NeoCompiler Eco: experimentação de consenso em blockchain e contratos inteligentes

Igor M. Coelho[1], **Vitor N. Coelho**[2]

[1] Instituto de Computação – Universidade Federal Fluminense (UFF)
Av. Gal. Milton Tavares de Souza, s/n - São Domingos, Niterói, RJ, 24210-346 – Brasil

[2]OptBlocks Consultoria LTDA
Belo Horizonte, MG – Brasil

imcoelho@ic.uff.br, vitor@optblocks.com

Abstract. *This paper presents the NeoCompiler Eco platform, focused on development and experimentation of smart contracts, so as consensus systems for blockchain. The NeoCompiler is a free project started in 2017, already established online (in production) with global scale users. It consists of several modules, including: support for compiling Turing-complete smart contracts written in popular languages into blockchain bytecode; testing and deploy of contracts in a public shared temporary network, with global reach; auxiliary and didactic tools for type conversion and internal blockchain explorer for visualization of transactions and blocks; interactive chat for questions between global participants; interaction and graphical visualization of consensus process. Among the achievements of the platform, we highlight: support for courses in blockchain topic; support for hackathons in Brazil, Latin America and worldwide; support for experimentation network "Nosso DLT" and development of innovative consensus algorithms, such as the dBFT 2.0 launched in 2019.*

Resumo. *Este artigo apresenta a plataforma NeoCompiler Eco, com foco em desenvolvimento e experimentação para contratos inteligentes, bem como sistemas de consenso para blockchain. O NeoCompiler é um projeto livre iniciado em 2017, já estabelecido online (em produção) e com usuários em escala global. Ele consiste em diversos módulos, dentre eles: suporte a compilação de contratos inteligentes Turing-completos escritos em linguagens populares para o bytecode da blockchain; testes e implantação de contratos em uma rede pública compartilhada temporária e de alcance global; ferramentas didáticas auxiliares para conversões de tipos e blockchain explorer interno para visualização de transações e blocos; chat interativo para dúvidas entre participantes (globais); interação e visualização gráfica do processo de consenso. Dentre as realizações da plataforma, ressaltamos: apoio a disciplinas no tópico de blockchain, realização de hackathons no Brasil, América Latina e mundiais; apoio à rede experimental "Nosso DLT" e desenvolvimento de algoritmos de consenso inovadores, como o dBFT 2.0 lançado em 2019.*

1. Introdução

A tecnologia blockchain surgiu com a criptomoeda Bitcoin [Nakamoto et al. 2008] em 2008, como uma maneira de contabilizar suas operações em um livro-razão público e

imutável. Nos anos seguintes, sugiram diversas outras tecnologias inspiradas no mesmo princípio de encadeamento de blocos de transações, sendo estas capazes de suportar uma variedade maior de operações computacionais, como o Ethereum [Buterin 2015] e Neo [Hongfei, Da and Zhang, Erik 2015]. Os códigos executados nessas redes, denominados contratos inteligentes, trouxeram novas possibilidades de interação segura entre partes interessadas, através de regras transparentes e verificáveis, mas ao mesmo tempo trouxeram novos desafios de desenvolvimento e experimentação.

Dentre os desafios atuais em blockchain, elencamos: mecanismos de gestão e governança da rede (incluindo algoritmos de consenso); mecanismos de execução de código e verificação das informações armazenadas na blockchain; ambientes e ferramentas de desenvolvimento de software como foco em contratos inteligentes. As diversas redes blockchain de acesso público se diferenciam na maneira como lidam com esses (e outros) desafios. Para lidar com alguns desses desafios, o projeto NeoCompiler Eco surgiu em 2017 como um ambiente online para desenvolvimento de contratos inteligentes e experimentação de algoritmos de consenso.

A motivação inicial do projeto se deve à dificuldade de oferecer ambientes confiáveis de desenvolvimento de contratos em múltiplas linguagens de programação na Neo Blockchain, como C#, Python e Java, que tipicamente exigem pacotes diferentes de acordo com o sistema operacional do usuário, causando diversos conflitos de configuração. Para se assegurar a corretude dos contratos compilados em bytecode, também são necessárias ferramentas de inspeção direta nos códigos de máquina gerados, dado que erros dos compiladores podem ser perpetuados como falhas em um contrato imutável implantado em uma rede. Logo a seguir, o protótipo do NeoCompiler foi estendido ao longo de 2018-2020 para incluir demais aspectos de experimentação em blockchain, como uma rede privada compartilhada para testes e implantação dos contratos compilados, bem como recursos de monitoramento dos nós de consenso dessa rede compartilhada. A Figura 1 apresenta a tela principal da plataforma NeoCompiler Eco, para protocolo da Neo Blockchain v2.

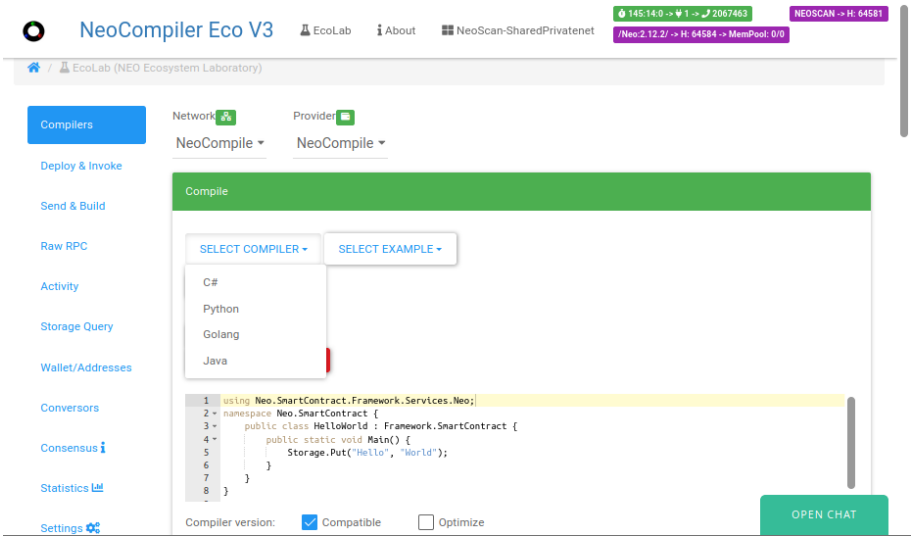

Figura 1. Frontend da plataforma NeoCompiler Eco para protocolo NEO 2.x

A plataforma NeoCompiler Eco foi desenvolvida de acordo com padrões da Neo

Blockchain[1] versão 2.x, uma blockchain Turing-completa idealizada em 2015 por Erik Zhang e Da Hongfei. Sua estrutura se assemelha à blockchain Ethereum, bastante popular para desenvolvimento de contratos inteligentes, também Turing-completa. Dentre as diferenças, ressaltamos: o Neo permite desenvolvimento de contratos inteligentes em linguagens populares como C#, Python, Java e Go, enquanto o Ethereum demanda compiladores para linguagens de domínio específico, como o Solidity; a rede do Ethereum é massivamente descentralizada, composta por milhares de nós que trabalham competitivamente através de prova de trabalho (do inglês, proof-of-work), enquanto o Neo se utiliza de consenso bizantino inspirado em PBFT [Castro and Liskov 1999] com um número reduzido de nós eleitos pelo token da rede. Ambas redes se utilizam do conceito de *GAS* como forma de pagar pelo gasto computacional das operações com contratos inteligentes, sendo esse *GAS* disponibilizado gratuitamente para todos os usuários da plataforma de experimentação NeoCompiler Eco.

Como um projeto de filosofia open-source, naturalmente já existem projetos alternativos inspirados no projeto original [Araújo, Rodolfo 2019] com o intuito de reutilizar a mesma interface em outros protocolos de blockchain, como o EOS [Grigg 2017]. Por essas características, a plataforma NeoCompiler Eco tem sido utilizada em universidades brasileiras para o ensino de blockchain, bem como no estabelecimento de uma rede experimental blockchain com escopo no Estado do Rio de Janeiro, suportada pelo Grupo de Pesquisa em Algoritmos e Logística Descentralizada (ALODE) [Coelho 2020].

Esse trabalho é pioneiro em publicar conteúdo sobre a plataforma do NeoCompiler Eco aqui abordada, bem como contribuir para o ecossistema da tecnologia do Neo, que tem pouco publicado. Além disso, existem poucas abordagens de redes nacionais com foco em experimentação blockchain.

Este artigo é organizado em cinco seções, incluindo esta introdução. A Seção 2 apresenta os módulos da plataforma NeoCompiler Eco, bem como aspectos básicos de configuração e interatividade da plataforma. Na Seção 3 são detalhados aspectos técnicos de experimentação de contratos em uma rede compartilhada de testes, bem como para experimentação de novos algoritmos de consenso da família dBFT. Na Seção 4 é abordado o processo de gestão da rede compartilhada de experimentação, bem como desafios de iniciativas brasileiras que utilizam o projeto, como o "Nosso DLT". Finalmente, a Seção 5 conclui o trabalho e apresenta perspectivas futuras para o projeto.

2. Módulos Principais do NeoCompiler Eco

O NeoCompiler Eco consiste em um ambiente de experimentação que pode ser obtido e replicado de forma livre (sob licença MIT). A página do projeto no GitHub é `https://github.com/neoresearch/neocompiler-eco`, que inclui informações de como instalar localmente no computador, através da tecnologia de micro-serviços Docker[2]. Para sistemas operacionais baseados em GNU/Linux, um script denominado `build_everything.sh` cuida da instalação de toda a plataforma e já coloca no ar (localmente) todos os serviços disponíveis. Uma versão de produção da plataforma para o protocolo da Neo Blockchain v2 pode ser acessada no site `https://neocompiler.`

[1] https://neo.org

[2] docker-ce, com guia disponível em `https://docs.docker.com/engine/install/ubuntu/`

io/neo2.

A plataforma é dividida em três módulos principais: frontend para interação; backend para serviços de compilação de contratos inteligentes; backend para gestão da rede de experimentação.

2.1. Frontend de interação

O frontend da plataforma é desenvolvido no formato web através de html/css/javascript, sendo uma maneira imediata de interação com todos os componentes disponíveis. Projetos implantados na rede não necessitam de uso do frontend, sendo todas as operações igualmente disponibilizadas via RPC.

Dentre os aspectos fundamentais do frontend, listamos: menu de topo (com informações da rede em operação); menu lateral com seções (ou padrão "hamburguer" em smartphones); tela principal com o conteúdo da seção atual. A seguir são apresentadas as seções:

2.1.1. Menu *Compilers*

O menu *compilers* apresenta uma caixa onde o usuário pode codificar seu contrato inteligente, ou carregar algum exemplo prático (como *tokens* padronizados). É possível escolher uma linguagem compatível com a rede e *frameworks* de desenvolvimento disponíveis (no caso do Neo v2, é disponibilizado suporte a C#, Python, Go e Java). O resultado da compilação é apresentado como uma sequência de bytes (em hexadecimais), que representam as instruções de máquina executadas pelo contrato inteligente correspondente (para a máquina virtual *NeoVM*[3], no caso). A Figura 2 ilustra esse processo.

Vale ressaltar que cada compilador é suportado por uma comunidade diferente, e frequentemente com foco em um sistema operacional diferente. Por exemplo, o compilador de C# para Neo é originalmente distribuído para Windows, tornando-se assim possível através do NeoCompiler Eco que usuários de demais sistemas se beneficiem da linguagem de programação que for preferida. Nota-se também que grande parte dos projetos de contratos inteligentes consistem de códigos bastante enxutos, como no desenvolvimento de *tokens* digitais, portanto a barreira removida por um compilador online se torna fundamental para atração e treinamento de novos usuários.

2.1.2. Menu *Deploy & Invoke*

O menu *Deploy & Invoke* permite implantar o contrato compilado na rede alvo do sistema. Por padrão, a rede alvo busca um endereço local do sistema, e caso não seja encontrada, coloca como alvo a rede compartilhada do `https://neocompiler.io`. O *GAS* consumido nas operações de implantação e chamadas de métodos do contrato é deduzido do endereço disponibilizado no Menu *Wallet/Address*.

[3]O projeto da *NeoVM* é hospedado em: `https://github.com/neo-project/neo-vm`

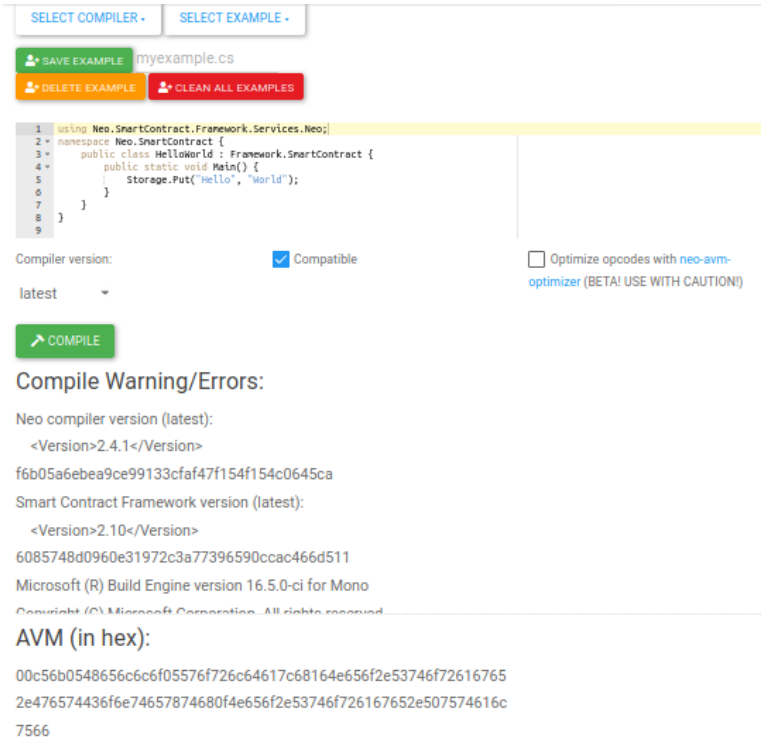

Figura 2. Compilação do "Hello World" em C# para NeoVM

2.1.3. Menu *Send & Build*

O menu *Send & Build* permite construir transações manualmente e enviá-las para a rede. Não é necessário usar o *Send & Build* durante o passo de *Deploy & Invoke*, pois o mesmo já é feito automaticamente (esse menu é utilizado para experimentação de tipos de transação mais avançados, bem como para anexação de *flags* e campos extras na transação).

2.1.4. Menu *Raw RPC*

O menu *Raw RPC* permite invocar operações de RPC da rede blockchain subjacente. Por exemplo, pode-se consultar o bloco mais recente anexado à blockchain, ou dados de uma determinada transação previamente incluída em bloco.

2.1.5. Menu *Activity*

O menu *Activity* permite visualizar operações recentes do mesmo usuário, bem como repetí-las (útil para testes nas operações de contratos inteligentes).

2.1.6. Menu *Storage Query*

O menu *Storage Query* permite consulta ao armazenamento de dados da blockchain. Por exemplo, após a execução do "Hello World" (apresentado anteriormente), pode-se veri-

ficar que o valor “World” foi atribuído à chave “Hello”, do respectivo contrato implantado. Para fazer a consulta, basta utilizar o identificador único apresentado na fase de compilação do contrato, denominado *scripthash*.

2.1.7. Menu *Wallet/Addresses*

O menu *Wallet/Addresses* apresenta os fundos em criptoativos e *tokens* de cada endereço registrado na plataforma. Vale ressaltar que diversos endereços de uso geral são oferecidos para fins de experimentação, portanto não existe valor “real” em nenhum deles (dado que o frontend se conecta à rede de experimentação, por padrão). É possível utilizar esses mecanismos para gestão de fundos “reais”, se conectando a uma rede piloto ou de produção, mas esse processo não é recomendado por razões de segurança (existem carteiras especializadas para redes em produção).

2.1.8. Menu *Conversors*

O menu *Conversors* apresenta ferramentas úteis relacionadas ao desenvolvimento de contratos inteligentes. Por exemplo, pode-se praticar o uso de algoritmos criptográficos como: SHA-256, RIPEMD-160, ECDSA secp256r1, etc.

2.1.9. Menu *Consensus*

No menu *Consensus* pode-se visualizar informações de operação dos nós de consenso da rede de experimentação. Esses dados tem sido utilizados juntamente com a ferramenta de visualização da plataforma[4] para encontrar falhas e aprimoramentos no algoritmo dBFT, sendo cruciais para o lançamento, em 2019, do algoritmo dBFT 2.0.

2.1.10. Menu *Statistics*

O menu *Statistics* apresenta estatísticas da rede, bem como tempos médios de geração de blocos e atrasos na rede.

2.1.11. Menu *Settings*

O menu *Settings* permite a configuração geral do frontend, incluindo a troca de rede de experimentação (local, compartilhada, piloto ou produção) e backends de serviço de compilação.

2.2. Backend de serviços de compilação

Os serviços de compilação são oferecidos de forma independente, tipicamente encapsulados através de *containers* de micro-serviços dockerizados. Diversas versões são oferecidas para cada tipo de compilador, com variações na linguagem de programação e

[4] `https://github.com/NeoResearch/consensus-draw`

frameworks disponibilizados. Para o protocolo do Neo Blockchain v2, dois compiladores são mais populares: compilador C# *neon* desenvolvido pela Neo Foundation[5]; e o compilador Python *boa* desenvolvido pelo grupo City of Zion[6].

Para prover balanceamento de carga, diversas máquinas físicas oferecem o serviço de compilação, sendo hospedadas em nuvens comerciais. Isso é necessário devido ao custo computacional do processo de compilação dos contratos (tipicamente alguns segundos), sendo que toda oferta de serviços da plataforma NeoCompiler são feitos de forma gratuita (mantidos por comunidades livres financiadas por doações). A plataforma já foi utilizada com sucesso em diversos hackathons na América Latina (durante 2018 e 2019) e hackathons mundiais na tecnologia do Neo, tendo o serviço demonstrado a resiliência necessária para lidar com dezenas de usuários simultâneos.

2.3. Backend da rede de experimentação

O NeoCompiler Eco introduziu o conceito de *shared privatenet*, ou seja, uma rede de natureza *privada* e *não permanente*, mas ao mesmo tempo, cujo acesso externo é permitido através de *endpoints* da plataforma. Sendo assim, os dados da blockchain de experimentação são considerados compartilhados entre todos participantes, e são removidos periodicamente para reinício da rede. A rede de experimentação da plataforma NeoCompiler é chamada *Eco*, dando o nome completo do projeto.

Essa estratégia traz duas vantagens principais. A manutenção de uma blockchain piloto ou produção traz consigo consideráveis custos de manutenção e de segurança, que dificultam a oferta de um serviço rápido e gratuito para experimentação. A evolução da rede, como novos algoritmos de consenso, pode ser introduzida rapidamente entre os ciclos de reinício da rede, o que foi fundamental para o desenvolvimento do algoritmo de consenso dBFT 2.0 (veja mais na Seção 3).

Por outro lado, usuários com uma rede local tem maior flexibilidade para gestão dos dados da rede de testes, enquanto existe uma dificuldade inerente de segurança em expor sua rede para efetuar testes de escala global (o que já está pronto para uso na rede de experimentação *Eco*). Desta maneira, embora se busque estabilidade na oferta do serviço, existem reinícios planejados (para evolução ou limpeza) que podem ser visualizados por qualquer usuário através de um relógio no topo do frontend, informando o tempo de vida restante da rede (Figura 3).

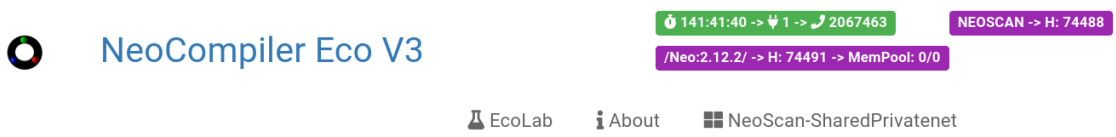

Figura 3. Tempo de 141 horas, 41 minutos e 40 segundos (aprox. 6 dias) restantes para o próximo reinício da rede

3. Experimentação de Contratos e de Algoritmos de Consenso

A plataforma NeoCompiler Eco permite a experimentação de contratos inteligentes, bem como algoritmos de consenso inovadores. Na perspectiva de desenvolvedores de con-

[5] https://github.com/neo-project/neo-devpack-dotnet
[6] https://github.com/CityOfZion/neo-boa

tratos, como visto na seção anterior, a plataforma atua como um Ambiente de Desenvolvimento Integrado (no inglês, *IDE*) oferecido online. Tal estratégia permite que usuários interajam com uma blockchain em operação e aprendam fundamentos básicos de programação e teste, mesmo a partir de um smartphone (ou sem necessitar instalar nada no computador). Para experimentos mais aprofundados, o usuário pode baixar a plataforma e instalar em seu computador, tendo assim exclusividade de uso sobre os recursos da plataforma, ao invés de compartilhar o uso de forma global (uso completamente *offline*). Quando *offline*, o usuário apenas perde acesso a recursos gerais como o *chat*, e a rede compartilhada de alcance global Eco, mas todos os demais serviços (inclusive as imagens de compilação para múltiplas linguagens) ficam disponibilizados em seu computador.

A estratégia *offline* é bastante útil para o desenvolvimento de novas técnicas de consenso, dado que basta substituir o módulo de rede e consenso por um novo, e utilizar normalmente os recursos de monitoramento, logs, e interação com a nova rede. O sistema de consenso padrão da plataforma é denominado dBFT 2.0, uma variante do algoritmo PBFT com três fases, adaptada para uso em uma blockchain. Vale ressaltar que, mesmo localmente, o sistema disponibiliza por padrão 4 nós de consenso, dentre eles no máximo 1 pode ficar indisponível simultaneamente para que se interrompa o processo de geração de blocos. Isso ocorre pois o sistema é baseado em falhas bizantinas, com número de nós $N = 3f + 1$, e f é o número máximo de nós falhos simultaneamente, e $2f + 1$ nós são necessários para atingir o *quorum* de consenso [Castro and Liskov 1999]. A plataforma NeoCompiler Eco pode lançar ainda mais processos (além dos 4 de consenso), dado que a comunicação entre os nós de consenso é feita por uma camada de peer-to-peer independente. Isso torna não-deterministico o processo de entrega de mensagens e comunicação, mesmo em uma rede local, sendo útil para a experimentação de cenários em que existe atraso ou perda de mensagens entre os nós[7]. Também é possível executar a rede com um único nó de consenso, sem nenhum outro na camada de redistribuição peer-to-peer, tornando a rede completamente suscetível a falhas bizantinas, porém reduzindo a sobrecarga de rede dos demais nós de consenso. Essa ferramenta tem possibilitado o desenvolvimento de trabalhos de cunho acadêmico na temática de consensos inovadores para blockchain, como [Coelho et al. 2020, Araujo et al. 2019].

4. Desafios de Governança da Rede de Experimentação

Conceito de *shared privatenet* foi introduzido na Seção 2.3, e permite que o NeoCompiler Eco ofereça um serviço de experimentação estável, atualizado, porém ao custo da não-retenção permanente dos dados. Existem, porém, outros desafios relacionados ao uso compartilhado da rede.

No contexto de *testbeds* para Internet do Futuro, destacamos o projeto *Future Internet Brazilian Environment for Experimentation* (FIBRE), construído no escopo de uma parceria Brasil-Europa em 2010 [Ciuffo et al. 2016]. Através da utilização de infraestrutura da "FIBREnet", formada acima do backbone da Rede Nacional de Ensino e Pesquisa [RNP 2021], são disponibilizadas 11 ilhas numa federação, com rede separada em uma camada de controle e outra para experimentação. Neste caso, a governança permite

[7]Como cada processo é independente e executado por um *container* de micro-serviço, é possível introduzir interrupções e atrasos diretamente na camada de rede que conecta esses serviços

a execução de operações em um escopo bem mais amplo do que blockchain, abarcando projetos gerais de Internet do Futuro. Por outro lado, a execução de códigos em uma rede confiável de natureza permanente ou semi-permanente, como na blockchain, traz maiores desafios na camada de governança. Em termos de infraestrutura computacional, a tecnologia blockchain se utiliza de protocolos consolidados como TCP/IP e estratégias de peer-to-peer para disseminação das informações, podendo assim usufruir da infraestrutura existente. Em nível de plataforma blockchain, a governança se dá através de acordos entre as entidades parceiras, através de um consórcio, no qual são oferecidos recursos computacionais e mão-de-obra qualificada para manutenção da rede.

No contexto de blockchain para a América Latina [Mendoza et al. 2019], pesquisadores destacam a importância de iniciativas como a Aliança Global LACChain [LACChain 2021], com potencial catalisador de novas iniciativas no ramo de blockchain e para o governo. Ainda nesse sentido, em 2018 foram discutidas estratégias para estabelecimento de uma rede blockchain latino-americana através da tecnologia do Neo [Coelho and Coelho 2018], sendo na época denominada "Rede Neo".

Já no contexto brasileiro, a perspectiva de uma rede interinstitucional de blockchain entre universidades no Estado do Rio de Janeiro foi publicamente apresentada [Costa et al. 2019] pelo proponente da rede "Nosso DLT", em operação desde Outubro/2019.

4.1. Caso de Uso: Nosso DLT

A rede "Nosso DLT" é uma rede experimental inicialmente hospedada em universidades no estado do rio de janeiro, e tem sido testada, para fins acadêmicos, utilizando a plataforma NeoCompiler Eco. Através de um piloto/produção desse tipo de rede, espera-se viabilizar operações acadêmicas em um "Campus Inteligente" (como intercâmbio de notas/boletins entre instituições de forma segura, emissão de diplomas, gestão de eventos e plataformas científicas descentralizadas, etc), bem como permitir que startups e parceiros tecnológicos hospedem serviços. Um endpoint hospedado em `http://redeneo1.ime.uerj.br:8000` e demais nós de operação estão visíveis na página do grupo ALODE [Coelho 2020], podendo ser usado de forma pública e gratuita. A rede é reiniciada de forma periódica, não se enquadrando como rede piloto ou produção, no momento da escrita deste artigo. A segurança da rede depende da resiliência de 66% das instituições participantes (de forma simultânea), devido às características do algoritmo de consenso dBFT, utilizado na rede (de forma inicial). De forma distinta da rede pública da Neo Blockchain, que opera por meio de votação (através de tokens NEO), o "Nosso DLT" é configurado diretamente entre as entidades participantes, sendo reservado o uso dos tokens NEO (internos) apenas para essa finalidade (o que trás dificuldades técnicas para a gestão interna desses tokens).

A forma de utilização desta rede é semelhante ao uso da plataforma global `https://neocompiler.io`, utilizando-se dos controles visuais descritos na Seção 2.1. Tal interface permite um monitoramento dos logs de operação dos nós do consenso dBFT, detectando possíveis anomalias e falhas (bem como possíveis ataques bizantinos). Também é possível monitorar os tokens nativos (NEO/GAS) através do blockchain explorer NeoScan acoplado na rede de testes. Observe que isso é válido tanto na rede pública, bem como uma instância local do sistema, como o da rede NossoDLT.

Em termos de operação de uma plataforma privada/permissionada, basta o usuário baixar o software do NeoCompiler Eco e executar o script de instalação. Este processo irá baixar automaticamente todas ferramentas necessárias, como sistema de consenso, compiladores de contratos inteligentes em diversas linguagens, tudo através de tecnologias de microsserviços (para garantir o isolamento e funcionalidade correta dos diversos componentes envolvidos). O usuário pode então definir o tempo estimado de geração de blocos, bem como demais detalhes de operação da rede. Para atualizar o algoritmo de consenso, basta atualizar a pasta *Consensus* com um novo código (na linguagem C#), e substituir o arquivo binário do(s) nó(s) de operação. Esse processo foi efetuado diversas vezes durante o desenvolvimento de uma nova versão do dBFT, denominada dBFT 2.0, atualmente em produção na rede pública da Neo Blockchain.

Dentre os diversos desafios para manutenção e operação de uma rede experimental com escopo local, elencamos: desafios técnicos relacionados a energia elétrica (quedas frequentes afetam a operação e estabilidade de componentes de rede fundamentais); dependência de pessoal técnico para manutenção do acesso de rede; dificuldade na obtenção de material para ensino da tecnologia em português (e até mesmo em inglês); atualização frequente de software devido a mudanças no protocolo utilizado pela rede (no caso, a Neo Blockchain); ausência de um conceito formal de "comitê gestor" no software blockchain utilizado (o que gera estratégias artificiais através do uso de tokens para controle da autoridade sobre a rede). Devido a esses desafios, esperava-se que em 2021 a rede já estivesse mais estável, com mais usuários e mais provedores de nó de consenso, porém atualmente apenas quatro nós encontram-se efetivamente configurados para testes (em duas localizações físicas distintas), com falhas frequentes de acesso. Ainda assim, diversos avanços tem sido conquistados, decorrentes da experiência de uso e colaboração ativa entre os participantes, sendo esperada uma expansão da rede em um futuro próximo.

Finalmente, a tecnologia Neo está em constante modificação, sendo previsto o lançamento de uma nova versão em 2021, denominada N3[8], com conceitos explícitos de "Comitê Gestor" e Armazenamento de Dados Distribuídos (NeoFS), que podem facilitar a gestão futura da rede experimental.

5. Conclusões e Perspectivas Futuras

Neste trabalho foi apresentada a plataforma NeoCompiler Eco, que disponibiliza acesso a serviços de compilação e testes de contratos inteligentes na tecnologia Neo Blockchain, bem como permite o gerenciamento de redes de consenso para uso compartilhado. Também foram apresentados os fundamentos do "Nosso DLT", um projeto de rede experimental blockchain, de caráter público permissionado, que se utiliza atualmente da plataforma NeoCompiler Eco para gerenciar seus nós de operação.

Com diversas melhorias propostas no projeto N3, espera-se que uma próxima versão da rede tenha maior facilidade na gestão do comitê participante do "Nosso DLT", dado que esse conceito é parte fundamental do protocolo Neo v3. Espera-se também oferecer demais funcionalidades em versão aprimorada do NeoCompiler Eco para N3.

[8]N3 é um acrônimo para NEO v3

Agradecimentos

Os autores agradecem às agências de fomento CAPES, FAPERJ e CNPq (bolsa PQ-2), bem como o apoio e equipamentos instalados no PPG-CComp/UERJ, LabIME/UERJ e LabIC/UFF. Em especial, os autores agradecem a Fabio Cardoso, Rodolfo Araújo e contribuidores anônimos do GitHub, e a Peter Lin, Grace Gui, Da Hongfei, Erik Zhang, bem como comunidades de desenvolvimento NeoResearch e Neo Foundation pelo apoio.

Referências

Araujo, R. P., Coelho, I. M., Ochi, L. S., and Coelho, V. N. (2019). Libbft: A high-performace timed automata library collection for byzantine fault tolerance. In *2019 31st SBAC-PAD*, pages 234–240. IEEE.

Araújo, Rodolfo (2019). eoscompiler on github.com/rodoufu. Technical report, https://github.com/rodoufu/eoscompiler.

Buterin, V. (2015). On public and private blockchains. *Ethereum Blog*, 7.

Castro, M. and Liskov, B. (1999). Practical byzantine fault tolerance. In *OSDI*, volume 99, pages 173–186.

Ciuffo, L., Salmito, T., Rezende, J., and Machado, I. (2016). Testbed fibre: Passado, presente e perspectivas. In *Anais do WPEIF 2016 Workshop de Pesquisa Experimental da Internet do Futuro*, pages 3–6. sn.

Coelho, I. M. (2020). Grupo de pesquisa em algoritmos e logística descentralizada. Technical report, https://github.com/gp-alode.

Coelho, I. M., Coelho, V. N., Araujo, R. P., Yong Qiang, W., and Rhodes, B. D. (2020). Challenges of pbft-inspired consensus for blockchain and enhancements over neo dbft. *Future Internet*, 12(8).

Coelho, V. N. and Coelho, I. M. (2018). Apresentação: "why neo has potential for south america, the connection with oriental developers". In *Blockchain Summit Uruguay (05/09/2018)*.

Costa, J., Coelho, I. M., Formigoni, J. R., Cristiano, M. L., Macadar, M. A., and Almeida, V. (2019). Mesa redonda: "varanda its - blockchain como infraestrutura de governo". In *ITS Rio (02/12/2019)*.

Grigg, I. (2017). Eos-an introduction. *White paper. https://whitepaperdatabase. com/eos-whitepaper*.

Hongfei, Da and Zhang, Erik (2015). Neo: A distributed network for the smart economy. Technical report, NEO Foundation.

LACChain (2021). Aliança global lacchain. Technical report, https://www.lacchain.net/home.

Mendoza, K. S. A., Vera, H. R. M., Medranda, W. M. O., and Navarrete, R. F. B. (2019). El impacto de la tecnología blockchain y sus diversas ventajas aplicadas en américa latina.

Nakamoto, S. et al. (2008). Bitcoin: A peer-to-peer electronic cash system. *White Paper*.

RNP (2021). Rede nacional de ensino e pesquisa - rnp. Technical report, https://www.rnp.br/.