchain challenges and opportunities: A survey**. 2016.

ANEXO A - O Problema Dos Generais Bizantinos

Segundo Lamport et al. (1982), imagine que há diversas divisões de um exército bizantino realizando o cerco de uma cidade inimiga. Os generais podem se comunicar entre si apenas com mensageiros. Depois de observar o inimigo eles devem decidir um plano comum de ação, entretanto podem existir traidores tentando dissuadir os generais leais de chegarem a um acordo. Os generais precisam de um algoritmo que garanta que:

- A. Todos os generais leais decidam o mesmo plano de ação. O algoritmo deve garantir a condição A independentemente do que os traidores fazem. Os generais leais não devem apenas chegar a um acordo, mas devem concordar por um plano razoável. Por isso, também deve-se assegurar que
- B. Um pequeno número de traidores não pode fazer com que os generais leais adotem plano ruim. A condição B é difícil de formalizar, pois requer definir precisamente o que é um plano ruim, e nós não tentamos fazê-lo. Em vez disso, considera-se como os generais devem chegar a uma decisão. Cada general observa o inimigo e comunica suas observações aos outros. Seja $v(i)$ a informação comunicada pelo i -ésimo general. Cada general usa algum método para combinar os valores $v(1) \dots v(n)$ em um único plano de ação, onde n é o número de generais.

A condição A é atingida fazendo com que todos os generais usem o mesmo método para combinar a informação, e a condição B é atingida usando um método robusto. Por exemplo, se a única decisão a ser feita é se devem atacar ou recuar, então $v(i)$ pode ser a opinião do general i sobre qual a melhor alternativa, e a decisão final pode ser baseada após uma votação majoritária entre eles. Um pequeno número de traidores pode afetar decisão apenas se os generais leais estivessem quase igualmente divididos entre as duas possibilidades, caso em que nenhuma decisão poderia ser considerada ruim. (...). O método óbvio para se comunicarem é o i -ésimo general enviar $v(i)$ por mensageiro para o outro general. No entanto, isso não funciona, porque para satisfazer a condição A exige que todo general leal obtenha o mesmos valores $v(1) \dots v(n)$ e um general traidor pode enviar valores diferentes para diferentes generais. Para que a condição A seja satisfeita, o seguinte deve ser verdadeiro:

1. Todo general leal deve obter as mesmas informações $v(1) \dots, v(n)$.
2. Se o i -ésimo geral é leal, então o valor que ele envia deve ser usado por todos os generais leais como o valor de $v(i)$.

Podemos, portanto, restringir nossa consideração ao problema de como um único general envia seu valor para os outros.