

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

AFONSO HAFNER DE BRITTO

**Integração à Rede Blockchain Brasil:
Deploy de Nó Observador e Avaliação de
Contratos Inteligentes**

Monografia apresentada como requisito parcial
para a obtenção do grau de Bacharel em Ciência
da Computação

Orientador: Prof. Dr. Eder John Scheid

Porto Alegre
2025

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Reitora: Prof^ª. Marcia Cristina Bernardes Barbosa

Vice-Reitor: Prof. Pedro de Almeida Costa

Pró-Reitora de Graduação: Prof^ª. Nádyá Pesce da Silveira

Diretor do Instituto de Informática: Prof. Luciano Paschoal Gaspary

Coordenador do Curso de Ciência de Computação: Prof. Álvaro Freitas Moreira

Bibliotecário-chefe do Instituto de Informática: Alexsander Borges Ribeiro

AGRADECIMENTOS

Agradeço a todos e todas que fizeram parte da minha trajetória enquanto estudante de Ciência Computação. Gostaria de destacar o agradecimento à minha família, em especial meu pai e minha mãe por terem possibilitado tudo isso, desde o começo, à minha namorada, Marina, que está comigo desde antes do meu ingresso na Universidade e esteve comigo em todos os momentos que vivi enquanto estudante. Não podia deixar de mencionar os meus colegas de curso, que hoje posso chamar de amigos. E todos os professores que fizeram parte da história que construí.

RESUMO

Este trabalho apresenta a implantação e a avaliação de um nó observador (observer) na Rede Blockchain Brasil (RBB), bem como a análise dos contratos inteligentes executados nessa infraestrutura. Inicialmente, descreve-se a arquitetura da RBB, detalhando os diferentes tipos de nós e as formas de interação entre os diversos atores que compõem a rede. Em seguida, são examinados os contratos disponibilizados no repositório oficial da plataforma. Por fim, os contratos inteligentes são validados e discutem-se os principais desafios enfrentados durante o processo de implantação do nó observador na infraestrutura do Instituto de Informática (INF) da Universidade Federal do Rio Grande do Sul (UFRGS).

Palavras-chave: Blockchain. rede blockchain brasil. contratos inteligentes.

Integration to the Rede Blockchain Brasil: Deploy of an Observer Node and Evaluation of Smart Contracts

ABSTRACT

This work presents the deployment and evaluation of an observer node in the Rede Blockchain Brasil (RBB), as well as the analysis of the smart contracts executed on this infrastructure. It begins by describing the architecture of the RBB, detailing the different types of nodes and the forms of interaction among the various actors that compose the network. Next, it examines the contracts made available in the platform's official repository. Finally, the smart contracts are validated, and the main challenges encountered during the deployment of the observer node within the infrastructure of the Institute of Informatics (INF) at the Federal University of Rio Grande do Sul (UFRGS) are discussed.

Keywords: blockchain. rede blockchain brasil. smart contracts..

LISTA DE FIGURAS

Figura 2.1	Diagrama simplificado do process de <i>hashing</i> usando o algoritmo SHA256	16
Figura 2.2	Diagrama simplificado da estrutura de um bloco	18
Figura 2.3	Diagrama simplificado da estrutura de uma rede <i>blockchain</i>	19
Figura 2.4	Tipos de livros razão distribuídos	23
Figura 4.1	Representação gráfica da topologia da RBB	41
Figura 4.2	Legenda da representação da topologia da RBB	41
Figura 4.3	Exemplo de Explorador de Blocos da RBB	42
Figura 4.4	Visualização de transações da RBB.....	43
Figura 4.5	Visualização de informações a partir de um endereço de participante da RBB	43
Figura 4.6	Visualização de informações a partir de um endereço de participante da RBB	44
Figura 4.7	Esquema de permissionamento da RBB.....	45
Figura 5.1	Seleção da versão do compilador e da EVM.....	56
Figura 5.2	Verificação do contrato <i>AccountIngress</i>	57
Figura 5.3	Contratos que já foram verificados usando o explorador de blocos	58
Figura 5.4	Contratos <i>AccountIngress</i> verificado	58
Figura 5.5	Comparação do <i>bytecode</i> presente no bloco gênese com o <i>bytecode</i> gerado pelo explorador de blocos	58
Figura 5.6	Resultado compilação <i>AccountIngress</i>	59
Figura 5.7	Comparação do <i>bytecode</i> presente no bloco gênese com o <i>bytecode</i> gerado pelo <i>Remix</i>	60
Figura 5.8	Segunda parte do resultado da comparação de <i>bytecodes</i>	60
Figura 5.9	<i>Screenshot</i> da tentativa de instalação do nó observador	63

LISTA DE TABELAS

Tabela 2.1	Vantagens da implantação e uso de contratos inteligentes	26
Tabela 2.2	Barreiras e limitações dos contratos inteligentes.....	28
Tabela 3.1	Usos de <i>blockchain</i> por governos e problemas relacionados	36
Tabela 4.1	Participantes que já adicionaram blocos na RBB	40

LISTA DE ABREVIATURAS E SIGLAS

NAPR	Agência Nacional de Registro Públicos
AADCE	Algoritmo de Assinatura Digital com Curvas Elípticas
IDE	Ambiente de desenvolvimento integrado
BaaS	<i>Blockchain-as-a-Service</i>
BACEN	Banco Central
BNDES	Banco Nacional de Desenvolvimento Econômico
BPC	Banco Popular da China
EUA	Estados Unidos da América
EVM	<i>Ethereum Virtual Machine</i>
EBSI	<i>European Blockchain Services Infrastructure</i>
IDD	Identidade Digital Descentralizada
INF	Instituto de Informática
LAC	<i>LaCoin</i>
MDBC	Moedas Digitais de Banco Central
PDT	Prova de Trabalho
PDP	Prova de Participação
PDA	Prova de Autoridade
DREX	Real Digital
RBB	Rede <i>Blockchain</i> Brasil
TCU	Tribunal de Contas da União
UE	União Europeia
UFRGS	Universidade Federal do Rio Grande do Sul
e-CNY	Yuan Digital

SUMÁRIO

1 INTRODUÇÃO	11
2 BACKGROUND.....	15
2.1 <i>Hashing</i>	15
2.2 Criptografia público-privada	16
2.3 Transações	16
2.4 Bloco e <i>blockchain</i>	17
2.5 Mineração em redes <i>blockchains</i>	18
2.6 Protocolos de consenso	19
2.6.1 Problema dos Generais Bizantinos (PGB)	19
2.6.2 Prova de Trabalho (PDT)	20
2.6.3 Prova de Participação (PDP)	20
2.6.4 Prova de Autoridade (PDA)	20
2.7 Criptomoedas	21
2.8 <i>Ethereum</i>	21
2.9 <i>Hyperledger</i>	21
2.10 Tipos de livros razão distribuídos.....	23
2.10.1 Pública Permissionada	24
2.10.2 Pública Não-Permissionada	24
2.10.3 Privada Permissionada	24
2.10.4 Privada Não-Permissionada	24
2.11 Contratos inteligentes	25
2.11.1 Benefícios da implantação e uso de contratos inteligentes	26
2.11.2 Barreiras para a implantação e uso de contratos inteligentes	27
2.11.3 Questões legais sobre a implantação de contratos inteligentes.....	27
2.12 <i>Gas</i>	28
3 REVISÃO DA LITERATURA.....	29
3.1 Uso de <i>Blockchain</i> por Governos	29
3.1.1 Registro de Propriedades	29
3.1.2 Moeda Digital	31
3.1.3 Eleições	32
3.1.4 Identidade Digital Descentralizada	34
3.2 Discussão	35
4 DESENVOLVIMENTO.....	37
4.1 Criação da Rede <i>Blockchain</i> Brasil	37
4.2 Modelo e arquitetura da rede <i>blockchain</i> da RBB	38
4.3 Participantes da RBB	39
4.4 Topologia da RBB	40
4.5 Explorador de Blocos da RBB	41
4.6 Contratos inteligentes de permissionamento da RBB	44
5 ANÁLISE DE CONTRATOS E DISCUSSÃO	46
5.1 Análise do Contrato Inteligente <i>AccountIngress</i>	46
5.2 Análise do contrato <i>NodeIngress</i>	48
5.3 Análise dos Módulos Importados	51
5.3.1 <i>AccountRulesProxy</i>	51
5.3.2 <i>AdminProxy</i>	51
5.3.3 <i>Ingress</i>	52
5.3.4 <i>NodeRulesProxy</i>	54
5.4 Validação dos Contratos	55

5.5 Discussão	60
5.6 Dificuldades - Implantação do Nó Observador	61
6 CONCLUSÃO E TRABALHOS FUTUROS	64
REFERÊNCIAS.....	66

1 INTRODUÇÃO

Em agosto de 2008, o nome de domínio `bitcoin.org` foi oficialmente registrado (Internet Corporation for Assigned Names and Numbers (ICANN), 2025). Pouco tempo depois, em outubro do mesmo ano, o artigo intitulado *Bitcoin: A Peer-to-Peer Electronic Cash System* foi publicado (NAKAMOTO,). O artigo apresenta uma nova tecnologia de dinheiro digital utilizando prova de trabalho, a diferença, porém, era a maneira altamente distribuída, criando um livro-razão distribuído, e o fato de não haver necessidade de confiança entre os participantes da rede, dessa forma, solucionando o problema, nunca antes solucionado, de duplo gasto. No início do ano seguinte, em janeiro de 2009, Satoshi Nakamoto - pseudônimo da, até hoje não conhecida, mente por trás da criação do *Bitcoin* - deu o primeiro passo para a criação da rede, minerando o nó gênese (bloco de número 0) (Blockchain.com, 2025). Porém, foi apenas em 2010 que o *Bitcoin* cumpriu seu papel de moeda digital pela primeira vez: duas pizzas foram compradas usando 10.000 unidades dessa nova moeda digital. O dia ficou conhecido como *Pizza Day* (EXAME, 2023), e hoje 10.000 *Bitcoins* equivalem a mais de cinco trilhões de reais (Binance, 2025).

Embora a criação do *Bitcoin* tenha sido uma revolução para a área de dinheiro digital, ela ainda possuía uma estrutura inflexível causando limitações à rede, por exemplo, permitindo apenas a execução de programas não *Turing*-completos. Por conta disso, após o surgimento do *Bitcoin*, várias outras redes foram criadas, de forma a explorar outras propriedades importantes para o uso efetivo do dinheiro digital. Essas redes *blockchain* ficaram conhecidas como *altcoins* - *alt* vem do inglês *alternative* -, ou seja, qualquer rede que não seja a rede do *Bitcoin*, a original, é uma rede alternativa. A mais conhecida delas é a rede *Ethereum* (WOOD, 2018a), lançada em 2015, com o objetivo de expandir as possibilidades de uso para *blockchain* para além de criptomoedas. Assim como a rede original, a *Ethereum* também é uma rede altamente distribuída, que utiliza prova de trabalho e opera sem a necessidade de confiança entre os participantes; a diferença fundamental e principal contribuição dessa nova *blockchain* é a criação dos chamados contratos inteligentes. Esses contratos nada mais são que trechos de código que são armazenados na rede, de forma permanente e imutável, capazes de atender a pedidos dos usuários. Inicialmente, os contratos eram escritos na linguagem de programação *Solidity*, linguagem estaticamente tipada, baseada em *JavaScript* e, o mais importante, *Turing* completa, ou seja, é teoricamente possível representar qualquer problema computável através de contratos

inteligentes (Ethereum Foundation, 2025b) e publicá-los na rede. Para isso ser possível, foi criada a *Ethereum Virtual Machine* (EVM) (WOOD, 2018b), um ambiente virtual descentralizado capaz de executar código de maneira consistente e segura em todos os nós da rede, assim, a *Ethereum* é classificada como uma máquina de estados distribuída, ao invés de um livro razão distribuído como é o *Bitcoin*. Porém, nós que utilizam a EVM enfrentam um grande problema ao validar transações: quantificar a quantidade de recursos que serão necessários para tal, abrindo caminho para possíveis ataques de negação de serviço. Para contornar esse problema, a *Ethereum* criou o conceito de gás (Ethereum Foundation, 2025a), uma unidade que mede o esforço computacional requerido para executar uma operação em específico na rede; toda transação possui uma taxa de gás que é paga usando a criptomoeda nativa da rede, o Ether. Hoje, a EVM é capaz de compilar o código de contratos inteligentes em EVM *opcodes*, tornando possível a criação desses contratos na maioria das linguagens de programação modernas, cada vez mais popularizando a rede e fazendo com que o *Ether* seja o maior *altcoin*, e a segunda maior criptomoeda, atrás apenas do *Bitcoin*.

Após a criação da rede *Ethereum* e dos contratos inteligentes ampliarem os usos de *blockchain*, diversas novas aplicações surgiram, em várias áreas que podem se beneficiar do uso de uma tecnologia altamente descentralizada, como, por exemplo, na área da saúde e registros médicos, certificados acadêmicos, armazenamento em nuvem (). Com isso, governos estão explorando tal tecnologia para usos desde criptomoedas nacionais oficiais (InfoMoney, 2023) a votações eletrônicas em eleições presidenciais (DUAN; WANG et al., 2024). Além do uso para casos específicos, alguns governos estão apostando no paradigma de *Blockchain-as-a-Service* (*BaaS*), como é o caso da União Europeia (UE), que está desenvolvendo a *European Blockchain Services Infrastructure* (*EBSI*) (European Commission, 2025), uma iniciativa da UE que busca se aproveitar dos benefícios da *blockchain* para o aprimoramento de serviços públicos, em prol da população. O uso da EBSI é dividido em *famílias de caso de uso*, que são áreas de alto valor agregado para a sociedade em que a tecnologia da *blockchain* pode vir a desempenhar um papel chave para resolver problemas relevantes em setores dentro dessas áreas.

Mesmo fora do espectro governamental, a tecnologia *blockchain* ainda pode ter causado um alto impacto positivo na vida dos cidadãos de um país, ou mesmo de um continente, como é o caso da *LACChain* (Ripio, 2024), uma rede *blockchain* latino americana criada por um conglomerado de empresas, mas com o objetivo de servir à população com a ideia de ser uma *blockchain* totalmente orientada a casos de uso que satisfaçam as ne-

cessidades específicas de pessoas e instituições da América Latina, como, por exemplo, transações locais de *stablecoins* (Coinbase, 2025), reserva de valor, pagamentos individuais, liquidação de transações internacionais e remessas, e sistemas de identidade descentralizada (DID). A *LaChain* possui sua própria moeda, o *LaCoin* (LAC), uma moeda não inflacionária (CoinDesk, 2025) utilizada pelos usuários para pagar taxas de transação dentro da rede. Hoje, vários governos latino americanos já estão adotando o uso da *LaChain* como base para seus sistemas governamentais baseados em tecnologia *blockchain* (BLOCKCHAIN, 2025b).

O objetivo deste trabalho gira em torno da implantação de um nó observador da Rede *Blockchain* Brasil (RBB), diretamente na rede do Instituto de Informática (INF) da Universidade Federal do Rio Grande do Sul (UFRGS). A partir deste nó, será possível monitorar a rede, visualizando as transações que são realizadas, bem como os autores dessas operações e o objetivo de cada uma delas. Além disso, os principais contratos inteligentes usados para a criação e manutenção da rede serão tecnicamente avaliados, na tentativa de demonstrar o uso e importância desses contratos. Assim, demonstrando a auditabilidade da rede *blockchain* em questão, de forma a aumentar a transparência do uso dessa nova tecnologia por parte do governo.

A importância desse estudo está na expansão do conhecimento acerca da tecnologia da *blockchain*, dentro e fora do meio acadêmico. Ao fazer a implantação de um nó observador na rede do INF, abrimos caminho para novas pesquisas e futuros trabalhos na área que possam se aproveitar dos dados armazenados e ações realizadas na RBB, que podem ser obtidos utilizando o nó. E ao fazer a análise técnica dos principais contratos inteligentes armazenados na rede, demonstra-se as novas possibilidades que essa tecnologia traz para a computação. Assim, espera-se chamar a atenção para uma área ainda pouco explorada pelo instituto, fomentando a pesquisa e desenvolvimento e contribuindo para o estado da arte.

O presente trabalho de conclusão de curso é dividido em seis capítulos. No primeiro, e presente, capítulo, ocorre a introdução do trabalho, com a motivação por traz e a descrição do trabalho. No segundo capítulo, intitulado *Background*, são apresentadas as informações técnicas necessárias para compreender o trabalho desenvolvido. No capítulo seguinte, revisão da literatura, é feita uma análise do estado da arte de trabalhos que se relacionam com o estudo que será feito. No quarto capítulo, é onde será feito o desenvolvimento deste trabalho, explicando o estudo e os problemas que buscamos explicar. Após o desenvolvimento, no penúltimo capítulo, serão expostos os resultados

obtidos, sendo feita uma discussão acerca deles, com uma análise técnica e empírica. Por fim, na última seção, será apresentado um breve sumário das atividades realizadas durante o desenvolvimento deste trabalho, as conclusões que foram tiradas, além de discutir sobre possíveis trabalhos futuros relacionados a este.

2 BACKGROUND

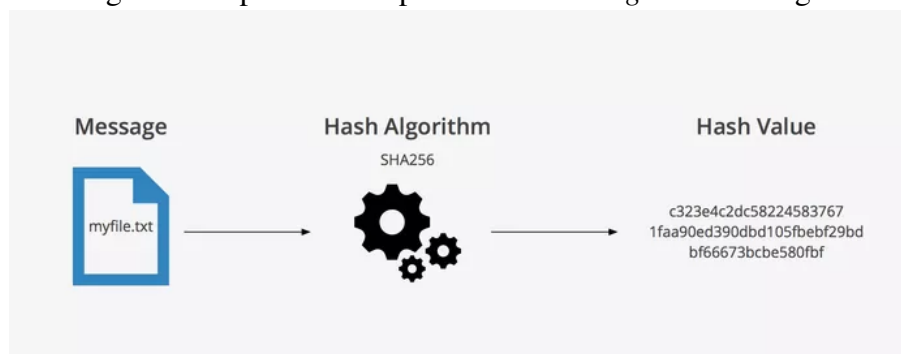
A tecnologia da *blockchain* é uma inovação tecnológica, porém a adoção, em massa, dessa tecnologia é recente. Com isso, muitos de suas propriedades e características não são amplamente conhecidas. Este capítulo apresenta detalhes técnicos sobre os conceitos envolvidos neste trabalho. Esses conceitos são essenciais para entender a tecnologia da *blockchain* e motivar o seu uso. Assim, o conteúdo aqui exposto e discutido é suficiente para o entendimento técnico deste trabalho.

2.1 Hashing

O processo de *hashing* é um processo unidirecional que consiste em utilizar um algoritmo, que recebe de entrada algum tipo de dado, seja ele um documento, um vídeo, uma foto, ou qualquer dado que possa ser interpretado como uma sequência de *bytes*, produzindo, como saída, uma sequência de caracteres de tamanho fixo que representa aquele dado (SWAN, 2015). Essa nova representação não pode ser utilizada em processos de engenharia reversa com o objetivo de obter o dado em sua forma original, por isso é um processo unidirecional. Essa saída é chamada de *hash digest*, e toda vez que a mesma função de *hash* recebe a mesma entrada, a *hash digest* produzida é igual. Sempre que o algoritmo é executado, o resultado de saída gerado é do mesmo tamanho, independentemente do tamanho do arquivo de entrada, ou seja, para um arquivo de 10-GB e um arquivo de 10-KB, a saída é diferente, mas o tamanho da saída gerada é igual.

Um dos algoritmos mais utilizados no processo de *hashing* é o SHA256, um algoritmo da família *Secure Hash Algorithm*, que é capaz de transformar uma entrada de qualquer comprimento em um valor *hash* de tamanho fixo de 256 *bits*, por isso o nome. O SHA256 é dividido em uma série de etapas, sendo elas a preparação da entrada, a configuração inicial, o processamento dos dados em blocos, a função de compactação, para no fim produzir o *hash digest* de 256 *bits* único para aquela entrada. O processo de *hashing* pode ser visto, de maneira simplificado, no diagrama exposto na Figura 2.1.

Figura 2.1: Diagrama simplificado do process de *hashing* usando o algoritmo SHA256



Fonte: (KEYCDN, 2023)

2.2 Criptografia público-privada

Foi criada para resolver o problema da criptografia simétrica, em que era necessário a troca de chaves secretas (SCHEID, 2022). Criptografia assimétrica resolve isso usando um par de chaves pública e privada, associadas através de uma relação matemática, como, por exemplo o algoritmo de assinatura digital com curvas elípticas (AADCE) (JOHNSON; MENEZES; VANSTONE, 2001). O AADCE recebe como entrada o resumo criptográfico (*hash*) da mensagem e produz uma assinatura digital. Ter acesso a chave pública, não permite que o usuário saiba ou faça nada em relação a chave privada pareada a essa chave pública. Por outro lado, possuir acesso a chave privada, permite que informações criptografadas pela chave pública sejam descriptografadas. A relação matemática que associa a chave privada a pública utiliza operações matemáticas com alta complexidade de serem revertidas, chamadas de *trapdoors*, garantindo segurança e confiabilidade no processo.

2.3 Transações

Sempre que um usuário quer se comunicar (por exemplo, realizar uma transferência de valores) com outro dentro da rede, existe a necessidade da criação de uma transação (SCHEID, 2022). Uma transação sinaliza à rede que o usuário remetente autorizou a propagação de seus dados para o usuário destinatário. Para ocorrer a propagação, é necessário, primeiramente, criar um pedido de transação, esse pedido deve especificar todas os requisitos que o fluxo de dados deve seguir para chegar ao receptor. Todo e qualquer pedido de dentro da *blockchain* deve especificar o receptor e é, geralmente, identificado por um identificador único de transação, chamado *nonce*. Deve também ser especificado um

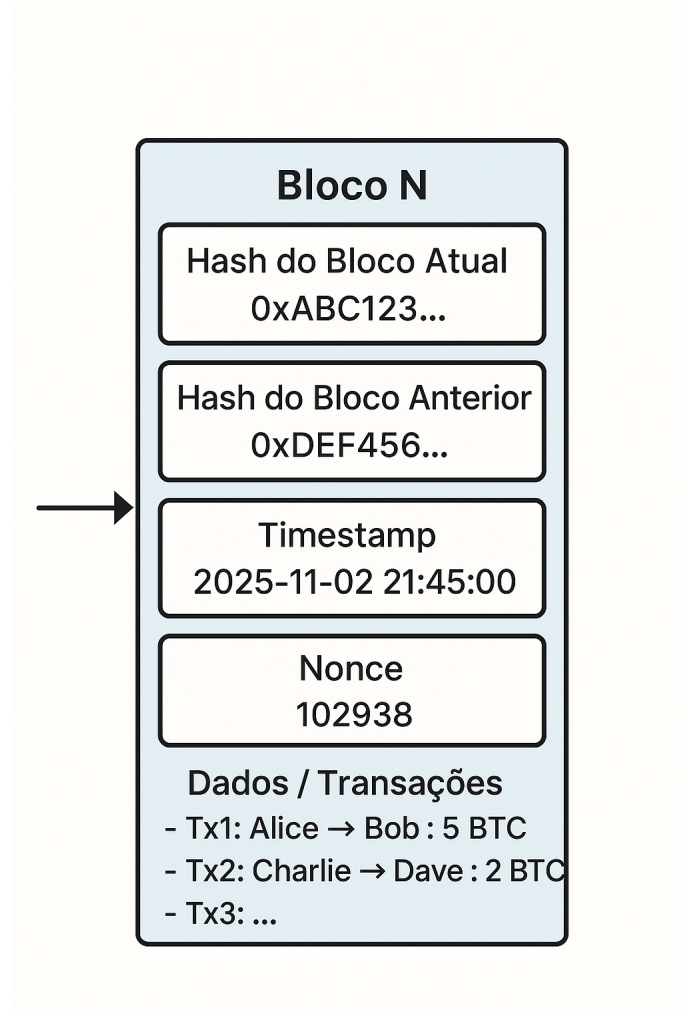
campo dentro do pedido que detalhe a entidade da transferência, por exemplo a quantidade de *tokens* para redes *blockchain* voltadas para moedas digitais. A tecnologia *blockchain* é capaz de suportar a existência de múltiplas fontes e múltiplos receptores, além de que uma transação pode conter mais receptores do que fontes e vice-versa. A proposta de transação deve ser assinada pelo usuário fonte utilizando métodos de criptografia publico-privada provando a posse do endereço que gerou a proposta. Após a assinatura, a transação pode enfim ser propagada pela rede.

Para serem capazes de validar e invalidar transações, todos os nós validadores da rede devem possuir total conhecimento e acesso a toda a informação que deve ser validada. Por conta disso, as transações devem chegar até esses nós o mais cedo possível. Cada nó na rede se conecta apenas com um número limitado de outros nós, que seriam os seus vizinhos. Assim, a propagação de mensagens em uma *blockchain* se assemelha ao algoritmo de busca em largura em um grafo, em que cada nó passa a mensagem para os seus vizinhos, que repassam para os seus vizinhos e assim por diante, até que todos tenham recebido a mensagem (ZARIR et al., 2020).

2.4 Bloco e *blockchain*

Um bloco é uma estrutura de dados que guarda informações da rede *blockchain* (SCHEID, 2022), a Figura 2.2 representa de maneira gráfica um bloco que guarda informações de transações e valores da rede. Cada bloco possui um identificador único e armazena o seu valor de *hash*, o valor de *hash* do bloco anterior na rede e uma série de transações. O bloco gênese (primeiro bloco da rede) é uma exceção, ele não armazena transações, nem tem o valor de *hash* do bloco anterior, isso acontece por ele ser o primeiro bloco da rede. A partir do valor de *hash* do bloco anterior, os blocos são interligados de maneira sequencial, podendo ser visto, de maneira abstrata, como uma lista encadeada, em que cada bloco é um nó e o valor de *hash* é a ligação entre eles. Assim, é criada a *blockchain*, um livro razão distribuído, que armazena, de maneira pública e imutável, blocos que contém transações, através de uma rede de computadores, utilizando um protocolo de consenso, a Figura 2.3 representa uma rede simples com quatro blocos, que guardam informações de transações e valores, encadeados.

Figura 2.2: Diagrama simplificado da estrutura de um bloco

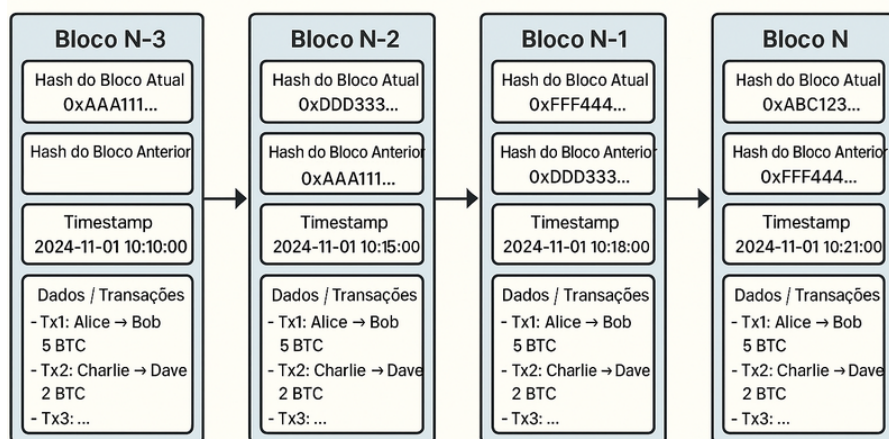


Fonte: (O Autor, 2025)

2.5 Mineração em redes *blockchains*

A mineração em uma rede *blockchain* pode consistir em validar transações, adicionar um bloco à rede, resolver um problema de criptografia e criar novas moedas são sinônimos (BELOTTI et al., 2019). A mineração é feita por blocos dedicados, mas não necessariamente pré-determinados em uma rede *blockchain*. Esses nodos são os chamados mineradores, que guardam todas as transações dessa rede, desde o bloco gênese (o primeiro bloco da rede) até o bloco mais recente adicionado. É necessário que os mineradores retenham essa informação, com o objetivo de verificar a validade de novas transações adicionadas à rede, isso leva a uma solução totalmente descentralizada. O nó minerador deve resolver um problema matemático de criptografia, ao encontrar o valor de *hash* que satisfaz o problema, deve enviar os dados (o bloco, o valor de *hash* e o *nonce* da operação) em *broadcast* para a rede *blockchain*, assim outros nós podem verificar a vali-

Figura 2.3: Diagrama simplificado da estrutura de uma rede *blockchain*



Fonte: (O Autor, 2025)

dade do valor encontrado. Se o valor for válido, é criada uma nova transação na rede que contém a recompensa para aquele nó. Essa recompensa é paga a um minerador, usando a moeda da rede, sempre que um bloco verificado por ele for adicionado à rede.

2.6 Protocolos de consenso

2.6.1 Problema dos Generais Bizantinos (PGB)

Sistemas altamente distribuídos são suscetíveis a falhas em algum de seus componentes, mas para serem considerados sistemas confiáveis, devem ser capazes de lidar com esses problemas e seguir operando corretamente quando ocorrerem. Essa situação pode ser ilustrada pela alegoria do *PGB*. Um grupo de generais do exército bizantino está acampado junto a suas tropas ao redor de uma cidade inimiga. Se comunicando apenas por mensageiros, os generais devem decidir o plano de batalha. Porém, um ou mais deles podem ser traidores com a intenção de confundir os outros. A solução para o problema é encontrar um algoritmo que garanta que os generais que não são traidores cheguem a um veredito (LAMPORT; SHOSTAK; PEASE, 2019). Em uma *blockchain*, esse problema pode ocorrer quando mais de um bloco é validado aproximadamente ao mesmo tempo, ou a latência da rede é alta, resultando em duas redes válidas, com o mesmo número de blocos. Para que a rede possa ser considerada tolerante a falhas Bizantinas, é necessário um número mínimo de quatro nós validadores, já que se um nó agir de forma maliciosa, os outros três conseguem chegar a um consenso independentemente desse nó. Por esse mesmo motivo, é necessário que todos os blocos sejam assinados pela maioria dos nós

validadores, ou seja, no mínimo $\frac{2}{3}$ dos nós validadores. Paralelamente, caso mais de $\frac{1}{3}$ dos nós validadores se tornem indisponíveis, a rede irá parar de produzir transações e adicionar novos blocos a rede. Porém, quanto maior o número de nós validadores que a rede possui, maior será o tempo para que um novo bloco seja aprovado e adicionado à rede.

2.6.2 Prova de Trabalho (PDT)

Baseada em tarefas complicadas de serem executadas, mas triviais de serem verificadas (BELOTTI et al., 2019). O procedimento geralmente é computacionalmente custoso, geralmente sendo necessário o uso de resolução por força bruta. Os nodos da rede competem entre si para encontrar a solução para o problema. A esses nós competidores é dado o nome de mineradores. Quando um minerador encontra uma solução válida, ele exerce o papel de líder, recebendo o direito de adicionar o novo bloco na rede e recebendo a recompensa. Para resolver o *PGB*, é usada a regra da rede mais longa, ou seja, a rede que possui a maior quantidade de blocos, é considerada a rede válida.

2.6.3 Prova de Participação (PDP)

A eleição do líder é baseada no investimento do usuário na rede, ou seja, na quantidade de *tokens* virtuais dessa *blockchain* que o usuário possui (BELOTTI et al., 2019). A probabilidade de se tornar o líder, e ter o direito de adicionar a transação à rede, é proporcional ao investimento, quanto mais *tokens*, maior a chance de poder adicionar um bloco, e vice-versa. Para solucionar o *PGB*, a *PDP* usa a política de rede mais rica, a rede com mais investimento acumulado é considerada a rede válida.

2.6.4 Prova de Autoridade (PDA)

Os nós que podem validar transações são pré-determinados, de forma que apenas esses nós tem permissão de escrita na rede (BELOTTI et al., 2019). Nesse protocolo, o processo de validação de transações ocorre em rodadas, de modo que, em cada rodada, um nó tem a chance de adicionar um novo bloco à rede. As rodadas ocorrem em uma modalidade *round-robin*, ou seja, os validadores são organizados em uma fila e após a sua rodada, vão para o final dessa fila para esperar a sua próxima oportunidade de validar

uma transação. A lista de nós validadores está incluída na bloco gênese, porém, algumas *blockchains* modernas implementam a funcionalidade de eleição, assim, é possível que um nó validador possa ser adicionado ou removido da lista se assim for decidido pela maioria do grupo.

2.7 Criptomoedas

Uma criptomoeda é uma moeda digital baseada no princípio da descentralização, implementada através de uma rede de computadores interligados (BELOTTI et al., 2019). Dessa forma, essas moedas não são vinculadas a nenhuma autoridade controladora central, como bancos ou governos, assim como acontece com as moedas tradicionais. A posse individual de criptomoedas de um usuário é armazenada em uma rede *blockchain*, e redes *blockchains*, geralmente, possuem sua própria moeda nativa, usada para o pagamento de taxas e recompensas, por Ethereum possui *Ether*.

2.8 Ethereum

Ethereum é uma rede *blockchain* que busca combinar e melhorar os conceitos de *scripting*, criptomoedas e meta-protocolos que rodam na rede, permitindo que desenvolvedores criem aplicações baseadas em consenso que tenham escalabilidade, padronização, completude, facilidade de desenvolvimento e interoperabilidade oferecida por esses paradigmas (BUTERIN, 2013). A rede *Ethereum* atingiu esse objetivo ao combinar a tecnologia da *blockchain* com uma linguagem Turing completa embutida, *Solidity*, que permite a criação de contratos inteligentes e aplicações descentralizadas. Dessa forma, diferentemente da rede *blockchain* do *Bitcoin* que se caracteriza como um livro razão distribuído, a *Ethereum* pode ser melhor representada, de maneira abstrata, como uma máquina de estados distribuída.

2.9 Hyperledger

O *Hyperledger* é um projeto que engloba vários setores da indústria de tecnologia, adota o princípio de código fonte aberto e é gerenciado pela *Linux Foundation*, mas possui uma comunidade de organizações mantenedoras de quase 300 organizações (BELOTTI

et al., 2019). O *Hyperledger* é composto por diversos projetos, sendo divididos principalmente em livros razão distribuídos e módulos de suporte. Entidades que desejam fazer parte da rede, devem ser autorizadas e autenticadas, fazendo com que o uso dessa tecnologia seja voltado para aplicações *blockchain* permissionadas. O *Hyperledger* possui uma arquitetura que pode ser dividida em componentes. Com isso, é possível identificar três camadas principais, a camada de consenso, em que os blocos de transações são verificados e validados, a camada de contratos inteligentes, onde acontece o processamento de transações, e a camada de comunicação, que é responsável pelo transporte ponta a ponta das informações na rede. Além dessas camadas, existem outros componentes com funções específicas, como as abstrações de dados e *criptomoedas*, responsáveis, respectivamente, por armazenamento de diferentes tipos de dados e algoritmos de *criptomoedas*. Por fim, o *Hyperledger* oferece serviços importantes, como um serviço de identidade, que possibilita o estabelecimento de uma raiz de confiança durante a implantação de uma instância de rede *blockchain*. Assim, a performance de um sistema baseado em *Hyperledger* excede, significativamente, a performance de redes tradicionais baseadas em PDT, ao ponto de não ser necessário o uso de PDT quando usado o *Hyperledger*.

Besu é um dos projetos que compõem o *Hyperledger* (SCHEID, 2022). *Besu* é um cliente, de código aberto, da rede *blockchain Ethereum*. O projeto tem como objetivo ser amigável ao uso corporativo e compatível com redes *Ethereum* já existentes, sendo possível utilizá-lo em redes públicas ou privadas. O *Hyperledger Besu* possui diversos parâmetros de rede que podem ser configurados, como, por exemplo, tamanho de bloco, tempo do bloco, poder de computação dos nós, complexidade de transação balanceamento de carga. (FAN et al., 2022) fez uma análise desses parâmetros utilizando o *Hyperledger Caliper*, uma ferramenta de *benchmark* de *blockchain*, que permite que usuários testem diferentes aplicações com casos de uso pré-determinados, e obtenham um conjunto de resultado da performance dos testes (Hyperledger Caliper Community, 2025). De acordo com a análise feita, foi concluído que os parâmetros de tamanho de bloco e tempo de bloco são os mais importantes para determinar a performance. Por outro lado, gargalos de performance são causados por execuções de transação em mudanças de estado na rede, que são definidos a partir dos parâmetros de poder de computação de nós, complexidade de transação e balanceamento de carga. Além disso, o estudo analisou algoritmos de prova de trabalho, sendo o mais efetivo deles o protocolo de consenso QBFT, de forma que uma rede do *Hyperledger Besu* que utiliza esse protocolo, pode escalar até 14 nós validadores sem perda de performance perceptível.

Cada nó em uma rede *blockchain* possui um par de chaves pública e privada e um endereço. O *Hyperledger Besu* usa esse par de chaves para assinar e validar transações, e o endereço como um identificador para o nó.

Para identificar um nó na rede, é usado o que é chamado de *Enode URL*. Esse endereço possui o formato *enode://<id>@<host:port>[?discport=<port>]*. Em que, *<id>* é a chave pública do nó, dispensando o *0x* do início, *<host:port>* é o endereço e porta que o nó usa para escutar por novas conexões.

2.10 Tipos de livros razão distribuídos

Embora o uso do termo *blockchain* seja amplo, existem diversas classificações e grupos divididos com base em suas características de permissão (SCHEID et al., 2021). A *blockchain* provê uma tecnologia de armazenamento de dados confiável, descentralizada e disponível publicamente. Porém, para que uma aplicação distribuída possa ser caracterizada como uma rede *blockchain*, é necessário que essa aplicação seja totalmente pública no que diz respeito a leitura, escrita e participação no protocolo de consenso. Assim, se alguma dessas ações for restrita a algum grupo específico de participantes, essa aplicação distribuída é melhor caracterizada como um livro razão distribuído, ao invés de uma rede *blockchain*. Por conta disso, livros razão distribuídos podem ser divididos em grupos diferentes, com base nas suas permissões de escrita e leitura. A Figura 2.4 demonstra de forma gráfica os tipos de livro razão distribuídos com base nas diferentes permissões de escrita e leitura do livro.

Figura 2.4: Tipos de livros razão distribuídos

		Pública			
Permissionada	Pública Permissionada Registro de dados público com permissão de leitura liberada, mas permissão de escrita restrita	Pública Não-permissionada Registro totalmente público com permissão de leitura e permissão de escrita aberta		Privada	Não-permissionada
	Privada Permissionada Registro de dados no modelo de consórcio com permissão de leitura e permissão de escrita restrita	Privada Não-permissionada Registro de dados no modelo de consórcio com permissão de leitura aberta, mas permissão de escrita restrita			
		Privada			

2.10.1 Pública Permissionada

Nessa modalidade, a permissão de escrita é restrita a um grupo seletivo de nós participantes na rede. Porém, a permissão de leitura é liberada, ou seja, qualquer um pode ler as informações que estão armazenadas na rede.

2.10.2 Pública Não-Permissionada

Nesse tipo, as três propriedades são satisfeitas, ou seja, a permissão de escrita, a permissão de leitura e a participação no protocolo de consenso é livre, de forma que qualquer usuário com acesso à internet pode participar da rede. Assim, uma aplicação distribuída pública não-permissionada pode ser considerada uma *blockchain*.

2.10.3 Privada Permissionada

É semelhante à implementação de tecnologias de bancos de dados tradicionais. A permissão de escrita e a permissão de leitura são restritas, de forma que apenas entidades autorizadas têm acesso aos dados armazenados. Ao restringir essas permissões a apenas algumas entidades, características como transparência, imutabilidade e descentralização podem ser prejudicadas. Por conta disso, uma aplicação privada permissionada não pode ser considerada uma rede *blockchain*.

2.10.4 Privada Não-Permissionada

Se assemelha a implementação pública não-permissionada, porém, a permissão de escrita e a permissão de leitura é restrita a apenas um grupo seletivo de nós participantes da rede. Esse grupo deve ser pré-determinado. Assim, apenas os nós que fazem parte desse grupo privado podem ler os dados armazenados na rede, e podem escrever dados na rede, mas qualquer membro da rede pode criar transações, sem limitação alguma.

2.11 Contratos inteligentes

Um contrato inteligente é um protocolo usado para propagar os termos de um acordo feito de maneira digital. Com o uso desse contratos, usuários desconhecidos e, possivelmente, não confiáveis, podem realizar transações entre si sem a necessidade de nenhum intermediário confiável, pois eles não são controlados por nenhuma autoridade central, nem permitem intervenção humana (MOHANTY, 2018). Contratos inteligentes são formados por uma coleção de algoritmos auto-executáveis, auto-verificáveis, e resistentes a modificação. Os termos do contrato são descritos em código, que encapsula e replica a lógica de um contrato legal real, porém, digitalmente. Em redes *blockchain* que implementam e permitem o uso de contratos inteligentes, é possível completar uma transação mais rapidamente, levando a uma execução quase em tempo real, resultando em um menor custo por transação e, consequentemente, aumentando a segurança da rede (TAHERDOOST, 2023). Sempre que um contrato inteligente é usado em uma transação, o código é executado automaticamente e o resultado é propagado por toda a rede *blockchain* à qual o contrato inteligente faz parte. O *Listing 2.1* é um exemplo básico de um contrato inteligente escrito em Solidity. Na linha 1, é definido a versão do compilador que deve ser usado para gerar o contrato inteligente. Na linha 3, é declarado um contrato, chamado *SimpleCounter*. Na linhas 4 e 5 são declaradas variáveis que serão usadas no contrato, a primeira guarda um valor numérico, que será usado como contador, a segunda guarda o endereço do usuário que instanciou o contrato. Como são variáveis de estado, ficam armazenadas na *blockchain*. O modificador *public* torna possível lê-las externamente via função gerada automaticamente pelo compilador. Na linha 6, temos o construtor do contrato, uma função que é chamada automaticamente ao se criar uma nova instância desse contrato. Dentro do construtor, na linha 7, temos a atribuição de um valor à variável *owner*, que recebe o emissor da mensagem. Na linha 10, temos a definição de uma função pública, ou seja, qualquer usuário pode tentar utilizá-la, porém, na linha 11, há o uso da função *require*, que requer que o usuário que chamou a função seja o dono para que a execução continue. Caso o usuário seja o dono, na linha 12, temos o incremento da variável de contador do contrato.

Listing 2.1: Exemplo de contrato inteligente em Solidity: contador simples

```

1 pragma solidity ^0.8.20;
2
3 contract SimpleCounter {
4     uint256 public count;

```

```
5  address public owner;
6  constructor() {
7      owner = msg.sender;
8  }
9
10 function increment() public {
11     require(msg.sender == owner, "Apenas o dono pode aumentar");
12     count++;
13 }
14 }
```

Fonte: O Autor (2025)

2.11.1 Benefícios da implantação e uso de contratos inteligentes

Ao eliminar o uso de agentes intermediários, não há necessidade de adição de custo financeiro nem tempo de espera para concluir transações na rede *blockchain*. Como a execução de um contrato é automática, e o resultado deve ser propagado para toda a rede, para que agentes maliciosos consigam fazer alguma modificação na rede, seria necessário que essa mudança fosse propagada por toda a rede, adicionando uma camada de proteção à rede. Além da economia e segurança que os contratos inteligentes oferecem, o seu uso também possibilita a automação de processos, antes manuais, em diversos setores, inclusive governamentais. Podendo, assim, proporcionar o acesso a dados e serviços por demanda (TAHERDOOST, 2023). A Tabela 2.1 sumariza como a adoção de contratos inteligentes pode beneficiar uma rede *blockchain*.

Tabela 2.1: Vantagens da implantação e uso de contratos inteligentes

Vantagem	Motivo
Economia financeira	Ao eliminar intermediários, não há necessidade de custos adicionais nem tempo de espera para concluir transações.
Segurança	Para que agentes maliciosos consigam modificar dados na rede, seria necessário que a alteração fosse propagada e validada por todos os nós participantes.
Eficiência	Assim que as condições pré-determinadas são atingidas, o contrato inteligente é automaticamente executado.
Disponibilidade	Possibilita automação de processos, oferecendo acesso a dados e serviços sob demanda.

Fonte: (O Autor, 2025)

2.11.2 Barreiras para a implantação e uso de contratos inteligentes

Redes *blockchain* podem ser vistas como uma rede de banco de dados distribuída, em que cada nodo possui todos os dados armazenados, de forma que o poder de processamento da rede inteira pode ser limitado por apenas um nodo. Como os contratos inteligentes são executados sequencialmente, por consequência, pode haver um aumento significativo da carga total de processamento na rede, podendo causar atrasos e altas latências de processamento. Por conta do possível problema com o poder computacional da rede, contratos inteligentes não escalam bem o seu uso de memória na *blockchain*. Cada bloco possui um tamanho pré-determinado imutável, de forma que, em caso de erros de registro de dados, é necessário acrescentar o registro corrigido no fim da cadeia de blocos. Essa característica contribui para a alta descentralização da tecnologia, porém restringe a escalabilidade de armazenamento nos blocos. Por conta disso, atualmente, não existem maneiras eficazes de prover escalabilidade para o armazenamento de contratos inteligentes em redes *blockchain*, o que pode ser um problema se levado o alto número de transações de redes modernas (TAHERDOOST, 2023). A Tabela 2.2 sumariza os problemas que podem levar a barreiras na implantação de contratos inteligentes em uma rede *blockchain*.

2.11.3 Questões legais sobre a implantação de contratos inteligentes

A rede *Ethereum* introduziu o conceito de contratos inteligentes com a ideia de *código é lei* (BUTERIN, 2013). Por conta disso, diversas implicações legais recaem sobre a implantação e utilização desses contratos. Até mesmo a imutabilidade se torna um problema, visto que não uma maneira eficaz de detectar vulnerabilidades em contratos. A partir do momento que um contrato foi implementado na rede, não é mais possível alterá-lo, ou seja, para corrigir *bugs* introduzidos na fase de desenvolvimento é uma tarefa quase impossível. A principal maneira de corrigir esses problemas é criar um novo contrato e enviá-lo à rede, o que vai contra a ideia de *código é lei* introduzido pela *Ethereum*. Além disso, atualmente, os contratos inteligentes não possuem valor legal algum, sendo vistos apenas como simples programas de computador, de forma que não existe um consenso sobre a regulação do uso dessa tecnologia (TAHERDOOST, 2023).

Tabela 2.2: Barreiras e limitações dos contratos inteligentes

Problema	Motivo
Poder de processamento	Contratos inteligentes são executados sequencialmente na rede, consequentemente aumentando a carga total de processamento da rede <i>blockchain</i> inteira.
Armazenamento	Cada bloco na rede <i>blockchain</i> possui um tamanho mínimo pré-determinado e não é possível apagar blocos, logo o tamanho é sempre crescente.
Imutabilidade	A partir do momento que um contrato inteligente for implementado na rede <i>blockchain</i> , não é mais possível alterá-lo, o que dificulta lidar com problemas de segurança.
Integridade	O acesso aos dados armazenados em contratos inteligentes tende a ser restrito dentro das redes <i>blockchain</i> , exigindo o uso de aplicações terceiras (carteiras).
Segurança	Manter o código de um contrato inteligente após enviá-lo para a rede é extremamente difícil, já que não há uma maneira de corrigir <i>bugs</i> sem reenviar o contrato à <i>blockchain</i> , consumindo tempo e espaço.
Questões legais	Atualmente, contratos inteligentes não têm valor legal, sendo considerados apenas programas de computador.
Privacidade	Contratos inteligentes oferecem pseudônimos aos usuários, mas isso não garante privacidade total.

Fonte: (O Autor, 2025)

2.12 Gas

Gas é a unidade que mede a quantidade de poder computacional necessário para executar uma operação específica na rede *blockchain Ethereum* (BUTERIN, 2013). Todas as transações executada na rede requer recursos computacionais para serem executada, dessa forma, é possível que a rede sofra com problemas de spam ou fique presa em *loops* de execução infinitos. Por isso, esses recursos tem de ser pagos, com o objetivo de minimizar a probabilidade de ocorrência desses problemas. Esse pagamento é chamado de taxa de *gas*, ou seja, a quantia de *gas* usada para fazer alguma operação, multiplicada pelo custo da unidade de *gas*, e que deve ser paga, na moeda nativa da rede, independentemente da transação ter sucesso ou falhar. O próprio usuário pode definir a quantidade de *gas* que está disposto a pagar para enviar uma transação à rede. Quanto menor a quantidade oferecida, menor é a chance dos nós validadores escolherem sua transação para incluída no próximo bloco.

3 REVISÃO DA LITERATURA

Este trabalho é baseado na Rede Brasil *Blockchain*. Ou seja, uma rede que utiliza a tecnologia *blockchain* criada e controlada pelo governo Brasileiro. Assim, no escopo deste trabalho, é necessário entender quais os usos que outros governos ao redor mundo dão a essa tecnologia. Esta seção traz uma revisão da literatura sobre o uso da tecnologia da *blockchain* por governos. Além de expor os resultados da pesquisa, é feita uma discussão sobre os dados encontrados.

3.1 Uso de *Blockchain* por Governos

A partir da criação do *Bitcoin*, em 2009, vimos o primeiro exemplo de uso da tecnologia *blockchain*. Com isso, o uso dessa tecnologia não era mais restrito a um grupo técnico e capacitado. Qualquer um podia utilizar a rede do *Bitcoin* e usufruir dos seus benefícios.

Com a crescente adoção do uso da tecnologia da *blockchain* em diversos segmentos, governos ao redor do mundo começaram a pesquisar o uso e a adoção dessa tecnologia em setores públicos. Aplicações governamentais usadas para registro de propriedades, moedas digitais e até mesmo votações começaram a adotar o uso da *blockchain* com o intuito de trazer os benefícios dessa tecnologia aos cidadãos.

3.1.1 Registro de Propriedades

Segundo a Lei nº 6.015, de 31 de dezembro de 1973 – Lei dos Registros Públicos - Art. 188, o registro de posse de uma propriedade feito em cartório deve ter uma devolutiva em até 30 dias corridos. Em 2022, a Lei 14.382 alterou esse prazo para 30 dias úteis, ou seja, para obter o registro legal de propriedade, o comprador pode ter de esperar até 6 semanas. Então, mesmo após anos de evolução tecnológica e modernização dos processos, o tempo para registrar uma terra ou imóvel aumentou.

Diante disso, o uso da tecnologia de *blockchain* se mostra extremamente atrativo para esse setor (KACZOROWSKA, 2019), podendo trazer inúmeros benefícios, como, por exemplo, armazenar transações de propriedade com segurança na rede, aumento de confiança no processo, além da redução do custo operacional para o governo. A maior

diferença, porém, seria a facilidade de agilidade para realização de uma transferência de propriedade, dado que, com a tecnologia *blockchain*, basta apenas que a transação seja adicionada à rede. Essa ideia, porém, tem o potencial de causar problemas se implementada da forma errada, visto que transações, a partir do momento que adicionadas à rede, são irreversíveis e propagadas sem a intervenção de intermediários, o que leva a uma falta de controle centralizado por parte de órgãos públicos. Hoje, vários países já adotam soluções que integram a *blockchain* no seu serviço de registro de terras e imóveis.

Na Geórgia, a Agência Nacional de Registro Públicos (NAPR), começou a utilizar *blockchain* nos seus serviços de registros de propriedades (LAZUASHVILI; NORTA; DRAHEIM, 2019). O uso, porém, ainda é muito limitado, de forma que a tecnologia funciona apenas como um “apêndice” no já existente sistema de registros. Ou seja, o processo continua sendo feito da mesma forma, apenas usando a *blockchain* como uma plataforma de armazenamento de dados. Além disso, o modelo atual usado pela NAPR ainda apresenta uma grave falha de segurança, os dados dos cidadãos podem ser alterados antes do registro ser guardado, de forma imutável e irreversível, na rede, abrindo espaço para possíveis ataques de engenharia social.

Na Índia, o registro de imóveis e terras é um problema ainda nos dias de hoje. A maneira como esse bens são registrados pode mudar de acordo com o estado, isso causa uma grande insegurança aos habitantes, muitos temem que o vendedor não seja legalmente dono do bem, ou mesmo que ao fazer a compra, não se tornem legalmente donos da propriedade. A partir dessa análise da insegurança do povo, (THAKUR M.N. DOJA, 2020) analisa o problema e sugere uma solução baseada em *blockchain*. O artigo destaca que ao aplicar a tecnologia para registro de imóveis e terras, seria possível aproveitar os seus diversos benefícios, como, por exemplo, possuir um histórico de transações imutável, permanentemente armazenados na rede, de forma que fosse impossível que esses dados fossem alterados ou forjados, de forma a reduzir a insegurança da população ao adquirir um desses bens. Além disso, o artigo destaca o papel importante que a adoção da *blockchain* para esse cenário pode ter para atingir metas de sustentabilidade do país, além de lidar com desastres naturais vividos no país, como a inundação de 2018 no estado do Kerala, que destruiu inúmeros documentos oficiais, que ainda eram físicos, caso a tecnologia já estivesse em uso, todos os registros estariam armazenados digitalmente na rede e não teriam sido perdidos.

3.1.2 Moeda Digital

Após o surgimento do *Bitcoin*, em 2009, a popularidade de criptomoedas vem crescendo exponencialmente, com novas moedas sendo criadas e novas formas de utilização sendo implementadas. Hoje, muitos estabelecimentos comerciais já adotaram criptomoedas como uma de suas formas de pagamento. Segundo (MOFOKENG, 2024), o número de usuários que utiliza esse meio de pagamento cresceu radicalmente, isso advém de uma crescente percepção de confiabilidade que a tecnologia passa a esses usuários, os levando não apenas a utilizá-lo, como também a divulgar, de forma positiva, essa inovação tecnológica.

Em 8 de julho de 2021, a Assembleia Legislativa de El Salvador, aprovou a Lei Bitcoin, lei que dá ao bitcoin status de moeda corrente oficial no país. Desde o ano de 2000, o país já possuía uma economia dolarizada, ou seja, usava o dólar americano. Essa adoção da moeda estrangeira foi tomada durante um período de grave crise econômica nacional, trazendo estabilidade monetária ao país, mas ainda deixando lacunas macroeconômicas a serem resolvidas, da maneira semelhante, a adoção do *bitcoin* se deu após graves problemas econômicos causados pela pandemia da COVID-19. Segundo (ALONSO et al., 2024) ainda não é possível medir as consequências da lei, por um lado, há benefícios como, por exemplo, o aumento de transações e a possível valorização do preço da moeda digital, por outro lado, há inúmeros riscos, com destaque para a volatilidade de preço e a facilidade do uso da moeda para transações fora da lei. A medida mais discutível incluída na lei é a obrigatoriedade de aceitação da moeda em transações realizadas no país. Esses problemas demonstram uma possível insegurança na implantação dessa nova moeda corrente, de forma que mais da metade da população salvadoreense, 53,5% segundo a pesquisa (ALONSO; JORGE-VÁZQUEZ; HERNÁNDEZ, 2023).

No mês seguinte do mesmo ano, em setembro de 2021, a China, fez o contrário de El Salvador, tornou ilegal toda e qualquer transação feita com criptomoedas, incluindo até mesmo as mais conhecidas, *Bitcoin* e *Ether* (BROOKE, 2021). A medida foi tomada em conjunto por 10 órgãos governamentais, liderados pelo Banco Popular da China (BPC), e teve como base os potenciais riscos que adotar criptomoedas como moeda corrente traz, como os que El Salvador pode vir a enfrentar. Porém, o governo chinês entende o impacto e a relevância do dinheiro digital, por conta disso, a adoção dessa nova forma de pagamento como moeda corrente se deu de outra forma, através do que é chamado de Moedas Digitais de Banco Central (MDBC), ou seja, o governo possui a sua própria moeda digi-

tal, então, a China criou o Yuan Digital (e-CNY), a versão digital da sua moeda, o Yuan. Segundo (CHEN, 2023), a MDBC chinesa traz várias vantagens, como, por exemplo, segurança institucional, conveniência, taxa de transação gratuita e medidas protetivas contra possíveis usos criminais da moeda. Além disso, os cidadãos chineses também veem benefícios na adoção desse novo meio de pagamento, dado que não é necessário associar os seus e-CNY a uma conta bancária e não há cobrança de juros sobre a moeda. Por fim, a adoção do Yuan Digital, busca facilitar a produtividade baseada em dados, levando a uma digitalização de casos de uso de pagamentos públicos, levando a um ecossistema econômico digital saudável para o país.

No Brasil, transações utilizando criptomoedas tradicionais, como o *Bitcoin* e o *Ether*, são permitidas, mas nenhuma tem o *status* de moeda corrente do país. Semelhante a China, o governo brasileiro pretende criar a sua própria MDBC brasileira, liderados pelo Banco Central (BACEN), órgãos públicos vão criar o Real Digital (DREX). Segundo o BACEN (CENTRAL, 2025), o DREX deverá ter o mesmo valor e a mesma aceitação da do real tradicional, assumindo o *status* de moeda corrente no Brasil. O próprio BACEN será responsável pela emissão e regulação do DREX permitindo que vários tipos de transações financeiras seguras com ativos digitais estejam à disposição da população.

3.1.3 Eleições

Construir um sistema de votação digital e seguro é um desafio enfrentado por todos os governos democráticos ao redor do mundo. Em alguns países, como é o caso dos Estados Unidos da América (EUA), as eleições ainda são feitas utilizando o voto impresso. No Brasil, temos um dos sistemas de votação mais eficientes do mundo, os votos são eletrônicos e os resultados são divulgados horas após o fim do período de votação. Mesmo com um dos sistemas mais modernos e eficientes da atualidade, o governo brasileiro enfrenta problemas com o custo e a logística de levar máquinas físicas para todos os locais de votação, de forma a atender toda a população votante do país. Esses problemas, abrem espaço para o uso da tecnologia da *blockchain* aplicada ao cenário das eleições, que pode aumentar a segurança e diminuir os custos da organização de uma eleição. (HJÁLMARSSON et al., 2018) propõe um sistema de votação eletrônica baseado nessa tecnologia, que busca solucionar esses problemas e avaliar *frameworks* de *blockchain* populares com o propósito de construir esse sistema de votação.

Segundo a Constituição dos EUA (Artigo I, Seção 4), a responsabilidade de con-

dução das eleições recai primariamente sobre os estados. No entanto, existem leis federais como, por exemplo, *Voting Rights Act* (1965) e *Help America Vote Act* (2002) que estabelecem diretrizes nacionais com o intuito de promover acessibilidade à população e integridade à votação. Com base nisso, nas eleições de 2018, o estado da Virgínia Ocidental identificou uma lacuna no seu sistema de votação: militares embarcados que estavam em bases a milhares de quilômetros do país. Para atingir o objetivo da acessibilidade, o estado buscou uma solução baseada na tecnologia da *blockchain*, criando um aplicativo móvel chamado *Voatz* que, segundo a empresa criadora, mantém a segurança e integridade da eleição é preservada através de uma rede *blockchain* permissionada. Apesar da pouca documentação acerca do projeto, (SPECTER; KOPPEL; WEITZNER, 2020) busca, utilizando técnicas de engenharia reversa, analisar o nível de segurança que essa aplicação possui e proporciona ao voto do cidadão. Segundo os autores, o aplicativo apresenta falhas graves de segurança, de forma que um adversário passivo de rede pode descobrir o voto de um usuário, e um adversário ativo pode interromper a transmissão em resposta. Além disso, foram encontradas evidências que sugerem que um usuário com acesso ao servidor teria poder de observar, alterar e adicionar votos. Assim, a conclusão dos autores é que o experimento da Virgínia Ocidental falhou, e que o aplicativo não deve ser considerado para usos futuros. Eles finalizam a análise ao pedir que futuras soluções de votação eletrônica baseadas em *blockchain* tenham ampla documentação, de forma a garantir a segurança e integridade das votações, fazendo com que o uso dessa tecnologia em eleições seja viável.

Em Serra Leoa, também nas eleições de 2018, a tecnologia da *blockchain* também foi utilizada. Diferentemente do uso para votação, tal qual nos EUA, o governo utilizou a *blockchain* para a verificação dos resultados. O país vinha sofrendo há muito tempo com a falta de segurança nas suas eleições, fazendo com que os cidadãos muitas vezes contestassem os resultados. Afim de aumentar a confiança no seu processo de eleição, Serra Leoa usou os serviços da empresa suíça Agora, especializada em *blockchain* (KSHETRI; VOAS, 2018). Após o fim da votação, observadores da empresa de diferentes localidades, inseriram, manualmente, mais de quatrocentos mil votos na rede. Segundo a empresa, esse teste deve ser considerado apenas uma implementação parcial de uma rede, visto os votos foram apenas verificados, e não movidos pela tecnologia.

3.1.4 Identidade Digital Descentralizada

Sempre um usuário faz *login* ou cria uma conta em um *site*, essas informações são armazenadas em um grande banco de dados centralizados, controlados pela empresa dona do sistema. Esses dados podem ser pessoais, incluindo nome, data de nascimento, ou até mesmo dados bancários. Essas informações são extremamente valiosas para pessoas mal intencionadas, como *hackers*, que, se tiverem acesso a esse banco de dados centralizado, conseguem ver os dados de todos os usuários e até mesmo usá-los para ataques de engenharia social. Segundo (ALEXANDRA, 2025), o crescente uso de computação em nuvem aumenta ainda mais esses riscos da centralização de dados, visto que agora as empresas tem de confiar em terceiros para armazenar os seus dados, causando um aumento significativo no vazamento de dados de usuários. Esses problemas fazem com que a confiança do consumidor diminua, além de fazer com que as empresas tenham de enfrentar consequências legais, e até mesmo pagar altos valores em multas por conta desses vazamentos.

Por conta do armazenamento centralizado dos dados, esses vazamentos se tornam ainda mais perigosos, uma vez que ao ter acesso aos dados de um usuário, é possível ter acesso ao de todos. Com isso, o uso da tecnologia da *blockchain* se mostra promissor nessa área, sendo possível se beneficiar da alta descentralização que uma rede possui para mitigar esses problemas. A partir dessa ideia, foi criado o conceito de Identidade Digital Descentralizada (IDD) (NAIR, 2025), uma maneira de fazer com que nenhuma entidade controle suas informações pessoais, nem o governo, nem as empresas. O IDD não guarda nenhuma informação pessoal, apenas uma referência para um Documento IDD, um conjunto de metadados que inclui chaves públicas e outras informações verificáveis que podem comprovar a sua identidade. Assim, a *blockchain* se mostra a tecnologia ideal para a implementação de IDs, de forma que a rede pode guardar os Documentos, garantindo que eles são imutáveis e verificáveis.

Na Europa, a UE estima que até 2050, 68% da população irá viver em áreas urbanas (EUROPEIA, 2018). Com esse alto fluxo de pessoas, é esperado que os sistemas e processos governamentais, da forma como são hoje, acabem sendo sobrecarregados. Por conta disso, cada vez mais cidades estão buscando se modernizar e se tornar o que é chamado de "cidade inteligente", uma cidade que se utiliza de tecnologia de ponta para servir a população. A cidade de Zug, na Suíça, se preparando para o aumento da população urbana projetado pela UE, já começou a se adequar ao cenário, se tornando inteligente. Zug

passou a usar um sistema de IDD (YOUNG; VERHULST, 2018), baseado na *blockchain Ethereum*, que permite que seus cidadãos controlem e manipulem seus próprios dados, sem o risco da centralização dessas informações. Além disso, com a implementação da IDD na cidade, é esperado que haja uma economia milionária em custos de pessoas e produtividade para os serviços que a IDD cobre.

3.2 Discussão

A partir dessa análise do estado da arte, apresentada de forma resumida na Tabela 3.1, é possível entender a importância da tecnologia no contexto governamental. Com o crescente número de casos de uso da *blockchain*, como em registros de imóveis e terras (THAKUR M.N. DOJA, 2020) e criptomoedas (CHEN, 2023), e o número de governos que a utilizam em algum cenário, é possível perceber o papel que essa tecnologia pode assumir na transformação digital e modernização de um país, além dos benefícios e facilidades que pode trazer para os cidadãos. Hoje vemos um crescente uso de tecnologias no cotidiano das pessoas, como a *blockchain*, para usos como IDD ou votações, e também outras tecnologias, como a inteligência artificial, também usada em aplicações públicas (SCHMAGER et al., 2023). Com essa crescente adoção de tecnologias, os países estão apostando em uso de tecnologia em setores governamentais, com isso, estamos vendo cada vez mais investimentos públicos em pesquisa e desenvolvimento de tecnologia. Portanto, é esperado que a adoção dessas novas tecnologias, incluindo a *blockchain*, cresça por parte dos governos, beneficiando cada vez mais a população e extraindo o máximo possível dos benefícios desses avanços.

Tabela 3.1: Usos de *blockchain* por governos e problemas relacionados

Artigo	Ano	Uso	Resumo
(KACZOROWSKA, 2019)	2019	Registro de imóveis e terras	Visão geral do assunto
(LAZUASHVILI; NORTA; DRAHEIM, 2019)	2019	Registro de imóveis e terras	Estudo de caso da Geórgia
(THAKUR M.N. DOJA, 2020)	2020	Registro de imóveis e terras	Proposta de uso para a Índia
(MOFOKENG, 2024)	2024	Dinheiro digital	Análise sobre adoção de cripto como meio de pagamento
(ALONSO et al., 2024)	2022	Dinheiro digital	Análise econômica da adoção do Bitcoin em El Salvador
(ALONSO; JORGE-VÁZQUEZ; HERNÁNDEZ, 2023)	2023	Dinheiro digital	Opinião pública sobre uso de Bitcoin
(CHEN, 2023)	2019	Dinheiro digital	Moeda digital chinesa
(BROOKE, 2021)	2019	Dinheiro digital	Banimento de criptos na China
(CENTRAL, 2025)	2025	Dinheiro digital	Explicação do DREX
(HJÁLMARSSON et al., 2018)	2018	Eleição	Proposta de uso de blockchain para votação
(SPECTER; KOPPEL; WEITZNER, 2020)	2020	Eleição	Análise do sistema usado nos EUA em 2018
(ALEXANDRA, 2025)	2025	Vazamento de dados	Análise de vazamentos de dados em ambientes de computação em nuvem
(NAIR, 2025)	2025	Identidade Digital (DID)	Explicação sobre o conceito de DID
(EUROPEIA, 2018)	2025	Identidade Digital (DID)	Reportagem sobre o aumento da população humana
(YOUNG; VERHULST, 2018)	2018	Identidade Digital (DID)	Estudo de caso da Suíça

Fonte: (O Autor, 2025)

4 DESENVOLVIMENTO

Este capítulo apresenta o contexto histórico da criação e propósito da Rede *Blockchain* Brasil (RBB), incluindo detalhes técnicos como sua arquitetura e topologia, e detalhes de governança, como os participantes da rede e os modelos de participação. Além disso, é detalhado como os usuários de fora podem ter acesso aos dados que estão armazenados na rede. Por fim, detalha o funcionamento do permissionamento da rede, que controla o acesso de participantes. Este capítulo é baseado única e exclusivamente na documentação oficial da RBB, que está centralizado em um repositório remoto (BLOCKCHAIN, 2025b).

4.1 Criação da Rede *Blockchain* Brasil

Em 2019, foi notado que havia uma duplicação de esforços no uso de tecnologia por parte do governo. O problema era que, para cada caso de uso, era necessário que a entidade interessada superasse diversas barreiras legais, organizacionais, burocráticas e técnicas. Com isso, em diversos casos, era necessário resolver os mesmos problemas a cada aplicação, impedindo o avanço e inovação tecnológica. Além disso, a existência de diversas redes pode dificultar a integração entre iniciativas distintas, ou mesmo inibe o reuso de soluções entre diferentes organizações.

A Rede *Blockchain* Brasil (RBB) é uma rede *blockchain* governamental foi criada com o intuito de resolver esse problema. A rede, fundada pelo Banco Nacional de Desenvolvimento Econômico (BNDES) e o Tribunal de Contas da União (TCU), é formada por uma série de entidades e instituições, que abrange todo o país. O objetivo da RBB é possuir uma estrutura de governança e infraestrutura tecnológica que facilite a adoção da tecnologia da *blockchain* para aplicações de uso público. A utilização da rede permitirá a otimização de recursos, redução de custos e remoção de barreiras de entrada para o uso da tecnologia e inovação no setor público.

A RBB se inspirou em outras redes *blockchain* governamentais, como a *LAC-Chain*, rede para a América Latina, e a EBSI, rede do tipo BaaS da União Europeia. A partir disso, a RBB buscou seguir o caminho de ser uma rede público-permissionada. A rede é pública, já que qualquer pessoa poderá ler os dados armazenados nela, mas é permissionada porque nós precisamos de permissão para participar do protocolo de consenso. Assim, o tipo de rede público-permissionada se mostrou o mais adequado para uma im-

plementação de uma *blockchain* de uso governamental.

4.2 Modelo e arquitetura da rede *blockchain* da RBB

Blockchains públicas tem um potencial muito interessante para implantações que tem foco em transparência e confiança. Um dos maiores benefícios do uso dessa tecnologia é a possibilidade de implantação de medidas anti-corrupção e pró-transparência. Com isso, seria possível implantar processos transparentes e que integrem a conformidade regulatória e ética diretamente no design e desenvolvimento, ou seja, seguir o *Compliance by Design*. Porém, o uso de tecnologias de *blockchain* públicas impõe diversos desafios no ambiente de uso público, como, por exemplo, a aquisição de criptomoedas para o pagamento de recompensas da rede.

Em redes do tipo permissionada, muitas vezes vezes privadas, os nós responsáveis por validar transações são conhecidos de antemão e autorizados, de acordo com as regras e requisitos da própria rede, de forma que o processamento na rede seja um compromisso dos participantes e que cada um tenha suas próprias responsabilidades. Porém, em redes permissionadas, a permissão de leitura não é aberta, ou seja, usuários de fora da rede não poderiam ler os dados armazenados.

Por conta desses problemas, a RBB escolheu o modelo público-permissionado. Com ele, é possível aproveitar das melhores características de modelo público, principalmente a transparência e confiança que esse modelo propicia. E também, os benefícios de redes do tipo permissionada, como, por exemplo, o menor custo e o menos barreiras tecnológicas e regulatórias para a implantação. Assim, uma rede público-permissionada se torna o modelo ideal para uma aplicação de *blockchain* governamental.

A implementação desse rede no modelo público-permissionado foi feita usando o *Hyperledger Besu*, um cliente para redes *blockchain Ethereum*. Nós em uma rede baseada no *Hyperledger Besu* executam uma EVM, de forma que é possível armazenar e executar contratos inteligentes na rede, escritos em qualquer linguagem de programação que possa gerar instruções que a EVM consiga executar.

A RBB usa como algoritmo de consenso o protocolo QBFT, baseado em prova de autoridade. Nesse protocolo, contas aprovadas atuam como nós validadores, tendo o poder de validar transações e blocos da rede, e apenas esses nós admitidos e permissionados como validadores podem propor ou validar novos blocos. Esse grupo de nós também pode operar em modo de eleição, de forma a propor e votar pela admissão de novos nós

validadores ou também pela exclusão de nós que já fazem parte do grupo. No QBFT, não há bifurcações ou entrada concorrentes de novos blocos na rede, então, todo novo bloco é sempre inserido ao final da cadeia de blocos. Além disso, cada um dos nós desse grupo tem um intervalo de tempo finito para propor um novo bloco, caso esse tempo expire, um novo validador é escolhido e recebe a oportunidade de propor um novo bloco. E como discutido na Seção 2.6.1, a RBB satisfaz os requisitos para ser considerada uma rede *blockchain* tolerante a falhas bizantinas.

4.3 Participantes da RBB

Diversas entidades integram o projeto da Rede *Blockchain* Brasil, incluindo instituições públicas e privadas. Cada participante da rede pertence a um de três grupos, partícipes aderentes **parceiros**, partícipes aderentes **associados** ou partícipes **patronos**. Sendo esses grupos divididos em uma hierarquia com base nas suas permissões dentro da rede. A Tabela 4.1 destaca as instituições que já escreveram um bloco na rede, o grupo ao qual elas pertencem e o seu endereço dentro da RBB.

Partícipes aderentes **parceiros** devem promover o uso da RBB em aplicações do serviço público e de interesse da população, ao mesmo tempo devem seguir decisões sobre a RBB emitidas pela governança da rede. Esses participantes tem também o direito de executar os nós que enviam transações à rede e têm acesso à toda a *blockchain*. Por fim, os participantes aderentes parceiros têm o direito de apresentar propostas e participar das reuniões de governança.

Partícipes aderentes **associados** têm os mesmos direitos e deveres dos partícipes aderentes parceiros, ou seja, devem promover o uso da RBB e podem executar nós que enviam transações e participar de reuniões da governança. Porém, além de terem o direito de participar de reuniões, esses participantes podem votar nas propostas apresentadas por outros partícipes. Além disso, esses partícipes têm o compromisso de executar nós que possam participar do protocolo de consenso da rede *blockchain*.

Partícipes **patronos** são os fundadores da rede, ou seja, apenas o BNDES e o TCU são patronos da rede. Os direitos e deveres são os mesmos dos partícipes aderentes associados. Porém, essas duas entidades possuem o direito de desempatar votações, além de poderem vetar propostas apresentadas por outros participantes.

Caso uma nova instituição deseje se tornar uma participante, a mesma pode solicitar a adesão à rede. Essa solicitação se dá a partir do preenchimento do Termo de Adesão.

Tabela 4.1: Participantes que já adicionaram blocos na RBB

Instituição	Modo de participação	Endereço na rede <i>blockchain</i>
BNDES	Partícipe patrono	0x8cbcd6ba98fd722a54126edec5f20f1df0c713fc
TCU	Partícipe patrono	0xb7ff73c24d11de65c929d9643a1fe599d43ad13f
DATAPREV	Partícipe Aderente Associado	0x35a69e25ed7b31c18c9c005d2a6dade9cdb64b8a
CPQD	Partícipe Aderente Associado	0x484b67ecb3ae10fa984f7741ccd71ccc07dbdbb9
RNP	Partícipe Aderente Associado	0xd3b701e7dc92941634c30cc2dab3a0299f50e689
PRODEMGE	Partícipe Aderente Associado	0x742f72dce9a9dfcd9b12b431842d378540b40b44
SERPRO	Partícipe Aderente Associado	0xc88998c2994cbbefb668d43f96e19a65b3e08c55
IBICT	Partícipe Aderente Associado	0x9c9893d29e8350dbefe64e9215e58fa91815caf5

Fonte: (O Autor, 2025)

No termo, devem ser designados os representantes da organização que irão acompanhar o acordo, e também deve especificar se o possível novo aderente será um participante parceiro ou associado. O novo aderente não pode ser um participante patrono, visto que apenas o BNDES e TCU ocupam esse posto. Além do termo de adesão, é necessário que os representantes indicados no termo assinem um termo de confidencialidade, que deve ser apresentado após a aceitação do novo partícipe, caso ela venha a ocorrer.

4.4 Topologia da RBB

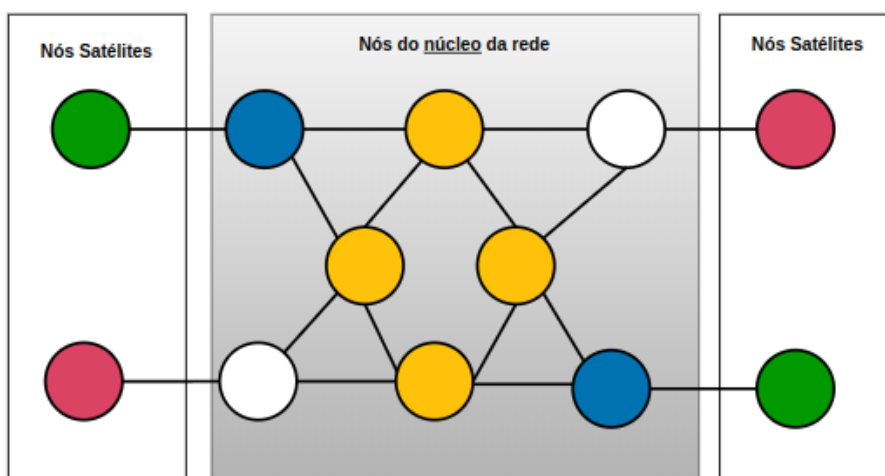
A RBB de base na rede *LAACChain* para definir o seu modelo de topologia. A rede possui dois grupos de nós, os nós núcleo e os nós satélites. Os nós do primeiro grupo desempenham um papel essencial para o correto funcionamento da rede, sem eles a rede não funciona. Já os nós do segundo grupo não possuem um papel essencial no correto funcionamento da rede, eles podem se conectar e desconectar da rede sem que isso prejudique o funcionamento geral. A Figura 4.1 demonstra de forma gráfica a topologia da rede, com os nós e relação existente entre eles, a Figura 4.2 traz uma legenda para melhor entendimento do diagrama.

Dentro do grupo dos nós núcleo, há dois tipos de nós, os nós conectores e os nós validadores. Nós conectores, ou *boot nodes*, são responsáveis pela comunicação entre nós validadores e nós satélites da rede. As tarefas de um nó conector são compartilhar o histórico e o estado dos blocos com novos nós, atualizar nós satélites sobre a geração de novos blocos geradores pelos nós validadores, repassar transações enviadas pelos nós registradores aos nós validadores, conectar-se com os nós validadores e nós registradores designados. Além desse tipo de nó, esse grupo possui os nós validadores, que são responsáveis por validar as transações submetidas à rede. As tarefas de um nó validador são participar do protocolo de consenso e ser responsável pela geração de um novo bloco e

conectar com outros nós do mesmo tipo e com nós conectores.

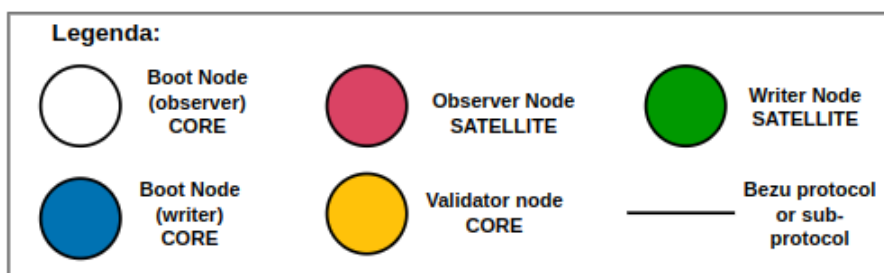
Dentro do grupo dos nós satélites, existem dois outros tipos de nós, os nós registradores e os nós observadores. Os nós registradores podem submeter transações à rede. O papel de um nó registrador é enviar transações aos nós conectores, para que essas transações sejam encaminhadas aos validadores, para isso esses nós devem se conectar com nós conectores designados. Os nós observadores só podem ler as informações e dados armazenados na rede. A única função de um nó observador é se conectar com nós conectores, mas esses conectores devem estar abertos para permitir a leitura de blocos.

Figura 4.1: Representação gráfica da topologia da RBB



Fonte: (BLOCKCHAIN, 2025b)

Figura 4.2: Legenda da representação da topologia da RBB



Fonte: (BLOCKCHAIN, 2025b)

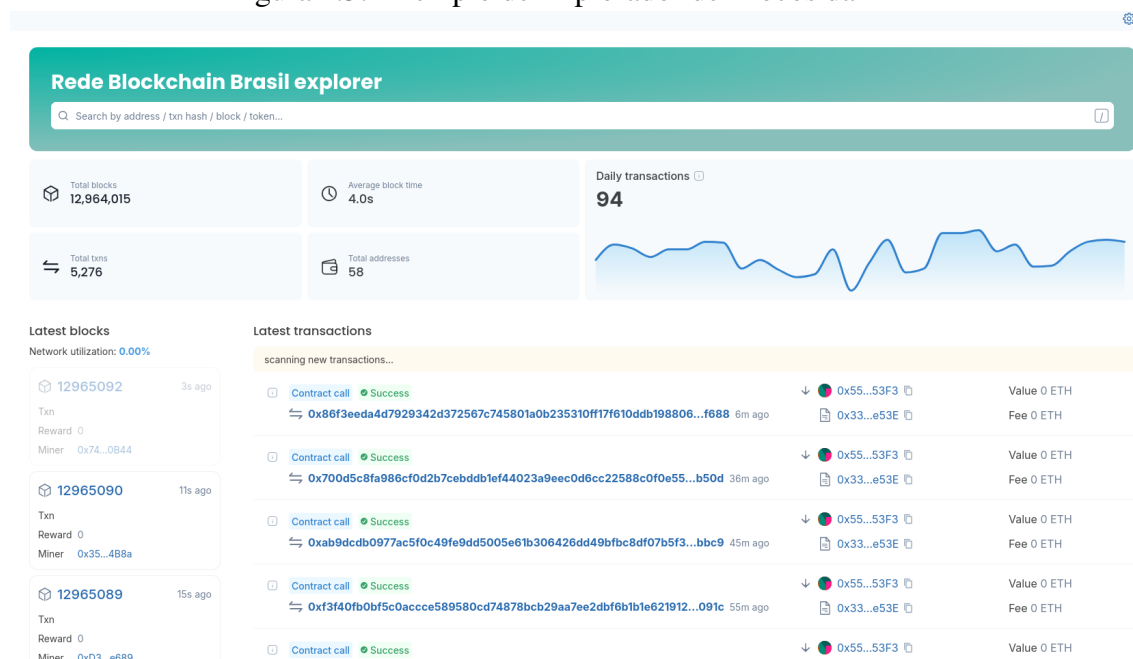
4.5 Explorador de Blocos da RBB

Como discutido na Seção 4.4, a RBB possui em sua topologia os nós observadores, que podem ler os dados armazenados na rede. A esses nós podem estar conectados os chamados exploradores de blocos. Um explorador possui uma interface gráfica, geral-

mente *online*, permitindo que usuários realizem pesquisas na rede *blockchain*, podendo obter e recuperar informações sobre transações, endereços, blocos e taxas. Cada explorador de bloco mostra as informações sobre uma determinada rede, podendo variar quais informações são exibidas de acordo com a arquitetura da rede.

A documentação da RBB fornece instruções para a instalação de um explorador de blocos, que podem ser seguidas por qualquer usuário que deseje ter acesso aos dados da rede e visualizá-los de forma gráfica. A Figura 4.3 mostra um exemplo de explorador de blocos da RBB, que pode ser acessado *online* por qualquer usuário.

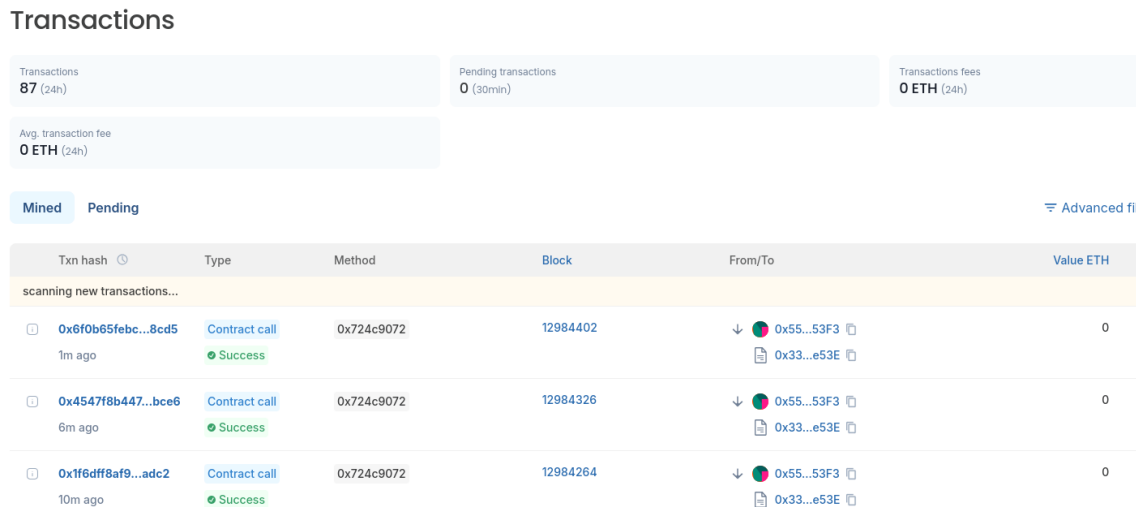
Figura 4.3: Exemplo de Explorador de Blocos da RBB



Fonte: (BLOCKCHAIN, 2025a)

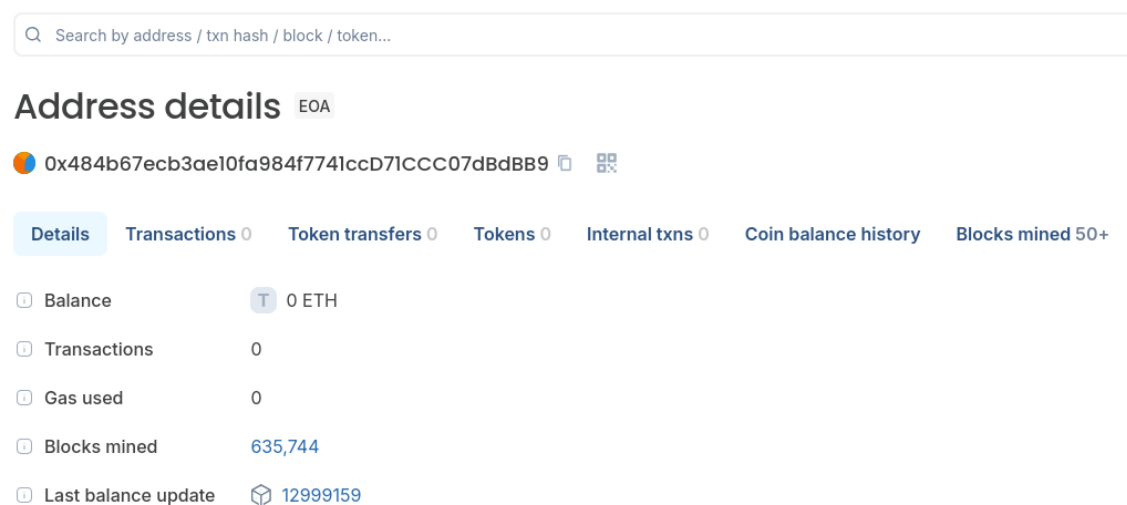
Através do explorador de blocos da RBB, podemos ver as transações da rede, como é mostrado na Figura 4.4. A partir dessa imagem, podemos identificar diversas informações, como o tipo da transação, identificador da transação, o número do bloco e o endereço dos participantes. Além disso, o usuário pode filtrar os dados que deseja ver a partir de informações como endereço, número do bloco ou identificador de transação. Na Figura 4.5 podemos verificar os detalhes de um participante utilizando seu endereço. Podemos verificar, através da Tabela 4.1, que este endereço é referente à organização CPQD, um participante aderente associado, que tem como dever executar nós da rede que participam do protocolo de consenso, o que pode ser verificado utilizando o explorador de nós, dado que, como apresentado na imagem, esse participante já minerou mais de 630.000 blocos da rede *blockchain*.

Figura 4.4: Visualização de transações da RBB



Fonte: (BLOCKCHAIN, 2025a)

Figura 4.5: Visualização de informações a partir de um endereço de participante da RBB



Fonte: (BLOCKCHAIN, 2025a)

Além disso, através desse explorador de blocos, é possível fazer a verificação de contratos inteligentes que estão armazenados na rede. Como é mostrado na Figura 4.6, é necessário o endereço do contrato na rede, a licença e o método de verificação. Assim, com todas essas funcionalidades oferecidas pelo uso de um nó explorador em conjunto com um explorador de blocos, é possível afirmar que a RBB é uma rede auditável pelo público.

Figura 4.6: Visualização de informações a partir de um endereço de participante da RBB

Verify & publish contract

Contract address to verify

Smart contract / Address (0x...)*

Contract license

For best practices, all contract s
holders, publishers and authors
also specify the accompanying l
verified contract source code pr

Currently, Blockscout supports 8 contract verification methods ⓘ

Verification method (compiler type)*

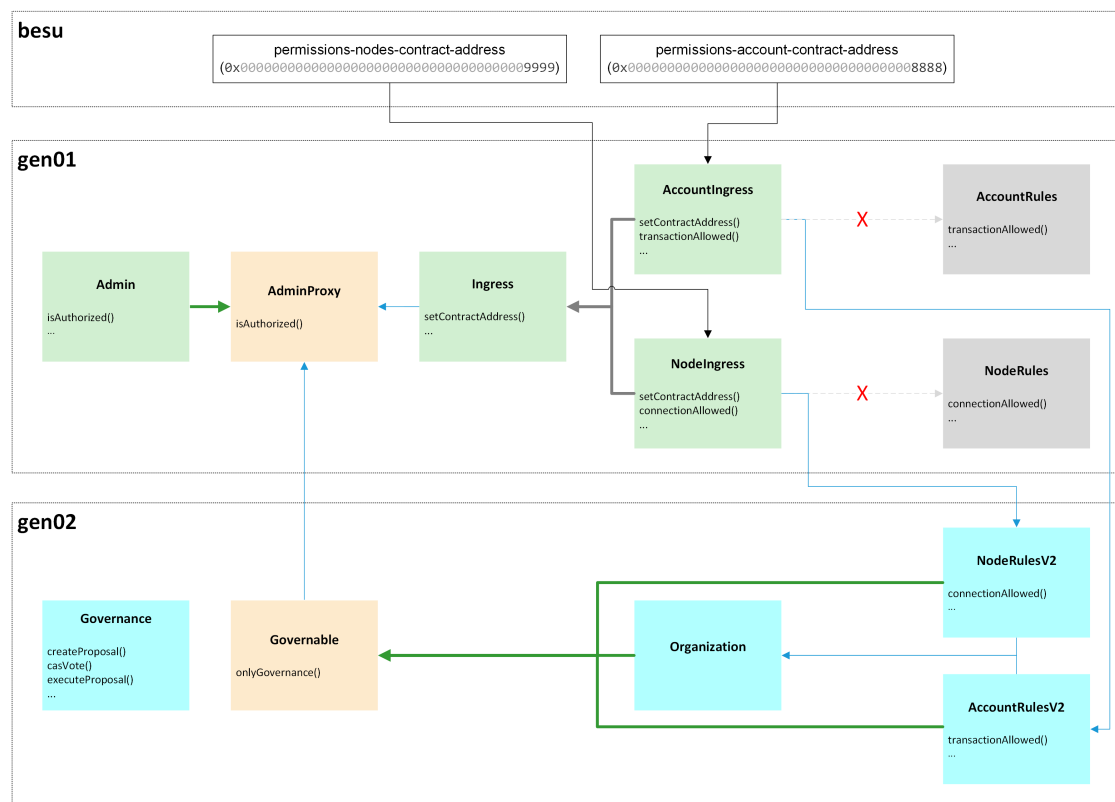
Fonte: (BLOCKCHAIN, 2025a)

4.6 Contratos inteligentes de permissionamento da RBB

Como citado na Seção 4.1, a RBB é uma rede *blockchain* no modelo público-permissionada, por conta disso é necessário que hajam certas restrições de acesso sendo aplicadas na rede. A RBB utiliza o padrão *Permissioned Blockchains Specification*, criado pela *Enterprise Ethereum Alliance*, assim permitindo que as regras de permissionamento sejam feitas *on chain*, ou seja, estejam armazenadas dentro da própria rede.

Para que esse permissionamento possa ser armazenado na rede, são utilizados os contratos inteligentes, desenvolvidos utilizando a linguagem de programação *Solidity*. Esses contratos contém as regras e são salvos, de forma imutável, na *blockchain*. A RBB divide esse esquema de permissionamento em várias gerações de evolução, cada uma contendo uma geração de contratos inteligentes. A imagem 4.7 demonstra de forma gráfica as gerações atuais da RBB, apresentando os contratos que cada uma possui e a relação entre as gerações.

Figura 4.7: Esquema de permissionamento da RBB



Fonte: (BLOCKCHAIN, 2025b)

A primeira geração, chamada de *gen01*, que foi criada no início da operação da RBB, possui a primeira geração de contratos inteligentes da rede. Essa geração possui os contratos inteligentes *AccountIngress* e *NodeIngress*. Esses são contratos inteligentes são cruciais para a RBB pois lidam com autorização de membros, por isso, são constantes no arquivo gênese da rede. Apenas o *bytecode*, que será executado na EVM, está permanente no gênese e não o código em *Solidity*. Logo, não podem ser alterados. Tais contratos serão detalhados e listados no Capítulo 5. Além disso, a *gen01* é importante pois contém os *proxies* *AccountIngress* e *NodeIngress*. Esses *proxies* permitem o reponteiramento de regras, ou seja, as regras de acesso para contas e nós e a gestão de contas de administração.

A segunda geração de contratos inteligentes da RBB, chamada *gen02*, introduz novas regras de permissionamento à rede. Essas novas regras contemplam novos conceitos e funcionalidades mais elaboradas para a gestão de acesso, como a gestão de organizações, definição de perfis de acesso e procedimentos de governança através de votações.

5 ANÁLISE DE CONTRATOS E DISCUSSÃO

Neste capítulo será apresentada a análise e discussão que este trabalho se propôs a fazer. Serão analisados os dois contrato inteligentes principais da RBB, o *AccountIngress*, apresentado no *Listing 5.1*, e o *NodeIngress*, apresentado no *Listing 5.2*. A partir disso, espera-se obter uma análise técnica do funcionamento desses contratos, bem como uma avaliação empírica dos mesmos. Além disso, será apresentado o processo da tentativa de implantação de um nó observador na rede do INF, detalhando as dificuldades e as barreiras enfrentadas que levaram à falha da implantação.

5.1 Análise do Contrato Inteligente *AccountIngress*

Como podemos ver pelo *Listing 5.1*, o contrato *AccountIngress* possui 31 linhas de código. Nessa seção, o contrato será explicado linha a linha, detalhando de maneira técnica o funcionamento do código.

Listing 5.1: Contrato inteligente *AccountIngress* da RBB

```

1  pragma solidity 0.5.9;
2
3  import "./AccountRulesProxy.sol";
4  import "./Ingress.sol";
5
6
7  contract AccountIngress is Ingress {
8      // version of this contract: semver eg 1.2.14 represented like 001002014
9      uint private version = 1000000;
10
11     function getContractVersion() public view returns(uint) {
12         return version;
13     }
14
15     function transactionAllowed(
16         address sender,
17         address target,
18         uint256 value,
19         uint256 gasPrice,
20         uint256 gasLimit,
21         bytes memory payload
22     ) public view returns (bool) {
23         if(getContractAddress(RULES_CONTRACT) == address(0)) {
24             return true;
25         }
26

```

```

27         return AccountRulesProxy(registry[RULES_CONTRACT]).transactionAllowed(
28             sender, target, value, gasPrice, gasLimit, payload
29         );
30     }
31 }

```

Fonte: (BLOCKCHAIN, 2025c)

Na primeira linha, temos uma linha comum a todos os contratos inteligentes escritos em *Solidity*, a definição da versão do compilador a ser usado para que esse código possa ser compilado e gerado os *opcodes* que serão executados na EVM.

Nas linhas 3 e 4, temos importações de outros módulos, assim como ocorre em linguagens de programação tradicionais. Nesse caso, são importados os arquivos *AccountRulesProxy*, apresentado no *Listing 5.3*, e *Ingress.sol*, presente no *Listing 5.5*. Ambas as importações são explicadas de maneira resumida na Seção 5.3

Na linha 7, é definido o contrato inteligente *AccountIngress* presente nesse módulo. Nessa mesma linha, temos o uso da *keyword is*, que indica o uso da propriedade de herança, ou seja, a o contrato *AccountIngress* herda os atributos do módulo *Ingress*. Como discutido na Subseção 5.3.3, o contrato *Ingress*, nesse contexto, a classe pai, possui seis funções que podem ser utilizadas por essa classe filha.

Dentro do contrato, na linha 9, temos a declaração de uma variável, chamada *version*, com o tipo *uint*. Em *Solidity*, *uint* é um nome alternativo para *uint256* (Solidity Documentation, 2025), um inteiro sem sinal de até 8 *bytes* de tamanho, ou seja, um valor entre 0 e $2^{256}-1$. Além disso, ela é declarada como privada, ou seja, pode ser acessada apenas dentro desse contrato onde ela é declarada. Essa variável guarda a versão desse contrato, nesse caso, ela está sendo inicializada com o valor 10000.

Após a declaração das variáveis do contrato, temos as definições das funções que compõem esse módulo. A primeira função implementada dentro do *AccountIngress*, na linha 11, é a *getContractVersion*, que não recebe nenhum argumento como parâmetro. Essa função é declarada com o nível de acesso público, ou seja, pode ser chamada de fora do contrato, além disso é usada a *keyword view* para identificar a mutabilidade da função, deixando claro que o seu uso não irá alterar o estado, nem nenhum dado armazenado por esse contrato. Por fim, é declarado o tipo do retorno da função como *uint*, tipo de dado já explicado anteriormente. Dentro da *getContractVersion*, temos apenas a declaração do retorno da função como sendo o valor armazenado na variável *version*, ou seja, a função é apenas um método utilizado para retornar a versão daquele contrato.

Após a declaração da função anterior, na linha 15, temos a definição da função

transactionAllowed. Essa função está presente na interface importada na linha 3 e, como explicado na Subseção 5.3.2, contratos que importam esse módulo devem implementar essa função a partir da assinatura definida na interface. Assim, os argumentos recebidos como parâmetro, o nível de acesso, a mutabilidade e o tipo retornado pela função, são os mesmos definidos na interface. Dentro da implementação da função, na linha 23, temos uma instrução condicional que utiliza a função *getContractAddress*, herdado do módulo *Ingress* e explicado na Subseção 5.3.3, assim, é verificado o endereço do contrato de contrato de regras na rede *blockchain*. Caso o contrato tenha endereço vazio, ou seja, não está armazenado na rede, o valor retornado pela função é o valor booleano verdadeiro. Por outro lado, caso o contrato exista na rede, a função continua sua lógica, e na linha 27, temos um retorno não condicional. Nessa parte, é buscado o endereço do contrato de regras dentro do registro de contratos inteligentes. A partir dessa informação, é informado ao compilador que esse contrato deve ser tratado como um *AccountRulesProxy*, que é uma interface, e deve ser chamada a função *transactionAllowed*, que deve ter sido implementada para o contrato de regras.

Em resumo, o módulo *AccountIngress* implementa um contrato inteligente que guarda uma versão de contrato e verifica se existe um contrato de regras definido na rede *blockchain*. Caso esse contrato não exista, toda e qualquer transação é permitida, caso ele exista, o *AccountIngress* atua como um roteador, repassando os dados para que a transação seja avaliada por esse contrato de regras.

5.2 Análise do contrato *NodeIngress*

Assim como o contrato inteligente *AccountIngress*, discutido na Seção 5.1, o módulo do *NodeIngress*, mostrado no *Listing 5.2*, começa por definir, logo na primeira linha, a versão do compilador. E logo após, nas linhas 3 e 4, temos a importação dos módulos *NodeRulesProxy* e *Ingress*, explicados na Seção 5.3, que serão utilizados nesse contrato.

Listing 5.2: Contrato inteligente *NodeIngress* da RBB

```

1 pragma solidity 0.5.9;
2
3 import "./NodeRulesProxy.sol";
4 import "./Ingress.sol";
5
6
7 contract NodeIngress is Ingress {
8     // version of this contract: semver eg 1.2.14 represented like 001002014

```

```

9      uint private version = 1000000;
10
11      event NodePermissionsUpdated(
12          bool addsRestrictions
13      );
14
15      function getContractVersion() public view returns(uint) {
16          return version;
17      }
18
19      function emitRulesChangeEvent(bool addsRestrictions) public {
20          require(registry[RULES_CONTRACT] == msg.sender, "Only Rules contract can trigger
21              ↳ Rules change events");
22          emit NodePermissionsUpdated(addsRestrictions);
23      }
24
25      function connectionAllowed(
26          bytes32 sourceEnodeHigh,
27          bytes32 sourceEnodeLow,
28          bytes16 sourceEnodeIp,
29          uint16 sourceEnodePort,
30          bytes32 destinationEnodeHigh,
31          bytes32 destinationEnodeLow,
32          bytes16 destinationEnodeIp,
33          uint16 destinationEnodePort
34      ) public view returns (bytes32) {
35          if(getContractAddress(RULES_CONTRACT) == address(0)) {
36              //accept connection
37              return 0xffffffffffffffffffffffffffffffffffffffffffffffffffffffff;
38          }
39
40          return NodeRulesProxy(registry[RULES_CONTRACT]).connectionAllowed(
41              sourceEnodeHigh,
42              sourceEnodeLow,
43              sourceEnodeIp,
44              sourceEnodePort,
45              destinationEnodeHigh,
46              destinationEnodeLow,
47              destinationEnodeIp,
48              destinationEnodePort
49          );
50      }

```

Fonte: (BLOCKCHAIN, 2025c)

Novamente, assim como o contrato inteligente discutido na seção anterior, nesse módulo, o contrato *NodeIngress* é declarado usando a *keyword is*, indicando uma herança do contrato *Ingress*, também explicado na Seção 5.3. Além disso, esse contrato também guarda uma variável identificado a sua versão.

Após a declaração da variável que guarda a versão do contrato, é definido um evento, na linha 11, chamado *NodePermissionsUpdated*. Esse evento recebe como parâmetro um valor booleano, indicando se houve adição de restrições para as permissões daquele nó.

A partir da linha 15, temos a declaração das funções que compõe esse contrato inteligente. A primeira delas, chamada *getContractVersion*, também está presente no contrato *AccountIngress*, explicado na Seção 5.1, e funciona da mesma maneira, apenas retorna o valor armazenado na variável *version*.

Após a primeira função, na linha 19, temos a declaração da função *emitRulesChangeEvent*. Essa função recebe como parâmetro um valor booleano, que indica a adição de restrições para aquele nó. Dentro da definição da função, temos o uso da *keyword require*, usada para verificar condições e lançar exceções caso a condição não seja atendida. Nessa caso, são passados dois argumentos, o primeiro é a condição, e o segundo é o texto que será apresentado caso a exceção ocorra. Assim, é verificado se o nó remetente da mensagem é o nó que guarda o contrato inteligente de regras, e caso não seja, é jogada uma exceção com um texto indicando que apenas o contrato de regras pode emitir eventos de mudança de regras. No caso de a condição ser verdadeira, a função continua sua execução e o evento *NodePermissionsUpdated* é disparado. Essa função é declarada com visibilidade pública, ou seja, pode ser chamada de fora do contrato *AccountIngress*.

Por fim, temos a implementação da função *connectionAllowed*, definida na interface *NodeRulesProxy*, explicada na Subseção 5.3.4. Dentro da implementação, primeiramente é verificado se o contrato inteligente de regras está presente na rede *blockchain*, através da função *getContractAddress*, herdada do contrato *Ingress* e explicada na Subseção 5.3.3. Caso o endereço retornado seja 0, ou seja, o contrato não existe na rede, a conexão entre os dois nós é permitida. Se o contrato de regras existir, ele é buscado no registro de contratos e é chamada a função *connectionAllowed* desse contrato, passando como parâmetro os mesmos argumentos usados no contrato atual.

Em resumo, o módulo *NodeIngress* implementa um contrato inteligente que gerencia os nós da rede *blockchain*. É esse contrato que controla a mudança de restrições de permissão de um nó. Além disso, esse contrato controla a conexão entre dois nós da rede, verificando se o contrato de regras existe. Caso não exista, o próprio contrato aprova a conexão entre quaisquer dois nós, por outro lado, se o contrato de regras existir, o contrato inteligente *NodeIngress* atua como um roteador, repassando informações para a aprovação de conexão seja feita pelo contrato de regras.

5.3 Análise dos Módulos Importados

5.3.1 *AccountRulesProxy*

O módulo *AccountRulesProxy*, apresentado no *Listing 5.3*, é uma interface. Interfaces são tipos de dados, presentes em linguagens de programação orientadas a objetos, como a *Solidity*, que atuam como uma abstração de uma classe. Uma classe que utiliza a interface pode implementar os métodos da maneira que precisar.

Na *AccountRulesProxy*, temos apenas um método, uma função chamada *transactionAllowed*, que recebe como argumento uma série de parâmetros, sendo eles o remetente e o destinatário da transação, o valor, o preço do gás, o valor limite de uso de gás para a transação e o conteúdo, chamado de *payload*. Além disso, a função é definida como externa, com o uso da *keyword external*, ou seja, só pode ser chamada de fora do contrato, o uso da *keyword view* indica que a função não altera o estado do contrato. Além disso, é definido o retorno da função como sendo um valor booleano.

Listing 5.3: Código da interface *AccountRulesProxy* da RBB

```

1 pragma solidity 0.5.9;
2
3 interface AccountRulesProxy {
4     function transactionAllowed(
5         address sender,
6         address target,
7         uint256 value,
8         uint256 gasPrice,
9         uint256 gasLimit,
10        bytes calldata payload
11    ) external view returns (bool);
12 }
```

Fonte: (BLOCKCHAIN, 2025c)

5.3.2 *AdminProxy*

O módulo *AdminProxy*, apresentado no *Listing 5.4*, também é uma interface, assim como o *AccountRulesProxy*. O funcionamento da interface é o mesmo explicado na Subseção 5.3.1.

Na *AdminProxy*, existe apenas um método, a função *isAuthorized*, que recebe como parâmetro um endereço da rede *blockchain*. Assim como a função *transacatio-*

nAllowed, discutida na Subseção 5.3.1, a função *isAuthorized* é declarada com o uso das *keywords external* e *view*, e também tem como retorno um valor booleano.

Listing 5.4: Código do contrato AdminProxy da RBB

```
1 pragma solidity 0.5.9;
2
3 interface AdminProxy {
4     function isAuthorized(address source) external view returns (bool);
5 }
```

Fonte: (BLOCKCHAIN, 2025c)

5.3.3 Ingress

O módulo *Ingress*, primeiramente importa o código do contrato *AdminProxy*, que é explicado na Subseção 5.3.2.

Nesse contrato, temos a implementação de um evento, um mecanismo de *Solidity* que permite registrar informações no histórico da rede *blockchain*. Nesse caso, o evento *RegistryUpdated* é disparado sempre um contrato é adicionado, removido ou alterado, sendo registrado o endereço e o nome do contrato.

Além disso, o contrato implementa seis funções. A função *getContractAddress*, que recebe o nome de um contrato como parâmetro e retorna o endereço desse contrato na rede *blockchain*. Em seguida, a função *getSize* pode ser chamada para verificar a quantidade de contratos que foram registrados. Logo após, temos a implementação do método *isAuthorized*, nesse módulo, ele é usado para verificar se um endereço tem permissão para modificar o registro de contratos. Temos também a função *setContractAddress*, que recebe o nome de um contrato e um endereço, para que possa associar esse endereço ao contrato. A função *removeContract*, recebe o nome de um contrato e, caso o autor da chamada tiver permissão, remove o contrato. Por fim, a função *getAllContractKeys*, retorna uma cópia do vetor de chaves dos contratos.

Em resumo, o módulo *Ingress* implementa um registro de endereços de contratos, que podem ser modificados ou removidos por usuários com permissão de administrador na rede.

Listing 5.5: Código do contrato Ingress da RBB

```
1 pragma solidity 0.5.9;
2
3 import "./AdminProxy.sol";
```

```

4
5 contract Ingress {
6     // Contract keys
7     bytes32 public RULES_CONTRACT = 0
8         ↪ x72756c65730000000000000000000000000000000000000000000000000000; // "rules"
9
10    bytes32 public ADMIN_CONTRACT = 0
11        ↪ x61646d696e69737472617469666e0000000000000000000000000000000000; // "
12        ↪ administration"
13
14    // Registry mapping indexing
15    mapping(bytes32 => address) internal registry;
16
17    bytes32[] internal contractKeys;
18    mapping (bytes32 => uint256) internal indexOf; //1 based indexing. 0 means non-
19        ↪ existent
20
21    event RegistryUpdated(
22        address contractAddress,
23        bytes32 contractName
24    );
25
26    function getContractAddress(bytes32 name) public view returns(address) {
27        require(name > 0, "Contract name must not be empty.");
28        return registry[name];
29    }
30
31    function getSize() public view returns (uint256) {
32        return contractKeys.length;
33    }
34
35    function isAuthorized(address account) public view returns(bool) {
36        if (registry[ADMIN_CONTRACT] == address(0)) {
37            return true;
38        } else {
39            return AdminProxy(registry[ADMIN_CONTRACT]).isAuthorized(account);
40        }
41    }
42
43    function setContractAddress(bytes32 name, address addr) public returns (bool) {
44        require(name > 0, "Contract name must not be empty.");
45        require(addr != address(0), "Contract address must not be zero.");
46        require(isAuthorized(msg.sender), "Not authorized to update contract registry.")
47            ↪ ;
48
49        if (indexOf[name] == 0) {
50            indexOf[name] = contractKeys.push(name);
51        }
52
53        registry[name] = addr;
54
55        emit RegistryUpdated(addr, name);
56    }
57

```

```

51         return true;
52     }
53
54     function removeContract(bytes32 _name) public returns(bool) {
55         require(_name > 0, "Contract name must not be empty.");
56         require(contractKeys.length > 0, "Must have at least one registered contract to
           ↳ execute delete operation.");
57         require(isAuthorized(msg.sender), "Not authorized to update contract registry.")
           ↳ ;
58
59         uint256 index = indexOf[_name];
60         if (index > 0 && index <= contractKeys.length) { //1-based indexing
61             //move last address into index being vacated (unless we are dealing with
           ↳ last index)
62             if (index != contractKeys.length) {
63                 bytes32 lastKey = contractKeys[contractKeys.length - 1];
64                 contractKeys[index - 1] = lastKey;
65                 indexOf[lastKey] = index;
66             }
67
68             //shrink contract keys array
69             contractKeys.pop();
70             indexOf[_name] = 0;
71             registry[_name] = address(0);
72             emit RegistryUpdated(address(0), _name);
73             return true;
74         }
75         return false;
76     }
77
78     function getAllContractKeys() public view returns(bytes32[] memory) {
79         return contractKeys;
80     }
81 }

```

Fonte: (BLOCKCHAIN, 2025c)

5.3.4 NodeRulesProxy

O módulo *NodeRulesProxy*, mostrado no *Listing 5.3*, assim como os outros *proxies* apresentados, também é uma interface.

Na *NodeRulesProxy*, temos apenas um método, a função *connectionAllowed*. Essa função recebe um total de oito argumentos como parâmetros, que formam dois *enode URLs*, como explicado na Subseção 2.9. Porém, aqui temos o endereço dos nós divididos entre seus *bytes* mais significativos e os seus *bytes* menos significativos. Isso ocorre porque uma chave pública na rede *Ethereum* possui 64 *bytes*, mas em *Solidity*, o tamanho

máximo para um vetor de tamanho fixo é 32 *bytes* (Solidity Documentation, 2025). Ou seja, a função recebe como argumento a chave pública do nó de origem, o seu endereço IP e sua porta, assim formando o seu *enode URL*. Da mesma forma, é recebido os mesmo dados para o nó de destino.

Por fim, assim como as outras interfaces, discutidas nas subseções 5.3.1 e 5.3.2, a função *connectionAllowed* é declarada com o uso das *keywords external* e *view*. Porém, o tipo retornado pela função difere dos métodos das outras interfaces, nesse caso, a função deve retornar um vetor de tamanho fixo de 32 *bytes*.

Listing 5.6: Código da interface NodeRulesProxy da RBB

```

1 pragma solidity 0.5.9;
2
3 interface NodeRulesProxy {
4     function connectionAllowed(
5         bytes32 sourceEnodeHigh,
6         bytes32 sourceEnodeLow,
7         bytes16 sourceEnodeIp,
8         uint16 sourceEnodePort,
9         bytes32 destinationEnodeHigh,
10        bytes32 destinationEnodeLow,
11        bytes16 destinationEnodeIp,
12        uint16 destinationEnodePort
13    ) external view returns (bytes32);
14 }
```

Fonte: (BLOCKCHAIN, 2025c)

5.4 Validação dos Contratos

A validação de um contrato inteligente significa enviar o seu código fonte e verificar se o *bytecode* que está armazenado na rede *blockchain* corresponde àquele código.

Para verificar os contratos da RBB, é possível utilizar o explorador de blocos, como é mostrado na Seção 4.5. Para isso, é necessário informar o endereço do contrato na rede, a licença do contrato e o método que deve ser utilizado para a verificação, assim como mostrado na Figura 4.6. Após essas inserir informações, é necessário selecionar a versão do compilador e da EVM que devem ser usados, assim como é mostrado na Figura 5.1. Além disso, pode ser ativada a otimização, porém, sabe-se que ao otimizar o código, o *bytecode* dele muda, o que impossibilitaria a verificação, dado que o *bytecode* que está armazenado na rede *blockchain* não possui a otimização. Por fim, é necessário incluir o código fonte do contrato inteligente que se deseja verificar. Caso o contrato importe

outros contratos, é necessário incluir o código desses contratos também.

Figura 5.1: Seleção da versão do compilador e da EVM

Contract verification via Solidity (multi-part files)

Compiler*

▼

EVM Version*

default

▼

☒ Optimization enabled

Sources *.sol or *.yul files

Drop files or click here

The compiler version is specified in `pragma solidity X.X.X`. Use the compiler version rather than the nightly build. If using the Solidity compiler, run `solc --version` to check.

The EVM version the contract is written for. If the bytecode does not match the version, we try to verify using the latest EVM version. [EVM version details](#)

Upload all Solidity or Yul contract source files.

Fonte: (BLOCKCHAIN, 2025a)

Primeiramente, foi feita verificação do contrato inteligente *AccountIngress*. O endereço do contrato foi retirado do bloco gênese da rede. Para realizar a verificação, foi necessário escolher a versão do compilador *0.5.9*, assim como definida no código do contrato. Além disso, foi necessário adicionar os módulos importados pelo *AccountIngress*, sendo eles o módulo *AccountRulesProxy* e o módulo *Ingress*. A ferramenta de verificação sendo utilizada para verificar esse contrato está demonstrada na Figura 5.2.

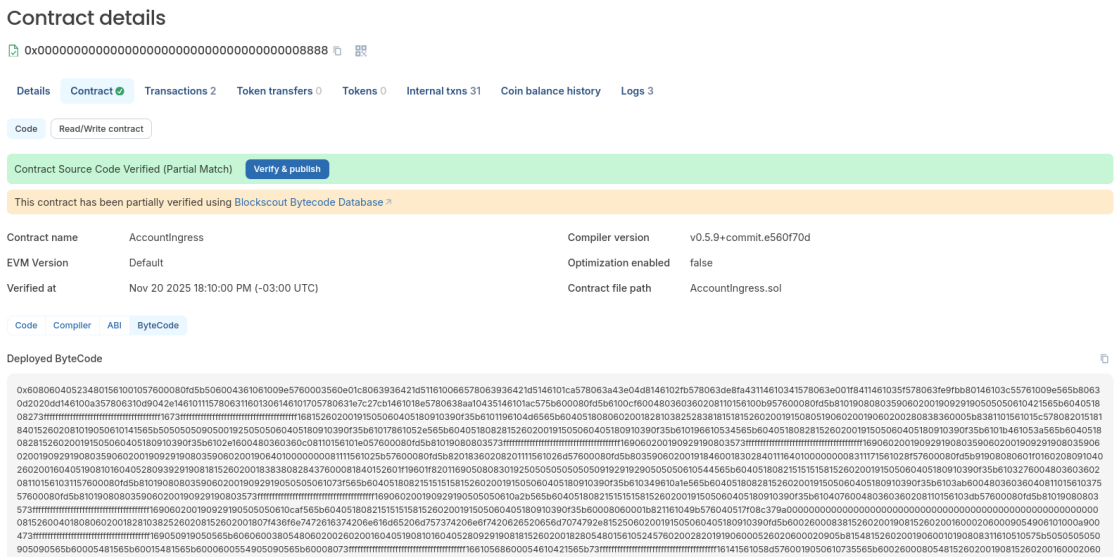
A verificação não foi concluída com sucesso. Como é possível ver na Figura 5.2, é exibido o erro *Cannot update partially verified smart contract with another partially verified contract*. Esse erro é conhecido para a explorador de blocos *Blockscout*, que é explorador utilizado nesse caso. Usuários já relataram o mesmo problema no repositório oficial do *Blockscout* (GITHUB, 2025), porém, ainda não há uma solução oficial para o problema.

Figura 5.3: Contratos que já foram verificados usando o explorador de blocos



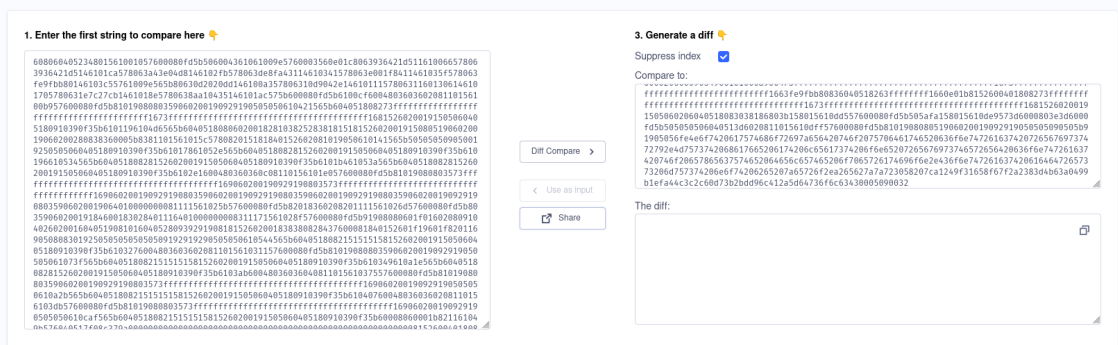
Fonte: (BLOCKCHAIN, 2025a)

Figura 5.4: Contratos *AccountIngress* verificado



Fonte: (BLOCKCHAIN, 2025a)

Figura 5.5: Comparação do *bytecode* presente no bloco gênese com o *bytecode* gerado pelo explorador de blocos



Fonte: (COMPARE, 2025)

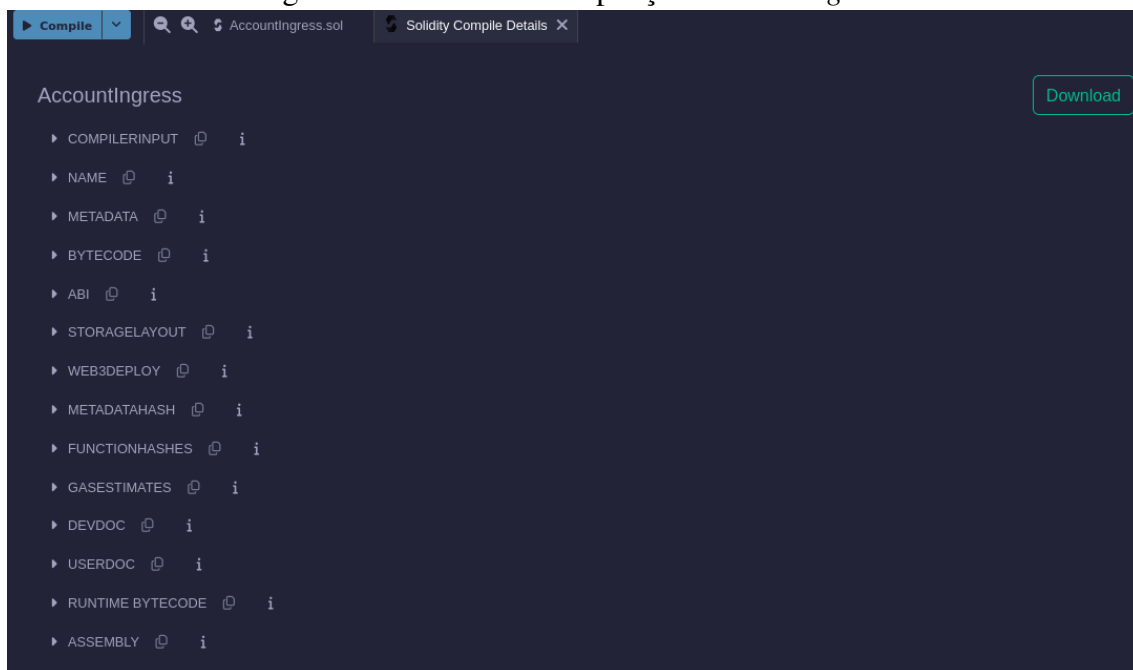
Como a verificação utilizando o explorador de blocos falhou, foi feita a verificação de maneira manual do contrato inteligente *AccountIngress*. Para isso, foi necessário fazer a compilação do código *Solidity*, de forma a gerar o *bytecode*. A partir disso, foi feita a

mesma comparação, entre o *bytecode* armazenado no bloco gênese e o *bytecode* gerado pelo compilador.

A ferramenta utilizada foi a *Remix IDE* (IDE, 2025). O *Remix* é uma ferramenta que pode ser executada no navegador, sendo considerada um ambiente de desenvolvimento integrado (*IDE*) que pode ser utilizado para escrever, testar, depurar e implantar contratos inteligentes escritos em *Solidity* para uma rede *blockchain Ethereum*.

Para verificar o contrato, foi necessário carregar os arquivos dos códigos dos contratos no *Remix*, definir a versão do compilador para que seja igual à declarada no início do código mostrado no *Listing 5.1*. Após a compilação, temos diversas informações geradas, como pode ser visto na Figura 5.6. Uma dessas informações é o *bytecode* gerado pela compilação.

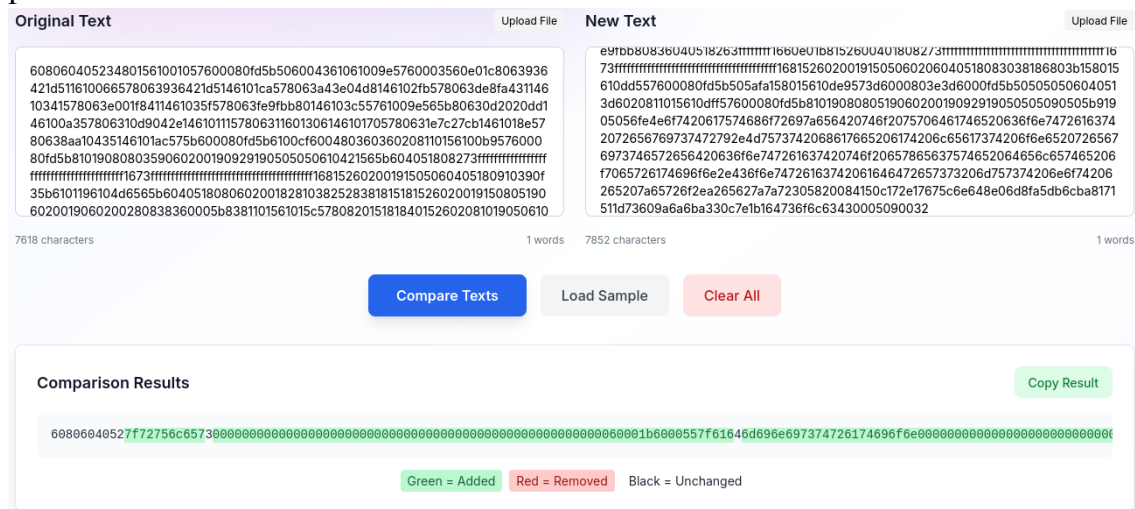
Figura 5.6: Resultado compilação *AccountIngress*



Fonte: (IDE, 2025)

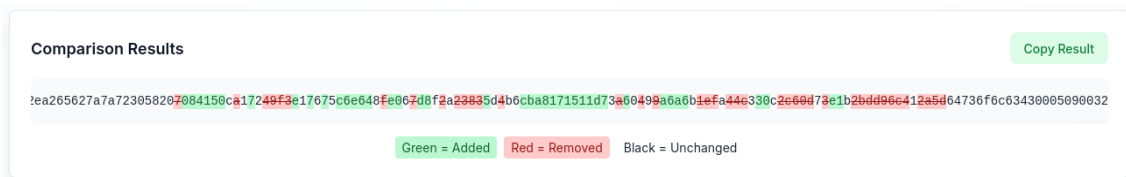
A partir dessa informação, foi feito mesmo processo descrito na Seção 5.4, em que o *bytecode* gerado pela compilação foi comparado com o *bytecode* presente no bloco gênese da rede *blockchain*. Como é possível ver na Figura 5.7, o *bytecode* gerado pela compilação feita no *Remix* é diferente do *bytecode* armazenado no bloco gênese, possuindo caracteres a mais, bem como caracteres removidos, como é mostrado na Figura 5.8.

Figura 5.7: Comparação do *bytecode* presente no bloco gênese com o *bytecode* gerado pelo *Remix*



Fonte: (TOOL, 2025)

Figura 5.8: Segunda parte do resultado da comparação de *bytecodes*



Fonte: (TOOL, 2025)

5.5 Discussão

O uso de contratos inteligentes pode trazer diversos benefícios para uma aplicação. Uma das grandes vantagens de ter o código implantado, por meio de contratos inteligentes, em uma rede *blockchain* é a possibilidade de verificar esse código. Como discutido na Seção 5.4, ao verificar um contrato, podemos atestar que o código que está na rede é realmente o código ao qual temos acesso, gerando uma maior confiança em aplicações que utilizam a tecnologia da *blockchain*. Por outro lado, códigos que executam em um servidor centralizado mantido por uma única organização, geralmente em um servidor remoto, não possuem garantia alguma para o usuário, de forma que não temos confirmação alguma que o código irá executar da maneira correta.

A RBB não disponibiliza apenas a ferramenta de verificação, mas também o uso de nós observadores, que têm seu funcionamento detalhado na Seção 4.4, tornando a rede totalmente auditável pelo público externo. A esse nó observador pode ser aliado o uso de

um explorador de blocos, tornando a experiência mais intuitiva ao mostrar os dados de forma gráfica e permitindo a validação dos contratos. A auditoria é um fator crucial para o sucesso da uma rede *blockchain* como a RBB, que se propõem a ser uma rede de uso governamental.

Além disso, os contratos inteligentes analisados, *AccountIngress* e *NodeIngress*, podem nos dizer muito sobre a adoção da tecnologia da *blockchain*. Em apenas poucas linhas de código, foi implementado, com sucesso, uma maneira de controlar o ingresso de novas contas e nós, e as suas relações dentro da rede *blockchain* da RBB. Porém, esses contratos podem apresentar problemas para o funcionamento da rede. Como, por exemplo, caso ocorresse um incidente que impactasse o contrato de regras, levaria a um cenário desastroso. Pois, caso esse contrato não seja encontrado, no contrato *AccountIngress*, como explicado na Seção 5.1, toda transação terá sua execução permitida e no contrato *NodeIngress*, como explicado na Seção 5.2, toda conexão entre quaisquer dois nós será permitida.

Por fim, é possível apontar possíveis melhorias do ponto de vista da engenharia de *software*. Ambos os contratos *AccountIngress* e *NodeIngress* possuem a mesma variável *version* com o mesmo valor, além da mesma função *getContractVersion*, que tem a mesma implementação nos dois casos, como pode ser visualizado nos *listings* 5.1 e 5.2. Ambos os contratos herdam as propriedades do contrato inteligente *Ingress*, essa função poderia ser definida dentro do contrato pai e apenas reutilizado pelos contratos filhos. Além disso, como podemos ver no início de ambos os módulos, a versão do compilador *Solidity* utilizado está defasada (SOLIDITY, 2025). Seria interessante atualizar esses contratos para versões mais novas do compilador, afim de evitar problemas que versões antigas possam enfrentar.

5.6 Dificuldades - Implantação do Nó Observador

Esse trabalho se propôs a implantar um nó observador na rede do INF. Porém, ao longo do desenvolvimento, foram encontradas barreiras que impossibilitaram a completude desse objetivo. O processo de implantação do nó observador foi feito seguindo o roteiro oficial da RBB (BLOCKCHAIN, 2025d).

Primeiramente, o nó foi implantado de maneira local, em um computador conectado a uma rede de Internet doméstica, com sucesso. Após termos um nó observador rodando, partimos para a etapa de conectar o nó a um explorador de blocos. O guia ofi-

cial, fornece duas alternativas, o *Chainlens* e o *Blockscout*. Optamos pela segunda opção, embora o seu uso exigisse mais poder computacional, visto que esse explorador oferece a ferramenta de validação de contratos da rede.

Após termos um explorador de blocos, conectado a um nó observador, rodando de maneira local, partimos para a implantação na rede do INF. No primeiro momento, foi utilizado um ambiente de contêineres. Para isso, foi necessário criar uma imagem *Docker* contendo as instruções de instalação do nó e do observador. Porém, o próprio nó e o explorador de blocos já são executados dentro de um contêiner *Docker*. Por conta disso, foi necessário criar um contêiner que executava um segundo contêiner. Isso fez com que a conexão da aplicação do explorador de blocos para o mundo externo fosse inviável.

Com essa primeira falha utilizando contêineres, tivemos de repensar o método a ser utilizado para a implantação. Como o problema foi o fato de termos de “contêinerizar” a instalação, partimos para a abordagem de instalar diretamente em uma máquina o nó e o explorador. A instalação foi feita com sucesso e o explorador de blocos era acessível dentro da rede do INF. Porém, o nó observador não conseguiu indexar nenhum bloco da rede. Isso aconteceu porque o nó observador precisa se conectar a, no mínimo, cinco *peers* distintos para indexar os blocos, porém, ele não conseguiu se conectar a nenhum, isso é demonstrado na Figura 5.9. É possível que o problema tenha sido causado por alguma restrição de rede do INF, ou algum *firewall* que tenha bloqueado a conexão com os outros nós da rede *blockchain* da RBB.

Caso a implantação do nó observador e do explorador de blocos tivesse obtido sucesso, teríamos um endereço que seria exposto ao mundo externo, de forma que outros usuários pudessem acessar essa ferramenta. O resultado final seria semelhante ao exposto na Figura 4.3.

Figura 5.9: *Screenshot* da tentativa de instalação do nó observador

```

observer-1 | 2025-11-18T17:01:22.153-0300 INFO Flat db mode found FULL
observer-1 | 2025-11-18T17:01:22.154-0300 INFO DB mode with code stored using code hash enabled = true
observer-1 | 2025-11-18T17:01:22.289-0300 INFO Enabling transaction pool
observer-1 | 2025-11-18T17:01:22.307-0300 INFO QBFT is configured, creating initial sync for BFT
observer-1 | 2025-11-18T17:01:22.308-0300 INFO Creating pivot block selector for BFT node
observer-1 | 2025-11-18T17:01:22.346-0300 INFO Resolved 7 bootnodes.
observer-1 | 2025-11-18T17:01:22.356-0300 INFO Detecting NAT service.
observer-1 | 2025-11-18T17:01:22.518-0300 INFO Starting external services ...
observer-1 | 2025-11-18T17:01:22.519-0300 INFO Starting metrics http service on 0.0.0.0:9545
observer-1 | 2025-11-18T17:01:22.542-0300 INFO Starting JSON-RPC service on 0.0.0.0:8545
observer-1 | 2025-11-18T17:01:22.667-0300 INFO JSON-RPC service started and listening on 0.0.0.0:8545
observer-1 | 2025-11-18T17:01:22.670-0300 INFO Starting auto transaction log bloom caching service.
observer-1 | 2025-11-18T17:01:22.671-0300 INFO Lookup cache metadata file in data directory: /var/lib/besu/caches
observer-1 | 2025-11-18T17:01:22.681-0300 INFO Starting Ethereum main loop ...
observer-1 | 2025-11-18T17:01:22.681-0300 INFO Starting docker NAT manager.
observer-1 | 2025-11-18T17:01:22.686-0300 INFO Starting Network.
observer-1 | 2025-11-18T17:01:22.697-0300 INFO P2P RLPx agent started and listening on [/0:0:0:0:0:0:0:0]:30303.
observer-1 | 2025-11-18T17:01:22.698-0300 INFO Starting peer discovery agent on host=0.0.0.0, port=30303. IPv6 available.
observer-1 | 2025-11-18T17:01:22.710-0300 INFO Started peer discovery agent successfully, on effective host=0:0:0:0:0:0:0:0 and port=30303
observer-1 | 2025-11-18T17:01:22.711-0300 INFO P2P peer discovery agent started and listening on [/0:0:0:0:0:0:0:0]:30303
observer-1 | 2025-11-18T17:01:22.760-0300 INFO Writing node record to disk. NodeRecord{seq=1, publicKey=0x03e90cfde3fa63870330fe260e741814db50545d97bfc76318179534b339b3d938, udpAddress=Optional[/127.0.0.1:30303], tcpAddress=Optional[/127.0.0.1:30303], udp6Address=Optional.empty, tcp6Address=Optional.empty, asBase64=-3e4QMf0lY0t-X3cW7CvUvd106a0DukJ63JbL233KcK2BsvydL3JJ3XdlEPMK0CjRHWtLF096C8slY08EHmua8ck2EBg2V8aMfGhKYg2-iAgnLqgnV8gmLwhH8AAAGJc2Vjc011Nmsx0QpDP2j-m0HAZD-Jp50CBTbuFRdl7.HYxgX1Ts0bPZOIN0Y3CCdL-DdWnWgnZf, nodeId=0xde833181f05f182aac9cd98bb3ff0114calcu3e238a1697dde8cfb087c6e79c, customFields={tcp=30303, udp=30303, eth=[[0xa620dbe8, 0x]], id=V4, ip=0x7f080001, secp256k1=0x03e90cfde3fa63870330fe260e741814db50545d97bfc76318179534b339b3d938}}
observer-1 | 2025-11-18T17:01:22.791-0300 INFO Enode URL enode://e90cfde3fa63870330fe260e741814db50545d97bfc76318179534b339b3d938771e06151e9090530bdf9b969ab27712310325ff7fab3254d226b2c42a4bc9690127.0.0.1:30303
observer-1 | 2025-11-18T17:01:22.791-0300 INFO Node address 0x03e90cfde3fa63870330fe260e741814db50545d97bfc76318179534b339b3d938
observer-1 | 2025-11-18T17:01:22.800-0300 INFO Supported capabilities: [eth/68, eth/69], [Istanbul/100], [snap/1]
observer-1 | 2025-11-18T17:01:22.801-0300 INFO Starting synchronizer.
observer-1 | 2025-11-18T17:01:22.806-0300 INFO Enabling transaction handling following initial sync
observer-1 | 2025-11-18T17:01:22.806-0300 INFO Starting BFT mining coordinator following initial sync
observer-1 | 2025-11-18T17:01:22.814-0300 INFO Starting full sync.
observer-1 | 2025-11-18T17:01:22.815-0300 INFO Unable to find sync target. Waiting for 5 peers minimum. Currently checking 0 peers for usefulness
observer-1 | 2025-11-18T17:01:22.818-0300 INFO Ethereum main loop is up.

```

Fonte: (O Autor, 2025)

6 CONCLUSÃO E TRABALHOS FUTUROS

Para a realização deste trabalho, foram necessárias diversas etapas. Primeiramente, foi necessário um estudo técnico aprofundado, de forma a construir o conhecimento necessário sobre a tecnologia da *blockchain*. Após esse contato inicial com o tema, foi feita uma pesquisa sobre o estado da arte na área de usos de *blockchain* por governos ao redor do mundo, visto que a rede a ser analisada se trata de uma rede do governo brasileiro.

Depois de obter o conhecimento necessário e entender o cenário atual em outros países, foi possível estudar a RBB em mais detalhes e entender o seu funcionamento e casos de uso. A partir desse estudo, foi possível entender como funcionava a implantação do nó observador, que este trabalho se propôs a fazer, e quais os principais contratos inteligentes da rede, que este trabalho se propôs a analisar.

A partir dessa etapa teórica, foi construída a parte prática do trabalho. Nessa parte, foi feita a tentativa de implantação do nó observador da rede *blockchain* na rede do INF, porém sem sucesso. Também foi feita a análise técnica dos principais contratos inteligentes que essa rede possui, identificados na etapa teórica. Além disso, foi criado um canal de comunicação com as entidades oficiais responsáveis por gerir a RBB. A partir desse canal, foram obtidas as informações de endereço dos participantes da rede, expostas na Tabela 4.1.

A partir do conteúdo apresentado neste trabalho de conclusão de curso, é possível afirmar que a tecnologia da *blockchain*, se usada da maneira correta, traz diversos benefícios e facilidades que tecnologias tradicionais não possuem. Como, por exemplo, a imutabilidade e segurança que uma rede *blockchain* possui.

Redes que possuem suporte à implantação de contratos inteligentes podem até mesmo substituir aplicações convencionais integralmente, dado a sua característica de completude de Turing. Além disso, a segurança para o usuário aumenta ainda mais, uma vez que os contratos armazenados, de forma imutável, na rede, podem ser verificados, confirmando o comportamento esperado da aplicação.

A criação e existência da Rede Brasil *Blockchain*, analisada neste trabalho, demonstra o avanço que tivemos no uso e adoção da tecnologia da *blockchain*, chegando até mesmo no espectro público. A RBB pode ser o passo inicial para a adoção dessa tecnologia em outros setores governamentais, como, por exemplo, para registro de propriedades, como foi feito pelo governo da Geórgia, ou até mesmo a criação de uma moeda digital

brasileira, como fez o governo Chinês.

As ferramentas que a RBB disponibiliza para os usuários também são de extrema relevância. A possibilidade de implantar um nó observador, podendo ter acesso a todas as ações realizadas e informações armazenadas na rede *blockchain* tornam a rede auditável pela população. Além disso, o uso de um explorador de blocos, possibilitando uma análise visual, de forma a garantir que mesmo usuários que não possuam histórico técnico consigam compreender o fluxo de informações, entendendo a importância e os benefícios que essa tecnologia traz para a sociedade. Cabe ressaltar que a RBB ainda é um projeto piloto e em andamento, servindo de base para a adoção da tecnologia da *blockchain* no espectro governamental no Brasil. A evolução da rede pode ser atestada pela existência de novas gerações de contratos inteligentes de permissionamento, como foi o caso com a *gen01* e *gen02*. Além disso, com a sua evolução, a RBB lançou um *white-paper* da rede (TCU, 2025), assim como foi feito para redes *blockchain* já consolidadas, como foi o caso da *Ethereum*. É válido mencionar que esse *white-paper* da RBB não foi utilizado como fonte deste trabalho, pois o seu lançamento foi no dia 26 de novembro de 2025.

A partir deste trabalho, muitas possibilidades foram criadas para possíveis novos trabalhos. Como forma de complemento a este trabalho, seria necessário realizar uma investigação técnica para entender o motivo dos problemas que levaram ao insucesso da implantação do nó observador na rede do INF, de forma a corrigir os problemas e possibilitar essa implantação no futuro. Além disso, como foi visto, a verificação dos contratos não foi realizada com sucesso utilizando o explorador de blocos *Blockscout*. Então, caso o nó observador possa ser devidamente implantado na rede do INF, seria de interesse da comunidade que o explorador de blocos utilizado fosse outro, que permitisse a validação dos contratos inteligentes da rede.

Para além do escopo específico deste trabalho, em relação à RBB, existe um leque de possibilidades com a própria tecnologia da *blockchain*. Seria de interesse da comunidade a condução de um estudo acerca da segurança que contratos inteligentes possuem quando lidam com dados sensíveis das pessoas, e como aumentá-la. De forma a contribuir com a expansão do uso da *blockchain* até mesmo no âmbito governamental.

REFERÊNCIAS

- ALEXANDRA, L. Data breaches in cloud environments causes, consequences, and prevention. **ResearchGate**, 2025.
- ALONSO, S. L. N. et al. Open-access el salvador: an analysis of the monetary integration law and the bitcoin law. **Brazilian Journal of Political Economy**, 2024.
- ALONSO, S. L. N.; JORGE-Vázquez, P. A. R. J.; HERNÁNDEZ, B. M. S. Gender gap in the ownership and use of cryptocurrencies: Empirical evidence from Spain. **Journal of Open Innovation: Technology, Market, and Complexity**, 2023.
- BELOTTI, M. et al. A vademecum on blockchain technologies: When, which and how. **IEEE Communications Surveys Tutorials**, 07 2019.
- Binance. **Preço do Bitcoin (BTC/BRL)**. 2025. Acessado em: 9 nov. 2025. Available from Internet: <<https://www.binance.com/pt-BR/price/bitcoin/BRL>>.
- BLOCKCHAIN, R. B. **Explorador de blocos da RBB**. 2025. <<https://rbbobserver.share.zrok.io/>>. Acessado em: 20 nov. 2025.
- BLOCKCHAIN, R. B. **RBB**. 2025. <<https://github.com/RBBNet/rbb>>. Acessado em: 9 nov. 2025.
- BLOCKCHAIN, R. B. **RBB Permissionamento**. 2025. <<https://github.com/RBBNet/Permissionamento>>. Acessado em: 9 nov. 2025.
- BLOCKCHAIN, R. B. **Roteiro Acesso Leitura RBB**. 2025. <https://github.com/RBBNet/rbb/blob/master/roteiro_acesso_leitura.md>. Acessado em: 20 nov. 2025.
- Blockchain.com. **Bloco Gênese do Bitcoin**. 2025. Acessado em: 9 nov. 2025. Available from Internet: <<https://www.blockchain.com/explorer/blocks/btc/0>>.
- BROOKE, S. **China Makes Cryptocurrency Transactions Illegal: An Explainer**. 2021. <<https://lookup.icann.org/en/lookup>>.
- BUTERIN, V. **Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform**. 2013. <<https://ethereum.org/en/whitepaper/>>.
- CENTRAL, B. **Drex – Real Digital**. 2025. <<https://www.bcb.gov.br/estabilidadefinanceira/drex>>.
- CHEN, I. O. N. J. Central bank digital currencies: Digital yuan and its role in Chinese digital economy development. **RUDN Journal of Economics**, 2023.
- Coinbase. **O que é uma Stablecoin?** 2025. Acessado em: 9 nov. 2025. Available from Internet: <<https://www.coinbase.com/pt-br/learn/crypto-basics/what-is-a-stablecoin>>.
- CoinDesk. **Criptomoedas inflacionárias e deflacionárias: qual é a diferença?** 2025. Acessado em: 9 nov. 2025. Available from Internet: <<https://www.coindesk.com/pt-br/learn/inflationary-and-deflationary-cryptocurrencies-whats-the-difference>>.
- COMPARE, O. D. **Comparação dos bytécodes**. 2025. <<https://string.is/diff-compare>>. Acessado em: 22 nov. 2025.

DUAN, Y.; WANG, X. et al. Blockchain-based e-voting systems: Opportunities and challenges. **Digital Government: Research and Practice**, 2024. ISSN 2666-6030. Acessado em: 9 nov. 2025. Available from Internet: <<https://www.sciencedirect.com/science/article/pii/S2666603024000046>>.

Ethereum Foundation. **Gas and Fees**. 2025. Acessado em: 9 nov. 2025. Available from Internet: <<https://ethereum.org/developers/docs/gas/>>.

Ethereum Foundation. **Smart Contracts**. 2025. Acessado em: 9 nov. 2025. Available from Internet: <<https://ethereum.org/developers/docs/smart-contracts/>>.

European Commission. **What is EBSI (European Blockchain Services Infrastructure)**. 2025. Acessado em: 9 nov. 2025. Available from Internet: <<https://ec.europa.eu/digital-building-blocks/sites/spaces/EBSI/pages/590447955/What+is+EBSI>>.

EUROPEIA, U. **68areas by 2050, says UN**. 2018.

EXAME. **Bitcoin Pizza Day: a história da refeição mais cara de todos os tempos**. 2023. Acessado em: 9 nov. 2025. Available from Internet: <<https://exame.com/future-of-money/criptoativos/bitcoin-pizza-day-a-historia-da-refeicao-mais-cara-de-todos-os-tempos/>>.

FAN, C. et al. Performance analysis of hyperledger besu in private blockchain. In: **2022 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS)**. [S.l.: s.n.], 2022. p. 64–73.

GITHUB. **Contract verified only partially not fully while using flatten method 772**. 2025. <<https://github.com/blockscout/blockscout-rs/issues/772>>. Acessado em: 22 nov. 2025.

HJÁL MARSSON, F. et al. Blockchain-based e-voting system. In: **2018 IEEE 11th International Conference on Cloud Computing (CLOUD)**. [S.l.: s.n.], 2018.

Hyperledger Caliper Community. **Hyperledger Caliper Documentation**. 2025. Acessado em: 9 nov. 2025. Available from Internet: <<https://caliper-doc-trial.readthedocs.io/en/latest/>>.

IDE, R. **Remix IDE**. 2025. <<https://remix.ethereum.org/#lang=en&optimize&runs=200&evmVersion&version=soljson-v0.8.30+commit.73712a01.js>>. Acessado em: 22 nov. 2025.

InfoMoney. **Yuan Digital: a moeda digital da China que o mundo admira e teme**. 2023. Acessado em: 9 nov. 2025. Available from Internet: <<https://www.infomoney.com.br/mercados/yuan-digital-a-moeda-digital-da-china-que-o-mundo-admira-e-teme/>>.

Internet Corporation for Assigned Names and Numbers (ICANN). **Domain Information on bitcoin.org**. 2025. <<https://lookup.icann.org/en/lookup>>.

JOHNSON, D.; MENEZES, A.; VANSTONE, S. The elliptic curve digital signature algorithm (ecdsa). **International Journal of Information Security**, v. 1, n. 1, p. 36–63, 2001.

KACZOROWSKA, M. Blockchain-based land registration: Possibilities and challenges. **Masaryk University Journal of Law and Technology**, 2019.

KEYCDN. **SHA1 vs SHA256**. 2023. <<https://www.keycdn.com/support/sha1-vs-sha256>>. Acessado em: 23 nov. 2025.

KSHETRI, N.; VOAS, J. Blockchain-enabled e-voting. **IEEE Software**, 2018.

LAMPORT, L.; SHOSTAK, R.; PEASE, M. The byzantine generals problem. In: _____. **Concurrency: The Works of Leslie Lamport**. [S.l.]: Association for Computing Machinery, 2019.

LAZUASHVILI, N.; NORTA, A.; DRAHEIM, D. Integration of blockchain technology into a land registration system for immutable traceability: A case study of georgia. In: **Business Process Management: Blockchain and Central and Eastern Europe Forum**. [S.l.]: Springer International Publishing, 2019.

MOFOKENG, S. M. . D. K. M. T. E. Bitcoin adoption in online payments: examining consumer intentions and word-of-mouth recommendations. **Future Business Journal**, 2024.

MOHANTY, D. **Ethereum for Architects and Developers**. [S.l.]: Apress, 2018.

NAIR, V. **What is Decentralised Digital Identity?** 2025.

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

Ripio. **LaChain: uma blockchain focada 100% na América Latina**. 2024. Acessado em: 9 nov. 2025. Available from Internet: <<https://launchpad-br.ripio.com/blog/lachain-a-blockchain-focada-100-na-america-latina>>.

SCHEID, E. J. **An Intent-based Blockchain-agnostic Interaction Environment**. Thesis (PhD) — University of Zurich UZH, 2022.

SCHEID, E. J. et al. Blockchains and Distributed Ledgers Uncovered: Clarifications, Achievements, and Open Issues. In: **Advancing Research in Information and Communication Technology**. Cham, Switzerland: Springer, 2021, (IFIP AICT Festschrifts, v. 600). p. 289–317.

SCHMAGER, S. et al. What do citizens think of ai adoption in public services? exploratory research on citizen attitudes through a social contract lens. 2023.

SOLIDITY, D. **Versões compilador Solidity**. 2025. <<https://etherscan.io/solcversions>>. Acessado em: 20 nov. 2025.

Solidity Documentation. **Types — Solidity**. 2025. <<https://docs.soliditylang.org/en/latest/types.html>>. Acessado em: 18 nov. 2025.

SPECTER, M. A.; KOPPEL, J.; WEITZNER, D. The ballot is busted before the blockchain: a security analysis of voatz, the first internet voting application used in u.s. federal elections. In: **Proceedings of the 29th USENIX Conference on Security Symposium**. [S.l.: s.n.], 2020.

SWAN, M. **Blockchain: Blueprint for a New Economy**. [S.l.]: O'Reilly Media, 2015.

TAHERDOOST, H. Smart contracts in blockchain technology: A critical review. **Information**, 02 2023.

TCU. **RBB – Rede Blockchain Brasil – Infraestrutura e Governança para execução de aplicações de interesse público**. [S.l.], 2025. Cartilha / Manual / Tutorial. Available from Internet: <<https://portal.tcu.gov.br/publicacoes-institucionais/cartilha,%20manual%20ou%20tutorial/rbb-white-paper>>.

THAKUR M.N. DOJA, Y. K. D. T. A. G. K. V. Land records on blockchain for implementation of land titling in india. **International Journal of Information Management**, 2020.

TOOL, S. D. **Comparação dos bytecodes por caracter**. 2025. <<https://stringdiff.com/>>. Acessado em: 27 nov. 2025.

WOOD, G. Ethereum: A secure decentralised generalised transaction ledger. In: SWAN, M. (Ed.). **Blockchain: Blueprint for a New Economy**. Springer, 2018. Acessado em: 9 nov. 2025. Available from Internet: <https://link.springer.com/chapter/10.1007/978-3-319-75650-9_14>.

WOOD, G. Ethereum virtual machine (evm). In: SWAN, M. (Ed.). **Blockchain: Blueprint for a New Economy**. Springer, 2018. Acessado em: 9 nov. 2025. Available from Internet: <https://link.springer.com/chapter/10.1007/978-3-319-75650-9_14>.

YOUNG, A.; VERHULST, S. **Self Sovereign Identity for Government Services in Zug, Switzerland**. 2018.

ZARIR, A. et al. Developing cost-effective blockchain-powered applications: A case study of the gas usage of smart contract transactions in the ethereum blockchain platform. **ACM Transactions on Software Engineering and Methodology**, 10 2020.