Dissertação apresentada à Pró-Reitoria de Pós-Graduação do Instituto Tecnológico de Aeronáutica, como parte dos requisitos para obtenção do título de Mestre em Ciências no Programa de Pós-Graduação em Engenharia Eletrônica e Computação, Área de Informática.

Rogério Caldas dos Santos

UMA ABORDAGEM COMPARATIVA ENTRE ICPS BASEADAS EM *BLOCKCHAIN*

Dissertação aprovada em sua versão final pelos abaixo assinados:

Prof. Dr. Cesar Augusto Cavalheiro Marcondes
Orientador

Prof. Dr. Pedro Teixeira Lacava
Pró-Reitor de Pós-Graduação

Campo Montenegro
São José dos Campos, SP - Brasil
2021

Dados Internacionais de Catalogação-na-Publicação (CIP)
Divisão de Informação e Documentação

Santos, Rogério Caldas dos
Uma abordagem Comparativa entre ICPs baseadas em *Blockchain* / Rogério Caldas dos Santos. São José dos Campos, 2021.
118f.

Dissertação de Mestrado – Curso de Engenharia Eletrônica e Computação. Área de Informática – Instituto Tecnológico de Aeronáutica, 2021. Orientador: Prof. Dr. Cesar Augusto Cavalheiro Marcondes.

1. Blockchain. 2. ICP. 3. Escalabilidade. 4. Desempenho. 5. Testbed. I. Instituto Tecnológico de Aeronáutica. II. Título.

REFERÊNCIA BIBLIOGRÁFICA

SANTOS, Rogério Caldas dos. **Uma abordagem Comparativa entre ICPs baseadas em *Blockchain***. 2021. 118f. Dissertação de Mestrado – Instituto Tecnológico de Aeronáutica, São José dos Campos.

CESSÃO DE DIREITOS

NOME DO AUTOR: Rogério Caldas dos Santos
TITULO DO TRABALHO: Uma abordagem Comparativa entre ICPs baseadas em *Blockchain.*
TIPO DO TRABALHO/ANO: Dissertação / 2021

Rogério Caldas dos Santos
Av. Cidade Jardim, 679
12.233-066 – São José dos Campos–SP

UMA ABORDAGEM COMPARATIVA ENTRE ICPS BASEADAS EM *BLOCKCHAIN*

Rogério Caldas dos Santos

Composição da Banca Examinadora:

Prof. Dr.	Adilson Marques da Cunha	Presidente	-	ITA
Prof. Dr.	Cesar Augusto Cavalheiro Marcondes	Orientador	-	ITA
Prof. Dr.	Lourenço Alves Pereira Junior	Membro Interno	-	ITA
Prof. Dr.	Nandamudi Lankalapalli Vijaykumar	Membro Externo	-	INPE

ITA

Dedico este trabalho a Deus, pois em tudo fui enriquecido Nele. À minha esposa e filho pelo apoio, dedicação, compreensão e ternura a mim dispensados. A todos que comigo, mesmo distantes, suportaram a separação de nosso convívio em prol deste trabalho.

Agradecimentos

A Deus, por confiar-me a vida e as obras que nela posso realizar, onde caminhos, pontes e multivariadas portas são apresentadas e Nele tenho o direcionamento das decisões.

Ao Prof. Dr. Cesar Marcondes por sua orientação, profissionalismo e competência tão importantes para realização desta pesquisa. Pela dedicação no ensinar e na disponibilização de recursos computacionais e de aprendizagem, sem os quais não seria possível entregar este contributo à Ciência.

"Ouve atentamente, escuta as palavras dos sábios e consagra o teu coração a estudar a minha experiência."
— Provérbios 22:17

Resumo

Os sistemas de certificação digital têm um fundamental papel na segurança das informações. Neste contexto, aplicações de Infraestruturas de Chaves Públicas (ICPs) baseadas na tecnologia de *Blockchain* vêm sendo estudadas para melhor atender às novas demandas tecnológicas da sociedade e do estado moderno. Neste cenário, esta pesquisa envolveu uma campanha de análise de desempenho, com o intuito de compreender as ICPs baseadas na tecnologia de Cadeia de Blocos (*Blockchain*) de forma realista, verificando aspectos como escalabilidade e tempo de respostas de transações. Com este objetivo, foram realizados experimentos no ambiente de teste da moeda virtual *Ethereum*, utilizando a rede *Rinkeby*, por meio de comparação de contratos inteligentes publicamente reconhecidos, variando-se a carga para avaliar a escalabilidade e o tempo de resposta. Além disso, foi desenvolvido um ambiente de teste (*testbed*) para execução desses experimentos e realizada uma caracterização do ambiente experimental. Para obter melhor acurácia, em relação à comparação dos algoritmos, foram utilizados modelos de regressão linear, realizadas alterações nos contratos inteligentes, exploradas funções criptográficas mais intensas e elaborada uma proposta de melhoria do tempo de execução por distribuição das requisições em diferentes blocos de *Blockchain*. Os resultados mostraram diferenças importantes de desempenho entre os algoritmos testados e as aplicações de ICPs baseadas em *Blockchain*, em termos de escalabilidade, limitadas pelas quantidades máximas de transações por bloco da tecnologia de *Blockchain*.

Abstract

Digital certification systems play a key role in information security. In this context, Blockchain-based on Public Key Infrastructures (PKIs) applications are being studied to better meet new technological, society, and state demands. In this scenario, this research carried out tests of performance analysis in order to understand a Blockchain-based PKIs in more realistic way, verifying aspects such as scalability and response time of transactions. For this purpose, experiments were carried out in the Ethereum test environment, thus using the Rinkeby network, by comparing publicly recognized smart contracts, varying the load to assess scalability and response time. In addition, a testbed was developed to conduct such experiments and to characterize the experimental environment. For better accuracy, regarding the comparison between algorithms, linear regression models were used. Likewise, changes were made into smart contracts by exploring more intense cryptographic functions and a proposal was made to improve the execution time by distributing requests in different blocks of the Blockchain. The main results have shown important performance differences between tested algorithms and Blockchain-based PKIs applications limited by the maximum amount of transactions per Blockchain block in terms of scalability.

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

ABI	Application Binary Interface
AC	Autoridade Certificadora
ACE	Authentication and Authorization for Constrained Environments
ACM	Association for Computing Machinery
ACME	Automated Certificate Management Environment
API	Interface de Programação de Aplicativos
AR	Autoridade de Registro
CBC	Certificado na Blockchain
CBFT	Concurrent Byzantine Fault Tolerance
CPU	Central Processing Unit
DApp	Decentralized Application (aplicações descentralizadas)
DDoS	Ataque de negação de serviço distribuída
DHT	Distributed Hash Table
DL	Bibliotecas Digitais
DLT	Distributed Ledger Technology
DNS	Domain Name System (sistema de nomes de domínio)
EPTS	Efficient Privacy-preserving Thin-client Scheme
EthPKI	Ethereum PKI
EVM	Máquina virtual Ethereum
HC	Hospital das Clínicas
HSM	Hardware Security Module
HTTP	Hypertext Transfer Protocol
ICP	Infraestrutura de Chaves Públicas
ICP-AC	Infraestrutura de Chaves Públicas baseadas em Autoridade Certificadora
Id	Identificador
IKP	Instant Karma PKI
IoT	Internet das Coisas
IPFS	Inter Planetary File System
IRTF	Internet Research Task Force
JVM	Máquina Virtual Java

LCR	Lista de Certificados Revogados
NPM	Node Package Manager
NVS	Name Value Service
P2P	Peer-to-peer
PIR	Private Information Retrieval
PoA	Proof of Authority
PoW	Proof of Work
PTS	Privacy-preserving Thin-client Scheme
RBC	Registro na Blockchain
RPC	Remote Procedure Calls (chamada remota de procedimento)
SCPKI	Smart Contract-based PKI and Identity System
SSH	Secure Shell
SSL	Secure Sockets Layer
STD	Desvio padrão
TLS	Transport Layer Security
TTP	Trusted Third Party
WoT	Web of Trust

Sumário

1 Introdução

Grandes corporações que transacionam substanciais volumes financeiros, empresas que existem em função de seus segredos industriais ou até mesmo adolescentes que acabam de ganhar seus *smartphones*, não ficam isentos de ataques de *hackers*.

Por décadas, os Sistemas de Certificação Digital têm exercido um fundamental papel na segurança das comunicações e das informações, pois conferem validade jurídica às transações digitais, identificando e protegendo, de forma segura e transparente, as partes envolvidas.

Entretanto, a certificação digital sozinha não consegue desempenhar todos os papéis do ciclo de vida de gerência de ativos certificadores criptográficos como: emissão, distribuição e revogação de certificados digitais. Portanto, ao longo dos últimos anos, vêm se delegando às Infraestruturas de Chaves Públicas (ICPs) um conjunto de políticas, papéis e procedimentos para instanciar, conduzir, armazenar e até mesmo revogar certificados digitais (AXON, 2015).

Portanto, a principal ideia ou função das ICPs é a de facilitar a intermediação e prover uma forma meticulosa de provar e confirmar identidades de terceiros envolvidos em transferências de valores, bem como em comunicações. Porém, a adoção destas ICPs apresenta muitas dificuldades e empecilhos.

É possível apresentar essas dificuldades sob várias visões. Do ponto de vista de uma ICP baseada em Autoridade Certificadora (ICP-AC) e centralizada por natureza, cuja emissão de um certificado digital é fundamentada em relações de confiança (subordinações) entre Autoridades Certificadoras (ACs), existem altos custos (SALMAN *et al.*, 2018) envolvidos na criação de infraestrutura tecnológica, envolvendo aquisição de Cofres Digitais (*Hardware Security Module* - HSMs), centros de dados (*datacenters*), conectividades de rede, entre outras infraestruturas.

Além disso, a introdução de procedimentos burocráticos para validação física de dados pessoais de usuários e a gerência do ciclo de vida de certificados sem padronização vêm adicionando custos e criando ecossistemas fragmentados, complexos e caros. Esta complexidade extra é oriunda da competição de mercado e, muitas vezes, as autoridades certificadoras propiciam vendas casadas com outros serviços como varreduras de vulnera-

bilidades e seguros, em caso de falha de seus certificados.

Por outro lado, do ponto de vista de servidores e sítios ou portais *web* que necessitam de certificados digitais, existem grandes dificuldades para se lidar com altos custos, que podem variar de $178 dólares no caso de certificados para o único nome de um portal até $766 dólares para certificados coringas. Estes últimos certificados podendo ser aplicados para uma empresa como um todo. Paralelamente à dificuldade de custos, também existe a dificuldade das interações, via software, envolvendo entidades certificadoras para a obtenção de produtos de chaves criptográficas e/ou automatizações de processos de criação, revogação e recriação de certificados.

Finalmente, o ponto de vista dos clientes é muito dependente dos cenários de navegadores e das capacidades de softwares específicos, em termos de suporte aos certificados. Existem muitas ICPs não reconhecidas pelos principais navegadores Firefox, Google Chrome, Safari, Internet Explorer e Opera, que precisam ser instaladas manualmente em sistemas, o que pode exigir níveis elevados de conhecimentos de clientes. Além disto, o navegador não é o usuário final. Então, os certificados utilizados por navegadores tendem a ser pouco efetivos para permitir que usuários se conectem, por exemplo, a um Banco pela Internet.

1.1 Motivação

Em diferentes cenários e necessidades, certificados digitais precisam ser atribuídos diretamente aos usuários, por meio de cofres digitais como (*pendrives*) que armazenem assinaturas digitais. Como exemplo, pode-se citar o uso de assinaturas digitais em processos judiciais, relatórios médicos, entre outras atividades. Estas assinaturas propiciam segurança, não-repúdio e responsabilidade legal sobre documentos assinados.

No caso de um grande hospital, como o Hospital das Clínicas (HC) da Universidade de São Paulo (USP), com seus 15 mil funcionários [1], entre enfermeiros, médicos e pessoal de apoio administrativo, a utilização de ICPs baseadas em ACs torna-se até mesmo uma necessidade jurídica.

Órgãos reguladores como o Ministério Público Federal precisam de evidências confiáveis para processar casos de má prática em saúde. Neste caso, certificados auto assinados não podem ser utilizados e certificados de ACs poderiam vir a custar milhões de reais de manutenção anual para se viabilizar um Centro de Certificação Digital (CCD) com pessoal especializado, seguros e elaboração de planos operacionais para um ambiente associado.

Portanto, aspectos que visam solucionar problemas: de escalabilidade, de ponto único

[1] https://pt.wikipedia.org/wiki/Hospital_das_Clínicas_da_Faculdade_de_Medicina_da_Universidade_de_São_Paulo

de falha; de facilidade de gerenciamento de ciclo de vida; de certificados digitais; e de custos vêm se tornando uma constante em instituições industriais e até mesmo no meio acadêmico.

Dentro desse contexto, as implementações de ICPs vêm sendo tradicionalmente conhecidas e agrupadas na literatura em dois tipos distintos de arquiteturas: a mais comum, a arquitetura de ICPs centralizadas; e a arquitetura de ICPs descentralizada, cuja emissão do certificado digital se configura em uma Rede de Confiança ("Web of Trust" - WoT) envolvendo entidades de mesmo nível, onde todos possuem privilégios para emitir e confirmar reciprocamente as suas chaves públicas.

Devido aos vários aspectos anteriormente relatados, as arquiteturas tradicionais de ICPs centralizadas vêm sendo gradualmente substituídas, em certos nichos, por projetos de certificação menos rigorosos como o caso do *Let's Encrypt* (AAS *et al.*, 2019), um projeto com o objetivo de emitir certificados do tipo de Segurança da Camada de Transporte / Camada de Soquetes Seguros (*Transport Layer Security / Secure Sockets Layer* - TLS/SSL)[2] gratuitamente para tornar a Web mais segura.

O projeto *Let's Encrypt* funciona a partir da criação de certificados digitais, por meio da intermediação do Sistema de Nomes de Domínio (*Domain Name System* - DNS) utilizado para se verificar endereços do Protocolo da Internet (*Internet Protocol* - IP) ou de portais da Internet válidos *online*, mas não resolve carências como certificações pessoais. Outra linha de interesse constitui-se na substituição das ICPs baseadas em Autoridades Certificadoras por alternativas mais hierárquicas para lidar com centralização e escalabilidade.

No caso das arquiteturas de ICPs descentralizadas, uma WoT possui capacidade técnica de confirmar a identidade de seus componentes, por meio de uma rede de confiança transitiva entre as chaves e as assinaturas digitais de seus integrantes. No entanto, uma arquitetura WoT carece de comprovar consistência de criação das chaves para as suas respectivas identidades (AL-BASSAM, 2017).

Para preencher essa lacuna de arquiteturas para as ICPs, vêm surgindo propostas inovadoras de implementação de ICPs baseadas em *Blockchain*, eliminando o ponto central de falhas presente nas ICPs tradicionais, pois cada nó pode armazenar as transações e *hashes* dos dados já validados, segundo o consenso de uma rede.

Estruturas baseadas em *Blockchain* também tornam inviáveis as alterações de dados por parte de possíveis atacantes, pois requerem elevado poder computacional para realizar mudanças no consenso estabelecido, tornando impossível se modificar armazenamentos locais.

[2]https://www.cert.br/forum2018/slides/ForumCSIRTs2018-AlexandreSantos.pdf

A utilização de uma terceira parte confiável no uso da certificação digital se torna inócua para as implementações baseadas em *Blockchain*, uma vez que por meio dos seus algoritmos de consenso, torna-se possível, por exemplo, determinar volumes de transações, taxar serviços prestados como forma de incentivo aos colaboradores do sistema e, sobretudo, não depender da discricionariedade de um controlador exercendo unilateralidades de decisões sobre as redes (REBELLO *et al.*, 2019).

Em consequência, as ICPs baseadas na tecnologia de *Blockchain* garantem a retenção de identidade. Assim, impede-se o registro da identidade ou da chave pública para outro usuário já cadastrado. Segundo (DONG *et al.*, 2018), os algoritmos de consenso são os principais responsáveis para se garantir este quesito. Todas estas características e seu baixo custo associado, tornam as ICPs baseadas na tecnologia de *Blockchain* potenciais de serem cada vez mais utilizadas no futuro.

1.2 Objetivos e Contribuições

Com a problemática de diferentes arquiteturas de ICPs no horizonte e, especialmente, com as promissoras soluções baseadas na tecnologia de *Blockchain*, cuja utilização pode propiciar a solução de vários aspectos funcionais, considerados como críticos de ICPs, ainda assim, ficam as seguintes questões de pesquisa sobre a viabilidade de cenários.

"Seria possível as ICPs baseadas nas tecnologias de *Blockchain* resolver problemas de certificação para centenas ou milhares de usuários finais, como no caso das necessidades dos sistemas do HC da USP?"

"Seria possível descrever se esses sistemas são escaláveis e se estimar seus tempos de resposta e suas capacidades totais?"

A partir de reais capacidades de Contratos Inteligentes (*Smart Contracts*), envolvidos em uma implementação que suporta transações de ICPs na rede Ethereum, **o objetivo desta pesquisa é realizar uma campanha de medições que possa responder questões de desempenho de ICPs baseadas em *Blockchain***.

Considerando-se que já existem na literatura vários trabalhos viáveis de ICPs baseadas em *Blockchain*, esta pesquisa não envolve a criação de uma nova arquitetura e/ou implementação, mas apenas se propõe a medir empiricamente o desempenho de Contratos Inteligentes.

Para tanto, nesta pesquisa, após se realizar um levantamento bibliográfico, escolheu-se 3 implementações de ICPs baseadas na tecnologia de *Blockchain* para a realização das respectivas medições. Todas estas implementações foram uniformizadas em um mesmo contexto e, por vezes, readequadas utilizando-se a linguagem Solidity.

Para isso, foi escolhido o ambiente experimental de rede de teste chamado Rinkeby, que permitiu a realização de uma campanha de experimentos com pouca variabilidade e tráfego de fundo, ao mesmo tempo, mantendo-se aspectos realísticos da rede principal Ethereum.

Nesta pesquisa, destaca-se o desenvolvimento de: 1) um ambiente de testes (*testbed*) para os experimentos; 2) um *software* de aplicação para um controlador; e 3) uma metodologia para financiamento automático das transações.

Uma vez propriamente financiadas centenas de carteiras eletrônicas, o ambiente desenvolvido deverá orquestrar experimentos de execução de funções de ICPs do tipo validação e revogação com cargas variadas de clientes e repetições para maior robustez estatística. A quantidade total de experimentos a serem simulados deverá ser suficiente para se explorar, apropriadamente: algoritmos completos; funções específicas; quantidades de clientes; e quantidades de repetições.

Também deverá ser realizada uma caracterização inicial da rede de testes Rinkeby, para se medir sua possível interferência com as outras medições. Os resultados obtidos devem focar em aspectos de desempenho, capturando métricas de interesse como: tempo de resposta para a execução dos contratos inteligentes; e utilização de *gas* (taxas utilizadas nas redes Ethereum).

Modelos de regressão linear devem ser usados para comparar os algoritmos do ponto de vista de escalabilidade. Finalmente, devem ser adicionadas modificações nos Contratos Inteligentes, para explorar aspectos criptográficos mais intensos e propostas de melhorias de tempo de execução por espalhamento das requisições em diferentes blocos de uma *Blockchain*.

1.3 Organização da Dissertação

Este Capítulo 1 descreveu a introdução e a contextualização do problema a ser atacado e um resumo dos objetivos do trabalho de pesquisa, destacando-se os seus principais resultados esperados. O Capítulo 2 apresenta conceitos, arquiteturas e tecnologias de ICPs e de *Blockchain* indispensáveis ao entendimento desta pesquisa. O Capítulo 3 descreve uma taxonomia sobre as ICPs baseadas na tecnologia de *Blockchain* e o estado da arte das pesquisas. Uma descrição aprofundada da metodologia, do ambiente de experimentação e das medições, bem como das ferramentas utilizadas e uma caracterização da rede de teste Rinkeby é apresentada no Capítulo 4. No Capítulo 5, apresentam-se todos os resultados da campanha de medição e suas escolhas (*tradeoffs*), bem como uma comparação utilizando-se regressão linear entre as abordagens e introduzindo melhorias possíveis para as ICPs baseadas em *Blockchain*. Finalmente, o Capítulo 6, apresenta as prinipais conclusões,

recomendações e sugestões para trabalhos futuros e alguma discussão sobre limitações existentes.

2 Fundamentação Teórica

Nos últimos anos, vêm crescendo os desafios para se garantir devidas proteções de acessos aos usuários, em meio a um aparato de comunicação digital fortemente representado pela 4ª Revolução Industrial [1], pela Internet das Coisas e pelas Cidades Inteligentes.

Nesse contexto, há que se ter mecanismos para garantir certificações digitais de arquivos e transações de envolvidos em sistemas computacionais. Além disto, aponta-se a criptografia de chave pública como uma base de segurança na comunicação atual, sendo as ICPs consideradas como uma solução prática para se realizar um gerenciamento apropriado de chaves públicas, por propiciar a criação de um elo entre estas e seus proprietários (DU, 2017).

2.1 Infraestruturas de Chaves Públicas - ICPs

As Infraestruturas de Chaves Públicas (ICPs) são estabelecidas por uma arquitetura composta por hardware, software, pessoas, políticas e processos necessários para a gestão do ciclo de vida dos certificados, que compreendem: criar, gerenciar, armazenar, distribuir e revogar certificados digitais fundamentados por criptografia assimétrica, conforme estabelecido por (SHIREY, 2000). Os principais componentes das ICPs são descritos nas subseções a seguir.

2.1.1 Autoridade Certificadora - AC

Uma Autoridade Certificadora (AC) é um agente que atua como um centralizador de serviços de certificação digital, que controla chaves criptográficas, gerencia permissões em um ambiente seguro e realiza gestão do Ciclo de Vida dos Certificados e da Lista de Certificados Revogados (LCR).

A fim de melhor distribuir demandas de serviços, potencializar crescimentos de ICPs e melhorar as chances de expansão geográfica de prestações de serviços das ICPs, uma AC

[1] https://rockcontent.com/br/blog/industria-4-0/

pode delegar alguns de seus serviços nativos para serem exercidos por uma Autoridade de Registro, por Serviços de Repositório e/ou por Autoridades Certificadoras Intermediárias.

Cabe ressaltar a necessidade de haver estreita relação de confiança na AC, como parte confiável na precisão de uma chave pública distribuída em um certificado digital (CHOKHANI; FORD, 1999). Isto se deve ao fato do certificado digital ser um arquivo de dados emitido pela AC, contendo uma chave pública e informações importantes de seu remetente.

2.1.2 Autoridade de Registro - AR

A Autoridade de Registro (AR) é o centro administrativo de uma ICP, onde o requerente pode se inscrever para obter seu certificado digital (SCHMEH, 2006). Atuando sob delegação de serviços pela AC, a AR realiza o processo de verificação e de validação de requerentes de certificados, conferindo as informações do usuário e, uma vez aprovando tais etapas, assina digitalmente este resultado sob a forma de requisição do requerente e a envia para a AC. O grau de confiança atribuído ao certificado está intimamente ligado ao processo de averiguação das informações realizado pela AR.

Com o intuito de aperfeiçoar a escalabilidade e minorar os custos operacionais, uma ICP pode ter várias ARs, por exemplo, em virtude de numerosos usuários finais ou quando se tem uma dispersão geográfica muito grande dos certificados digitais (ADAMS; LLOYD, 2003).

Ressalte-se que, independente do arquétipo estabelecido em uma ICP no tocante ao exercício técnico de uma AR, esta jamais terá permissão de emitir certificados ou LCRs, pois estas funções são exclusivas de uma AC (ADAMS; LLOYD, 2003).

2.1.3 Repositório de Certificados e Lista de Certificados Revogados - LCR

Atuando sob delegação de serviços pela AC, o Repositório de Certificados é o local de publicação dos certificados digitais e das Listas de Certificados Revogados (LCR) recentemente emitidos por uma ou mais ACs. Ressalta-se a necessidade de haver assinatura digital por parte da AC representada por este repositório para impedir ataques de substituições e fabricações. Para exequibilidade desta tarefa, o repositório tem que estar disponível todo o tempo, fato este que dever ser contemplado no projeto físico de suas instalações, na política de segurança ou em outros documentos que versem sobre a segurança de onde este repositório encontra-se localizado.

2.2 Modelos de Confiança em ICPs

Uma vez descritos os componentes de uma ICP, para o seu entendimento completo, é preciso lidar com os modelos de confiança utilizados. Define-se modelos de confianças como o conjunto de arquiteturas que estabelecem o relacionamento de confiança entre os entes de uma ICP. Assim, é possível se determinar questões tais como, em quais certificados uma entidade pode confiar, a maneira como esta confiança pode ser estabelecida e em quais situações esta confiança pode ser restrita ou regulada (ADAMS; LLOYD, 2003). A seguir, apresenta-se uma lista dos modelos de confiança mais utilizados em ICPs e suas descrições nas próximas Seções:

- Modelo de AC Única;
- Modelo Hierárquico;
- Modelo de Certificação Cruzada; e
- Modelo *Web of Trust*.

2.2.1 Modelo de AC Única

Neste modelo, uma única AC é responsável pela gerência do ciclo dos certificados e emissão das LCRs, bem como por ter o controle sobre tais informações. Para estabelecimento da confiança entre os usuários, basta a confiança nos certificados e nas LCRs desta AC eliminando-se assim a confiança em certificados de outras. Este modelo é utilizado somente em ambientes pequenos, de fácil depuração e improvável escalabilidade, conforme mostrado na Figura 2.1

FIGURA 2.1 – Modelo de Confiança - AC Única

2.2.2 Modelo Hierárquico

Neste modelo, a ICP apresenta-se visualmente como uma estrutura de dados denominada árvore, cuja raiz é a "âncora de confiança" para os demais entes da ICP sob o seu domínio. A AC Raiz é auto assinada e geralmente emite os certificados das ACs intermediárias, localizadas nos nós não folhas, enquanto estas emitem os certificados dos usuários finais, localizados nas folhas desta árvore, conforme mostrado na Figura 2.2

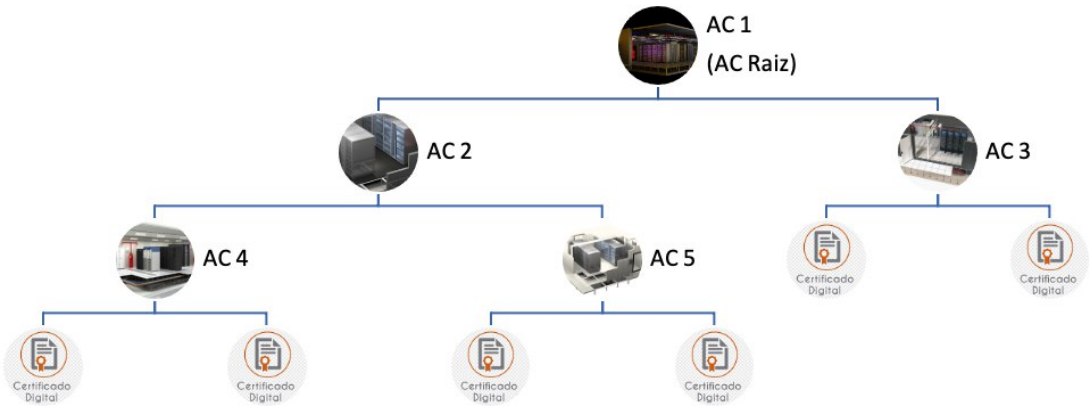


FIGURA 2.2 – Modelo de Confiança - Hierárquico

2.2.3 Modelo de Certificação Cruzada

Neste modelo, efetua-se a ligação entre duas ACs que se encontram em domínios diferentes, possibilitando a certificação entre os entes das duas ACs, aumentando assim a interoperabilidade, no que se refere à confiança, conforme mostrado na Figura 2.3

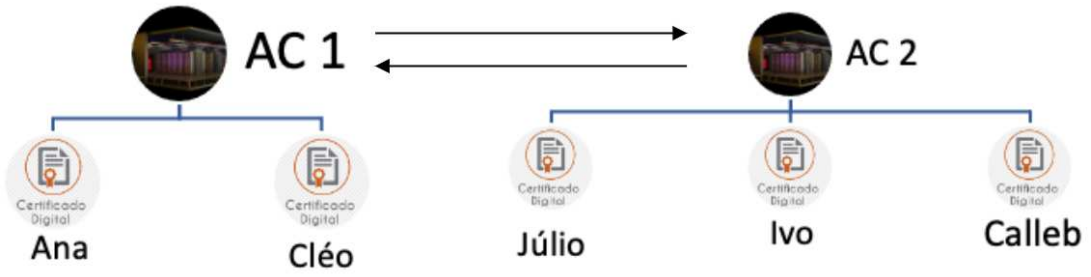


FIGURA 2.3 – Modelo de Confiança - Certificação Cruzada

2.2.4 Modelo *Web of Trust*

O modelo Web of Trust, também conhecido por modelo centrado no usuário, não adota a AC no sistema de certificação. Neste caso, a construção do modelo de confiança é de responsabilidade do próprio usuário, cuja estruturação consiste na agregação

das chaves classificadas como confiáveis juntamente com as chaves confiáveis de outros usuários, formando assim uma rede de confiança. Desta forma, segrega-se um possível comprometimento de toda a estrutura bastando revogar a chave do signatário, mitigando o impacto na rede de confiança. Portanto, este modelo supera as limitações da terceira parte confiável e do ponto único de falhas encontradas nas ICPs baseadas em AC.

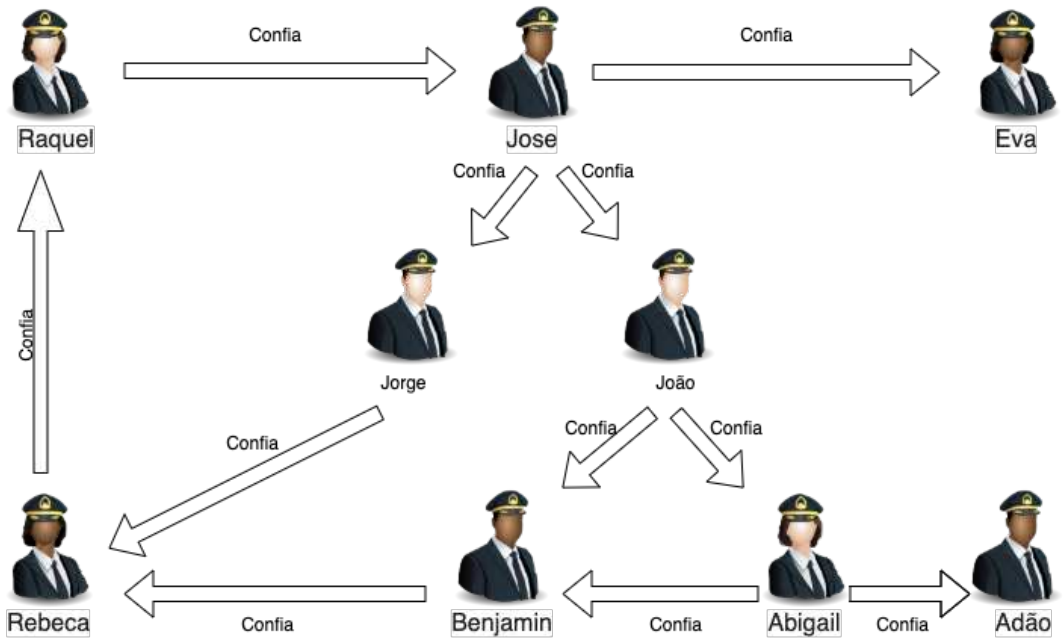


FIGURA 2.4 – Modelo de Confiança - Web of Trust

Analisando-se a Figura 2.4, torna-se simples compreender o funcionamento da rede de confiança proposta pelo modelo em tela. Por exemplo, Rebeca futuramente ao receber um certificado que por presunção pertence a Eva, toma conhecimento que este certificado foi assinado por José a quem ela não conhece. No entanto o certificado de José, fora assinado por Raquel a quem ela conhece e confia. De posse desta informação, Rebeca tem elementos para decidir confiar na chave de Eva (percorrendo a trilha de conhecimento transitivo entre Raquel, José e Eva) ou pode decidir rejeitar a chave de Eva, avaliando que não é conveniente introduzir Eva em sua rede, uma vez que Raquel adota como parâmetro para inclusão os links mais próximos de si.

2.3 A Tecnologia de *Blockchain*

Uma vez introduzidos os conceitos de ICPs e os modelos de confiança em ICPs mais difundidos, faz-se necessário introduzir a fundamentação teórica a cerca da nova tecnologia de *Blockchain* que pode ser utilizada como uma solução futura para as ICPs, formando assim um novo modelo de confiança.

A concepção de *Blockchain* teve sua origem atrelada à apresentação do artigo “Bitcoin:

a peer-to-peer electronic cash system" (NAKAMOTO, 2009), que detalha o funcionamento de um sistema de criptomoedas. Este artigo foi lançado logo após a crise do *subprime* [2] em 2008, e teve por objetivo criar uma moeda independente de bancos mais estável em relação às flutuações financeiras ocasionadas por políticas econômicas de cada Estado ou Bloco Econômico, tendo como tecnologia de suporte a este ineditismo a *Blockchain*.

A *Blockchain* pode ser definida como uma estrutura de dados que armazena, ordenadamente, uma cadeia de blocos ligados por uma camada criptográfica, constituindo um sistema de livro-razão público descentralizado, baseado na comunicação e autenticação de registros sobre uma rede *PeerPeer-to-Peer* (P2P), comumente chamada de *Distributed Ledger Technology* (DLT). (SWAN, 2015) realça que esta estrutura é composta por um banco de dados distribuído por todos os nós de uma rede, atualizado por mineradores, vigiado por todos e, no entanto, de propriedade de ninguém.

A estrutura de livro-razão é comumente utilizada por instituições financeiras assim como por demais autoridades centrais terceirizadas, pois sistematiza um livro de duplas entradas (débito e crédito, origem e destino) para cada transação, no intuito de não permitir o gasto duplo na execução desta transação.

Além da disponibilidade e da integridade das transações asseguradas pelo mecanismo de consenso que suporta este livro-razão público, convém destacar as seguintes características da tecnologia de *Blockchain* apontadas por (ZHANG; BOULOS, 2020):

- Descentralização - Não necessita de um nó central que tome as decisões na rede P2P, nem de uma entidade terceira para tal fim, pois as decisões são tomadas pela rede;
- Transparência e Auditabilidade - As transações registradas no livro-razão são públicas para o público a que se destina, permitindo verificação e auditoria. Cada transação pode ser vistoriada, assim como quaisquer propostas de mudanças têm que ser aprovadas por toda a rede, por meio do mecanismo de consenso; e
- Imutabilidade e Irrefutabilidade - Não há alteração no registro de um livro-razão da *Blockchain*. Possíveis atualizações só serão factíveis por meio da geração de novas transações e da realização de um novo consenso.

2.3.1 Mecanismo de Consenso

Oriundo da computação distribuída, em uma *Blockchain* o conceito de consenso visa estabelecer uma coordenação entre as diversas perspectivas dos pares que constituem uma rede *Blockchain*, a fim de conceber uma compreensão única sobre o modo de ordenamento dos blocos assim como dos seus conteúdos. Para o suporte necessário ao entendimento

[2] https://ab2l.org.br/bitcoin-historico-o-que-e-quem-criou/

deste trabalho de pesquisa, serão descritos a seguir os mecanismos de consenso mais conhecidos no ecossistema da tecnologia *Blockchain* e utilizados nos testes realizados nesta pesquisa, de acordo com (MIERS *et al.*, 2019):

- *Proof of Work* (PoW) – Este mecanismo de consenso é fundamentado em uma disputa matemática entre os mineradores. O vencedor desta disputa será o eleito para incluir o novo bloco na *Blockchain*. Na prática, esta disputa se traduz em descobrir uma entrada que possua valor de hash menor que um valor alvo estabelecido. Este foi o mecanismo adotado pela moeda virtual Bitcoin que iniciou a era *Blockchain*, sendo o mais difundido até hoje. O PoW estabelece que a segurança da rede é exercida por todos os nós, sendo estes devidamente recompensados pelo seu trabalho. Este mecanismo é geralmente empregado em *Blockchains* públicas, onde qualquer pessoa tem a possibilidade de participar como nó da rede (BASHIR, 2020). Os principais exemplos de *Blockchains* públicas ou não-permissionadas são: a Bitcoin e a *Ethereum*; e

- *Proof of Authority* (PoA) - Este mecanismo de consenso usa um conjunto de autoridades configuradas para colaborar sem confiança. Este conjunto tem privilégio para criar novos blocos e proteger a *Blockchain*. A assinatura de novos blocos por parte das autoridades se dá mediante ao uso de certificado digital válido. Isto decorre de iniciativas para impedir a personificação destes nós especiais por outros nós. Este mecanismo é geralmente empregado em *Blockchains* consorciadas ou federadas, pois existe uma pré-seleção de seus participantes. Os principais exemplos de *Blockchains* consorciadas são: Hyperledger e Quorum.

2.3.2 Bloco

Bloco é um container que armazena as transações enviadas para uma Blockchain. Cada bloco é identificado por uma assinatura dos dados do bloco denominada *hash*. Um bloco é composto por um cabeçalho e um corpo com suas respectivas transações. Reforçando o entendimento sobre *Blockchain* como uma implementação de um livro razão público descentralizado, (BADR *et al.*, 2018) descrevem o bloco como uma página ou tabela na qual se registra uma coleção de transações confirmadas. Como estes blocos estão distribuídos por toda a rede, apagar um bloco de uma *Blockchain* seria um grande esforço com improváveis chances de sucesso. As informações sobre a constituição de cada bloco ficam em seu cabeçalho e são representadas por: *Timestamp*, raiz Merkle, Nonce, Meta de Dificuldade (*Difficulty Target*), uma referência do bloco e uma referência ao bloco anterior.

2.3.3 Raiz da Árvore Merkle

A raiz da árvore Merkle, também conhecida como raiz Merkle, é o elemento do cabeçalho do bloco que sumariza todos os hashes das transações existentes em um bloco. Conforme ilustrado na Figura 2.5, a raiz Merkle proporciona uma forma simples de se manter e verificar a integridade dos dados por meio de sua árvore. Isto é possível pois a cadeia de hashes que constitui esta árvore, reduz consideravelmente o espaço em disco que seria utilizado caso fossem armazenados os dados no lugar destes hashes. A função da característica determinística do hash permite legitimar tais dados. Cabe destacar que o hash que representa a raiz Merkle será utilizado para compor o número do bloco no ato da criação do mesmo.

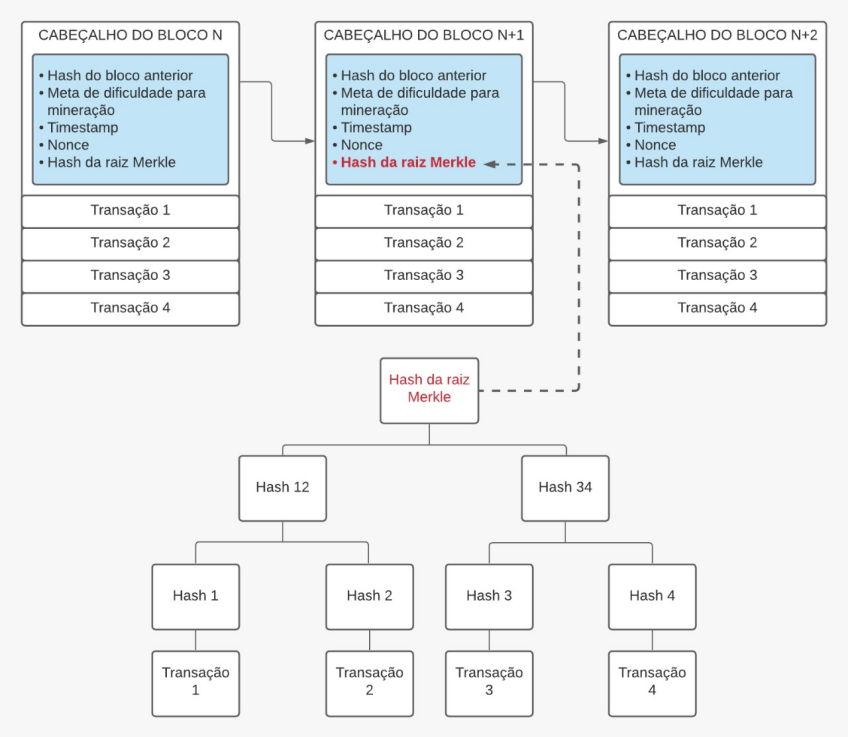


FIGURA 2.5 – Árvore de Merkle

(BRENNAN; LUNN, 2016) descrevem que a árvore Merkle produz comprovações incontestáveis de todas as transações que ocorreram por meio de um processo exaustivo e repetitivo que resulta na raiz Merkle, representando cada transação e seus dados. Convém situar que a árvore Merkle é uma árvore binária, cujos nós folhas caracterizam os hashes das transações e, recursivamente, cada ancestral é formado pela combinação dos nós filhos até resultar em um único valor de hash, denominado raiz Merkle. Esta arquitetura produz vários benefícios e, neste contexto, destacam-se: a representação de forma eficiente e escalável do estado da *Blockchain*, a sumarização de blocos de dados e a prova de sua existência. É possível obter esta verificação em tempo e espaço de O (log2 (N)) (SZYDLO, 2004).

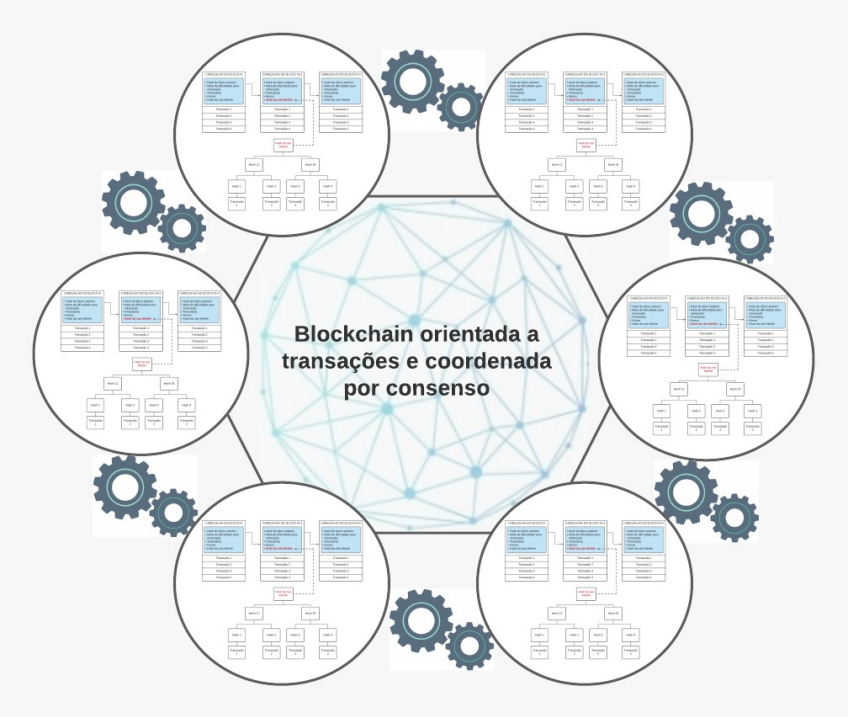


FIGURA 2.6 – Blockchain

2.3.4 Nonce

Nonce é um número gerado de forma arbitral e tem por premissa ser usado somente uma vez. Portanto, N = Number + once, onde N é igual a Nonce (WANG; XIE, 2019). O processo de inserção de um bloco na *Blockchain* requer a regulagem do Nonce para a produção de um hash que satisfaça um determinado critério, por exemplo, encontrar certa quantidade de dígitos zeros no início do endereço (0x0). Após o sucesso desta associação, o minerador faz a combinação dos dados do cabeçalho do bloco, o que inclui o Nonce, gerando bloco para a inserção na rede e, posteriormente, a validação pelos demais blocos da rede conforme padrão e formato definidos (TAPSCOTT; TAPSCOTT, 2016).

2.3.5 Meta de Dificuldade (*Difficulty Target*) de Mineração

A meta de dificuldade de mineração expressa, em termos de tempo e hashing, a mensuração do esforço dos mineradores para se obter um hash adequado para o seu bloco, ou seja, um bloco que esteja em condições de ser verificado e adicionado à *Blockchain*.

Em função desta adequação ao obter o hash do bloco, por exemplo, com certa quantidade de dígitos "0" no início de seu endereço, (WU *et al.*, 2019) retratam que tais inserções aumentam exponencialmente a configuração de dificuldade, tornando-se árdua a avaliação do Nonce associado, uma vez que esta dificuldade é resolvida pela própria rede.

Cada rede redimensiona sua dificuldade visando gerenciar melhor seu crescimento em

função da quantidade de mineradores e pelo consequente acúmulo de poder de hashes. Este fato impacta na redução do tempo de geração de blocos e, em última instância, pode desajustar a sincronização de blocos em meio a realização das transações.

2.3.6 *Timestamp*

O campo timestamp representa a data e a hora exata em que o bloco de transações foi gerado. A *Blockchain* implementa um servidor timestamp e consegue com exatidão realizar a prova de existência das transações. (NAKAMOTO; BITCOIN, 2008) estabelece a relação entre um timestamp e o seu timestamp anterior, assim como desta participação na composição do hash. Esta é uma maneira de garantir a integridade das transações a partir de sua assinatura.

2.3.7 Transações

As transações representam as mais diversas operações de gravação realizadas na cadeia de blocos, sendo possível realizar desde transferência de fundos por meio de criptomoedas, transações onlines e até execução de um sistema. Ao realizar tais operações, as transações podem ser visualizadas como a representação da mudança de estado na *Blockchain* (PALLADINO, 2019). A *Blockchain* tem um ecossistema próprio de funcionamento para que seja possível registrar transações, visando sua segurança, perenidade e consistência, mesmo em um ambiente que não requer a confiança e o conhecimento prévio entre os seus componentes. Por meio da execução de uma transação de revogação, muito comum nas ICPs baseadas em *Blockchain*, a Figura 2.7 ilustra e são descritas a seguir as principais etapas de uma transação:

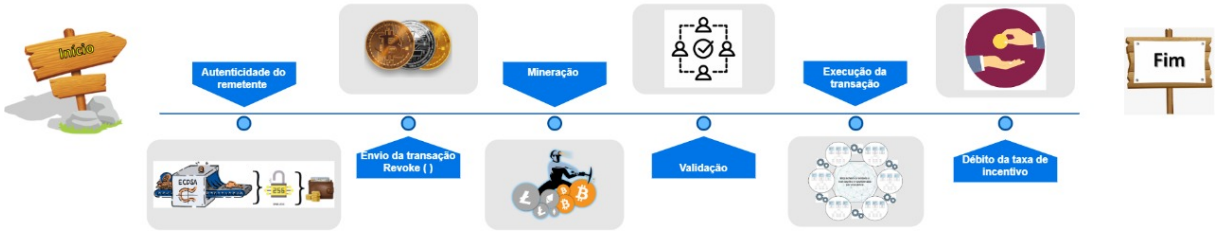


FIGURA 2.7 – Etapas de uma Transação

- O usuário, ao iniciar uma transação na *Blockchain*, faz uso de seu endereço de *Wallet* para assinar digitalmente a transação desejada. Nesta mesma etapa, ele ajusta a taxa de incentivo a ser cobrada pelo trabalho dos mineradores. Neste ponto, convém ressaltar a segurança requerida por todos os atores que fazem parte deste ambiente descentralizado. Tal fato pode ser observado pelas funções criptográficas coexistentes para a geração do endereço de *Wallet*. Igualmente, convém mencionar que o endereço de *Wallet*, ou endereço de carteira, é a identificação de um usuário no ambiente *Blockchain*. Tal endereço é criado a partir da combinação da chave pública e privada do usuário. Este endereço pode ser compartilhado publicamente sem expor as chaves do usuário;

- Uma vez enviada a transação e resolvida a disputa entre os mineradores, conforme descrito na Seção 2.2.1, o minerador vencedor reúne as mais variadas transações enviadas para execução e realiza uma averiguação de suas assinaturas inserindo-as em um bloco; e

- Desta forma o bloco é propagado pela rede e os demais nós validam o trabalho do minerador, realizando uma segunda verificação das assinaturas. Uma vez obtida a validação, a transação é executada na *Blockchain* e a taxa de incentivo é debitada do remetente.

A tecnologia *Blockchain* apresenta-se sobretudo como um novo entrante para tratar as atuais necessidades no que tange à descentralização dos serviços infraestruturais. De igual modo, observa-se a notabilidade desta tecnologia para lidar com questões de pesquisas que se encontram em aberto no gerenciamento de confiança, identidades e resolução de nomes proposto pelo *Internet Research Task Force* (IRTF) [3].

Esta pluralidade de escopo de atuação já constatada na tecnologia *Blockchain*, pode ser facilmente observada por dois marcos temporais caracterizados como *Blockchain* 1.0 e 2.0.

A *Blockchain* 1.0 é evidenciada com o surgimento do sistema de criptomoedas, principalmente aperfeiçoando a sistemática de transferência de recursos financeiros digitais comumente evidenciadas pelo uso de Bitcoins e Altcoins [4].

A *Blockchain* 2.0 é evidenciada pela expansão na empregabilidade do livro-razão em proveito das transações descentralizadas para registro, confirmação e transferência de patrimônio e propriedades, por meio dos chamados contratos inteligentes. Estes consistem na transmutação dos termos de um relacionamento entre as partes e na determinação de um relacionamento com código criptográfico.

[3] https://trac.ietf.org/trac/irtf/wiki/blockchain-federation
[4] https://exame.com/future-of-money/dinheiro-tendencias/o-que-sao-altcoins/

Desta ambiência surgem plataformas que têm por objetivo reduzir o esforço empreendido de pessoas e empresas no desenvolvimento das chamadas Aplicações Descentralizadas ou *Decentralized Apps* (DApps), aplicações cujo código de *back-end* (representado pelo contrato inteligente) é executado em uma rede *Peer-to-Peer* descentralizada (WU *et al.*, 2019). Dentre as plataformas existentes, a *Ethereum* é a de maior visibilidade que faz uso de contratos inteligentes.

Vitalik Buterin conceituou *Ethereum* em novembro de 2013 e implementou a sua primeira versão em maio de 2015 (WOOD *et al.*, 2014). O seu modelo de execução é especificado por meio do modelo formal de uma máquina de estado virtual, conhecido como máquina virtual *Ethereum* (*Ethereum Virtual Machine* - EVM). A EVM é uma camada de abstração que permite a conexão de clientes *Ethereum* com a *Blockchain*, viabilizando uma gama de serviços na cadeia de blocos. Uma importante função da EVM é gerenciar o estado da rede *Ethereum*, que ocorre na execução das transações e na interação entre as contas. Como consequência, todos os nós da rede *Ethereum* executam a EVM e validam as transações.

2.4 Contratos Inteligentes com *Ethereum*

Um dos aspectos mais importante para a implementação de uma ICP em *Blockchain* é que o aplicativo decentralizado precisa operar como um contrato inteligente, independente, sempre ativo, sem terceiros e sempre funcional. Estas características encontram-se disponíveis na rede principal *Ethereum*. Destaca-se aqui o papel do seu cliente e suas funcionalidades, usadas ao longo deste trabalho de pesquisa.

O cliente *Ethereum* é uma implementação do protocolo *Ethereum* que interage com a rede *Ethereum* e a *Blockchain*, realizando sincronismo das novas cadeias, fazendo conexão com pares (*peers*), transmitindo transações locais para a rede e fornecendo capacidade básica de mineração (IYER, 2018).

Um cliente *Ethereum* representa cada nó e, dependendo de sua configuração, tem condições de realizar a verificação de novas transações, a execução de novos contratos e o processamento de novos blocos.

A execução de um cliente *Ethereum* implica na sincronização deste cliente aos demais clientes da rede. Neste contexto, (SOLOMON, 2019) elenca os seguintes modos de sincronização:

- *Full* - o cliente baixa a cadeia de blocos completa para sua estação de trabalho, obtendo desta forma o cabeçalho e o corpo de cada bloco, validando as transações e blocos desde o bloco de origem. Possivelmente esta opção é a mais indicada quando

se quer obter uma resposta mais rápida para a sincronização, pois o nó não precisa realizar a solicitação de blocos ausentes, todos estão presentes;

- *Fast* - o cliente baixa a cadeia de blocos completa para as 1000 transações mais recentes. Esta opção é recomendada quando se deseja obter pelo menos o histórico das 1000 transações mais recentes, mas deseja-se economizar algum espaço em disco; e

- *Light* - o cliente baixa apenas o estado atual da *Blockchain* e faz solicitações pontuais quando falta algum bloco. Esta opção é recomendada quando se tem o objetivo de economizar espaço em disco, como no caso de dispositivos móveis.

Com esse cliente também é possível verificar o estado de um nó e o estado da rede, enviar transações e realizar o gerenciamento das contas. Foi adotado como cliente *Ethereum* para este trabalho o Geth, que é uma implementação de cliente *Ethereum* na linguagem de programação Go, por possuir vasta documentação e uma expressiva quantidade de funcionalidades implementadas, sendo também o mais popular.

2.4.1 Contas

A interação dos usuários com a rede *Ethereum* se dá por meio de contas, cujos endereços são definidos por camadas de criptografia assimétrica que assinam as transações enviadas à *Blockchain*. As contas *Ethereum* são classificadas em dois tipos: contas de propriedade externa, controladas via chave privada sem código computacional associado, e contas de contrato, que executam os contratos inteligentes acionados pela transação ou por chamadas recebidas de outros contratos (HOMESTEAD, 2016).

2.4.2 Transações

Um dos grandes diferenciais da rede *Ethereum* é a possibilidade de executar as mais diversas operações que encapsulam o conjunto de instruções pretendidas. Para isto, faz-se necessário que estas operações sejam assinadas com a chave privada do remetente. Tais transações implicam exclusivamente na realização de uma chamada de mensagem ou na criação de um contrato. A seguir, descrevem-se os campos comuns das transações anteriormente citadas (SOLOMON, 2019):

- Nonce - valor escalar unitariamente incrementado a cada transação encaminhada pelo remetente. Sendo assim, o campo Nonce aponta para a última transação executada e garante a ordenação adequada da execução das transações;

- Assinatura - consiste na assinatura digital do remetente, a fim de se comprovar sua identidade junto a cadeia de blocos;
- *Gas Price* ou Preço do Consumo Computacional - preço unitário que o remetente concorda em pagar para executar da transação;
- *Gas Limit* ou Limite de Consumo Computacional - representa o limite superior que o remetente concorda em pagar para a execução da transação;
- Destino - contém o endereço da conta do destinatário da transação; e
- Valor - representa a quantidade de Wei [5] a ser transferida para a conta de destino, no caso de uma chamada de mensagem. Caso seja uma criação de contrato, observa-se como uma dotação inicial para a conta recém-criada.

Devido à heterogeneidade das transações, principalmente quanto à complexidade e ao tamanho de suas execuções, estas representam custos computacionais que a plataforma *Ethereum* mapeia e utiliza para suas execuções. Estes custos são informados nos campos *gas price* e *gas limit*. Logo, ao realizar uma transação, é necessário fornecer Ether suficiente, precificação para o *gas*, para a referente transação. Quando a conta de destino for do tipo contrato, a execução do mesmo será realizada respeitando-se o limite do *gas* referente ao que está nesta conta. No entanto, existem casos em que a transação falha por escassez de saldo ou de *gas* atribuído à realização da transação, como mostrado na Figura 2.8.


```
Será realizado o experimento com o Address:0x7468600c6b34b2e890dcfce03a6326bbe2f6d1eb
Error: Transaction has been reverted by the EVM:
{
  "blockHash": "0xe2f8b980d0513f647cb5a398c574910071b9aeacecd9368b5d6fd4c9115f4b7e",
  "blockNumber": 7935020,
  "contractAddress": null,
  "cumulativeGasUsed": 352468,
  "from": "0x7468600c6b34b2e890dcfce03a6326bbe2f6d1eb",
  "gasUsed": 92131,
  "logsBloom": "0x00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000",
  "status": false,
  "to": "0xf62de2bba0337acf0dbff331a58fe144c5f42680",
  "transactionHash": "0x8d6ed558006372f12207ce1467f17b77d8ff7d49173fcf0619b7453fea674047",
  "transactionIndex": 3,
  "events": {}
}
    at /home/rogerio/Apps/node_modules/web3/node_modules/web3-core-method/src/index.js:364:46
```


FIGURA 2.8 – Erro : mau redimensionamento de *gas/Ether*

Neste caso, as mudanças de estados ocorridas são revertidas com exceção da taxa paga relativa ao trabalho dos mineradores e a transação na Blockhain é tida como falha, conforme mostrado na Figura 2.9.

A tarifação deste custo é realizada sobre cada byte e é incrementalmente proporcional ao tamanho da transação (BASHIR, 2020).

[5] Considera-se 1 Wei=10^{-18} ETH. Fonte - https://academy.binance.com/en/glossary/wei

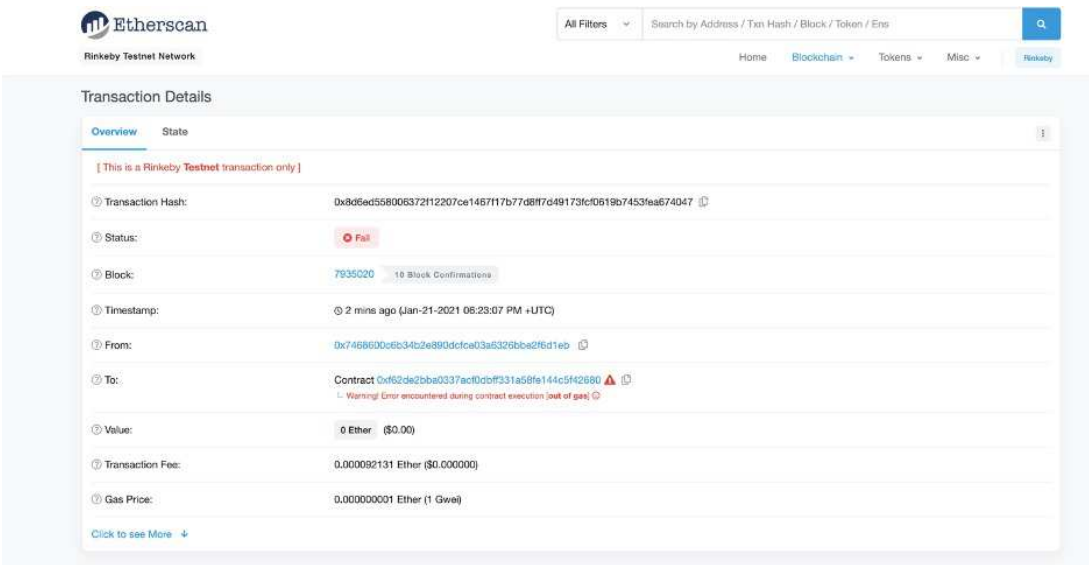


FIGURA 2.9 – Visualização de erro no Etherscan

2.4.3 Contratos Inteligentes

A concepção de contratos inteligentes teve sua primeira teorização realizada por Nick Szabo em 1997, por meio da apresentação do artigo *"Smart Contracts: Formalizing and Securing Relationships on Public Networks"*, onde previa criar uma sinergia entre princípios jurídicos, teorias econômicas e de protocolos confiáveis e seguros e, a utilização de criptografia e outros mecanismos de segurança para a proteção das partes (SZABO, 1997). No entanto, a primeira implementação de um contrato inteligente, mesmo que de forma deficitária, só foi realizada na cadeia de blocos do Bitcoin. A codificação deste contrato foi realizada na linguagem de script de transação da Bitcoin, chamada Script que permite apenas transferências de Bitcoins entre usuários, mas não implementa uma máquina de Turing-completa.

Este resumido panorama histórico aponta a necessidade de não simplificar o real significado dos contratos inteligentes, que guardam grandes semelhanças com as classes do paradigma orientado a objetos (DANNEN, 2017). No entanto, os contratos inteligentes devem traduzir as disposições contratuais para uma linguagem *Turing-Complete*, a fim de se minorar os intermediários nas operações e, sobretudo, ter por premissa a imutabilidade do contrato, ou seja, este contrato e seu respectivo identificador não poderão ser modificados.

Para a codificação de contratos inteligentes, costumeiramente, usa-se uma linguagem de alto nível e após esta etapa são gerados pela EVM os bytecodes implantados via *Blockchain*. Atualmente, a linguagem de alto nível *Turing-Complete* mais reconhecida para desenvolvimento de contratos inteligentes é a Solidity.

Além da rede de produção da *Ethereum*, os contratos inteligentes podem ser implanta-

dos em redes de testes (*testnets*), representando locais próprios para a realização de testes e réplicas da rede de produção. As principais redes de testes são a Ropsten, Rinkeby e Kovan. A rede Ropsten utiliza o algoritmo de consenso PoW e clientes *Ethereum* Geth e Parity. Já as redes Rinkeby e Kovan, ambas utilizam o algoritmo de consenso PoA e clientes *Ethereum* Geth e Parity respectivamente. Dentre as opções disponíveis, foi utilizado para este trabalho de pesquisa a rede de teste Rinkeby, que se mostrou mais resiliente aos ataques e problemas de redes de testes executados por (IYER, 2018).

Uma outra opção para implantação de contratos inteligentes são os ambientes simulados, que conteplam uma espécie de *testnets* privadas, geralmente utilizadas por principiantes de *Blockchain*, pois desonera o usuário de realizar várias tarefas para trabalhar neste ambiente. Por exemplo, geralmente as redes simuladas já vêm com uma quantidade de contas criadas, desbloqueadas e financiadas e, a implantação de contratos inteligentes nesta ambiência, é sobremodo facilitada, não sendo adequado para realizar quaisquer medições realísticas. Os principais ambientes que contemplam redes simuladas são, o *Ganache*, o *Ganache-cli* e o *truffle*.

Quanto à execução dos contratos inteligentes, faz-se necessário definir corretamente valores relativos aos custos de *gas* necessários para cada transação a ser executada dentro do contrato.

2.5 Síntese do Capítulo 2

Este Capítulo expôs os fundamentos imprescindíveis para compreensão deste trabalho de pesquisa, pois a compreensão de ICPs baseadas na tecnologia *Bockchain* requer inicialmente o conhecimento de ICPs e da tecnologia *Blockchain*.

Consoante às ICPs, foram abordados: A definição de ICPs; os seus principais componentes; e os modelos de confianças existentes. Referente à tecnologia de *Blockchain*, foram abordados: o mecanismo de consenso; a definição de bloco e seus componentes como timestamp, raiz Merkle, Nonce e difficulty target; e o estudo dos contratos inteligentes com *Ethereum* abordando a conceituação do cliente *Ethereum* e seus modos de sincronização, como, *Full, Fast e Light*.

Por oportuno, foi apresentado o conceito de contas, de transações e de contratos inteligentes na plataforma *Ethereum*. Igualmente, foi enunciada a definição de transações e de seus campos mais comuns como, Nonce, assinatura, *gas price*, *gas limit*, destino e valor. Além disto, foram apresentadas a conceituação de contratos inteligentes, a linguagem Solidity e o ambiente para testes destes contratos.

Os fundamentos expostos neste Capítulo encontram-se alinhados aos trabalhos relacio-

nados descritos no próximo Capítulo e dão suporte à construção da *testbed* e do ferramental necessários para a realização da abordagem comparativa, proposta nesta pesquisa.

3 Revisão de Literatura

Este Capítulo apresenta uma revisão da literatura, explicando a metodologia empregada para o levantamento sistemático, criando uma taxonomia sobre ICPs baseadas em *Blockchain* e descrevendo o estado da arte neste domínio de conhecimento.

3.1 Levantamento Sistemático

Para a realização do levantamento sistemático foi efetuada a captura dos artigos das bases de dados, a pré-seleção de acordo com os critérios de inclusão e exclusão, resultando nas sínteses dos trabalhos relacionados constantes nas próximas seções deste Capítulo.

A captura dos artigos foi realizada por meio de uma exploração das fontes buscando-se nas Bibliotecas Digitais (DLs) das sociedades científicas, como IEEE *Computer Society*, ACM (*Association for Computing Machinery*), Springer e Elsevier. Deste processo, geraram-se as seguintes etapas:

1. Realização de busca usando-se como assuntos *"public key infastructure"*, *"blockchain"* e *"smart contract"* que identificou artigos, Capítulos de livros e "*surveys*" relevantes e elucidários que refletem um denso conhecimento sobre as distintas formas de implementação de uma ICP baseada em *Blockchain*, bem como se deu o entendimento da maneira como os contratos inteligentes participam neste ecossistema e, por fim reafirmou a assimilação referente às características de cada tecnologia e suas respectivas bases estruturantes;

2. Realização de busca usando-se como assuntos *"public key infrastructure"*, *"decentralized application"* e *"smart contract"* que permitiu a construção de uma base de conhecimento que esclarece os impactos na execução de transações em relação a quantidade de *gas* e *gas price* associados as possíveis redes *Ethereum*, para teste e implantação de DApps, bem como as opções de mecanismos de consenso decorrentes da rede escolhida. Por fim, no intuito de abranger aspectos relacionados ao desempenho das redes de testes da *Ethereum* foi efetuada uma nova etapa de busca para este intento; e

3. Realização de busca usando-se como assuntos *"decentralized application"*, *"ethereum test-net"* e *"performance"* que retornou um conteúdo acadêmico de fundamental importância no que se refere ao desempenho das redes de teste da *Ethereum*, segundo alguns eixos comparativos. A partir desta etapa, também foi possível se obter uma ampliação da discussão de artefatos deste novo ambiente descentralizado, relacionados às contas *Ethereum* e o tipo de sincronização dos clientes na EVM.

Desde as primeiras investidas de buscas dos trabalhos à luz da temática proposta para esta pesquisa, foram observados os seguintes pontos relevantes:

1. Temática de alta relevância, considerando a capilaridade de assuntos abordados nos trabalhos retornados pelas buscas, o número de citação associado, o contributo à ciência e por decorrência à sociedade; e
2. Assertividade da busca nas Bibliotecas Digitais (DLs) em detrimento aos buscadores não acadêmicos. Tal medida atestou a estas etapas de pesquisa a obtenção de trabalhos com uma maior qualidade técnica.

Após a leitura dos trabalhos contextualizando-os com o problema de pesquisa proposto foram observados os seguintes pontos relevantes como critério de inclusão dos trabalhos relacionados:

1. Trabalhos publicados em língua inglesa, visto a exiguidade de trabalhos publicados em língua portuguesa;
2. Enfoque em arquiteturas, métodos, soluções computacionais, mecanismos, *frameworks*, plataformas, protótipos e sistemas implementados; e
3. Corte cronológico definido foi de 2008 a 2020, a partir do advento da tecnologia *Blockchain*.

Foram observados os seguintes pontos relevantes, como critérios de exclusão dos trabalhos relacionados:

1. Trabalhos que abordam as temáticas apresentadas somente de modo sociotécnico;
2. Trabalhos na forma de editorial, *keynote*, resumo, tutorial e poster; e
3. Trabalhos repetidos (somente o mais completo foi considerado).

A Tabela 3.1 sintetiza o retorno das buscas realizadas nas três etapas da captura dos artigos e expressa o interesse das pesquisas em torno das temáticas em tela, bem como

anuncia um campo de pesquisa abundante para exploração ao se analisar a interseção das temáticas propostas.

TABELA 3.1 – Resultados Encontrados

Base de Dados	**Etapa 1** **"public key infrastructure" "blockchain" "smart contract"**	**Etapa 2** **"public key infrastructure" "decentralized application" "smart contract"**	**Etapa 3** **"decentralized application" "ethereum test-net" "performance"**
IEEE	287	21	4
ACM	36	1	1
Springer Link	82	38	2
Science Direct	3	0	0
Portal CAPES	67	13	0
Total	**475**	**73**	**7**

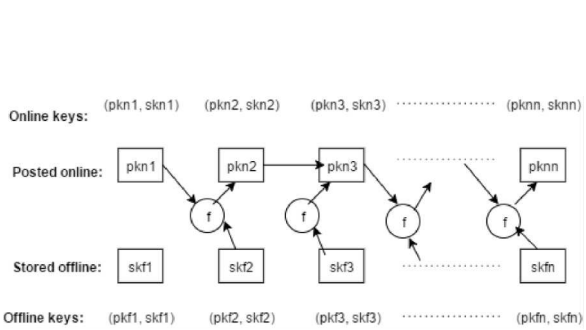
Os trabalhos eleitos foram obtidos por meio da observação dos critérios de inclusão e exclusão e sua pertinência com o tema proposto. O racional deste levantamento considerou a pertinência entre Resumo, Introdução e Conclusão de todos os 555 (475+73+7) trabalhos apontados na Tabela 3.1, resultando na escolha de 16 trabalhos como norteadores para a temática da pesquisa proposta.

3.2 Trabalhos relacionados com ICPs em Blockchain

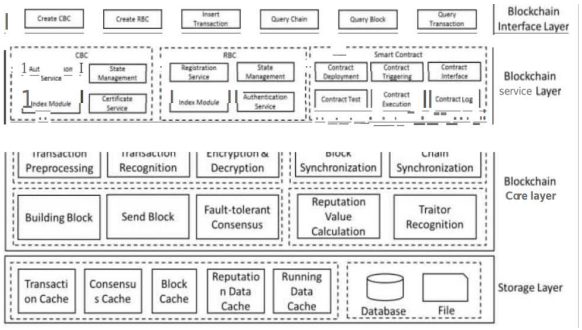
Nesta Seção, apresentam-se 13 trabalhos cujos estudos de ICPs baseadas em Blockchain propuseram arquiteturas, métodos, soluções computacionais, mecanismos e plataformas, mas não criaram implementações das soluções. De igual forma, identificou-se nesta mesma temática estudos que implementaram suas soluções, no entanto estas não estão disponíveis com os seus respectivos contratos inteligentes ou não estão codificados na linguagem Solidity.

3.2.1 *Privacy-awareness in blockchain-based PKI*

O trabalho (AXON, 2015) propõe o desenvolvimento de uma arquitetura de ICP baseada na Blockchain da Namecoin sem vinculação de identidade à chave pública. Nesta



(a) Arquitetura de ICP (AXON, 2015)



(b) Arquitetura do método proposto (WANG *et al.*, 2018)

FIGURA 3.1 – Arquiteturas de ICPs em Blockchain

arquitetura é concedido ao usuário o direito de escolha sobre a divulgação de suas identidades e de suas chaves públicas anteriormente utilizadas. Sendo assim, esta arquitetura se destina aos casos de uso onde se faz necessária a implementação de uma ICP com reconhecimento de privacidade integral ou parcial, podendo ser potencialmente utilizada nas áreas da computação ubíqua, Internet das Coisas, redes veiculares, fóruns e redes anônimos, como é o caso do Tor [1].

A arquitetura fundamenta-se no isolamento do valor de identidade em relação as suas respectivas chaves públicas de curto prazo publicadas na *Blockchain*, ou seja, a identidade e chave pública com reconhecimento de privacidade não devem ser vinculadas publicamente. Isto permite que uma vez determinado o Id de uma identidade, sua chave pode ser atualizada de forma anônima, se esta for a escolha do seu proprietário. A proposta também considera não haver ligação pública entre uma chave pública anterior e a chave pública atualizada. A ligação deve ser oculta, criada a partir das principais atualizações de chaves, como pode ser visualizado na Figura 3.1a.

A Figura 3.1a, ilustra o processo de vinculação de chaves offline desta ICP para uma entidade **E**. Para isto, cabe inicialmente considerar:

- pkf e skf são as chaves públicas e secretas offlines respectivamente; e
- pkn e skn são as chaves públicas e secretas online respectivamente.

A nova chave pública online em cada atualização é calculada em função da chave pública online anterior e da chave secreta offline. Enquanto cada chave pública online de curto prazo (pkn) postada está publicamente desvinculada da última, a entidade **E** mantém os links das chaves off-line, usada na função de atualização da chave online, pelos quais pode-se provar sua propriedade das chaves online passadas e ratificar a conexão entre a chave pública e sua identidade.

[1] https://www.torproject.org/download/

Com este esquema o autor demonstra o poder da arquitetura relacionado ao anonimato e a possibilidade de rastremanento caso seja uma necessidade de negócio, visto a existência da cadeia de chaves públicas online criadas neste processo.

O autor relata que os resultados relacionados à eficiência desta arquitetura apontam que o uso de acumuladores criptográficos traria um significativo aumento de performance para a atualização das chaves desta proposta.

3.2.2 *A privacy-aware PKI system based on permissioned blockchains*

O trabalho (WANG *et al.*, 2018) propõe um método de construção de um sistema de ICP com base em *Blockchains* permissionadas que prevê um esquema de publicação de certificados que consegue a separação entre a camada de registro e camada de autorização do usuário, e tendo por principais características o anonimato e a rastreabilidade condicional, de modo a realizar a proteção da privacidade da identidade do usuário. Desta forma, pretende resolver os problemas envolvendo a confiança mútua entre ACs múltiplas que possuem baixa eficiência na configuração de certificados.

Este método sistematiza a arquitetura de ICP hierárquica baseada em AC e AR por meio de Blockchains. Os papéis das ARs e das ACs são exercidos por estas Blockchains que adotam mecanismos de consenso CBFT (*Concurrent Byzantine Fault Tolerance*), ou protoclo de tolerância a falhas bizantinas conconrrentes que torna mais celere o processo de consenso nas redes que adotam o consenso de tolerância a falhas Bizantinas. O método proposto está ilustrado na Figura 3.1b

Conforme a Figura 3.1b, o modelo prevê quatro camadas, a saber, a camada de armazenamento, a camada central de *Blockchain*, a camada de serviço de *Blockchain* e a camada de interface da *Blockchain*.

A **camada de armazenamento** responde pelos vários armazenamentos de dados em cache e de dados persistentes na *Blockchain*, fornecendo suporte aos bancos de dados Redis, MySQL, HBase, LevelDB, dentre outros.

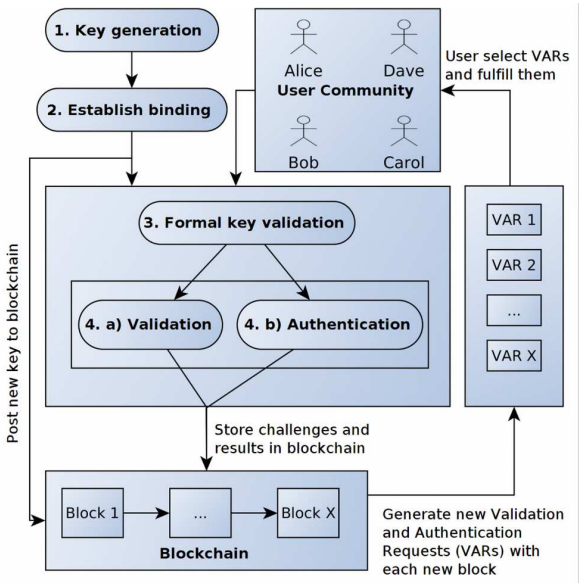
A **camada central** responde pelas operações de *Blockchain*, onde encontra-se o mecanismo de consenso, o mecanismo de reputação, o pré-processamento de transações, a criptografia e a descriptografia, a verificação de assinatura, o gerenciamento de autenticação e etc.

A **camada de serviço** inclui o Registro na *Blockchain* (RBC), o Certificado na *Blockchain* (CBC) e os contratos inteligentes. Sendo que, os nós RBC são responsáveis pela identificação do usuário e pela criptografia das informações de identidade do usuário,

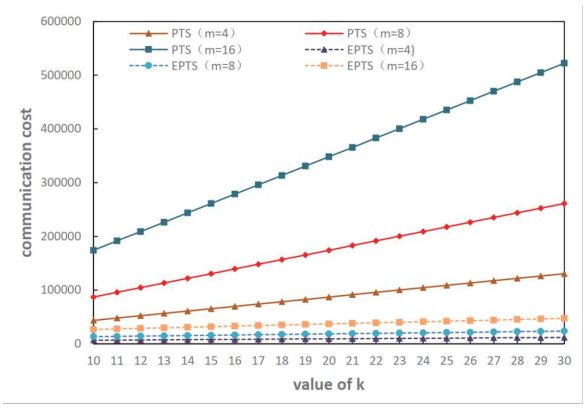
além da guarda dos dados do usuário autenticado após a criptografia. Já os nós CBC são responsáveis pela autenticação de legalidade do usuário, assim como pelos certificados com informações de autenticação e de serviço para usuários, além do armazenamento de certificados digitais e dados de certificados digitais anônimos. Por fim sob os contratos inteligentes são realizadas implantação, execução, acionamento e teste destes contratos.

A **camada de interface** fornece uma interface de serviço entre as aplicações e a *Blockchain* subjacente.

Este método proposto não indica nenhum tipo de sugestão e restrição de implementação, em relação as diversas *Blockchain* privadas existentes.



(a) Visão geral do fluxo de trabalho do Authcoin (LEIDING *et al.*, 2016)



(b) Custo de comunicação (JIANG *et al.*, 2018)

FIGURA 3.2 – Fluxo de trabalho e custo de comunicação

3.2.3 *Authcoin: Validation and authentication in decentralized networks*

O trabalho (LEIDING *et al.*, 2016) propõe uma arquitetura de ICP baseada na tecnologia *Blockchain* adaptável conforme a configuração de um desafio-resposta predefinido para as chaves públicas, certificados e contas de e-mail, utilizado nas etapas de validação e autenticação de chaves públicas, sendo estas etapas a condição suficiente e necessária para que haja uma assinatura entre as duas chaves. A arquitetura proposta encontra-se ilustrado na Figura 3.2a

Conforme ilustrado na Figura 3.2b, a arquitetura proposta permite publicamente o rastreamento das validações e autenticações utilizando-se *Blockchain*. De igual modo, tal tecnologia é utilizada para controlar as chaves, desafios, respostas, assinaturas e todas as

outras informações relevantes das partes interessadas. Igualmente, prevê o impedimento de criação de nós falsos nesta sistemática por meio da detecção destes, pelos nós genuínos existentes nesta rede.

A arquitetura proposta assume ter sido idealizada em uma *Blockchain* privada, mas explicita que é totalmente exequível a sua aplicação em *Blockchain*s como a da Namecoin e da *Ethereum*.

3.2.4 *A Privacy-preserving thin-client scheme in blockchain-based pki*

Outro artigo é o (JIANG *et al.*, 2018) que propõe duas arquiteturas para obter a preservação da privacidade do usuário de dispositivos thin-clients que utilizam uma *Blockchain*.

A primeira arquitetura, *Privacy-preserving Thin-client Scheme* (PTS), propõe que os thin-clients tenham as mesmas funções de dispositivos completos de forma que o usuário autenticado com o thin-client possa ser escondido em uma quantidade preestabelecida **k** de indistinguíveis identidades, provendo assim a sua privacidade.

A segunda arquitetura, *Efficient Privacy-preserving Thin-client Scheme* (EPTS), propõe um esquema de preservação de privacidade com maior ganho de eficiência e redução de custo. Para tal, substitui o método de autenticação praticado com os k indistinguíveis utilizado para privacidade, pelo método de recuperação de informação privada, *Private Information Retrieval* (PIR). A PIR é usada para garantir a privacidade de pesquisa do usuário ao recuperar dados do servidor, desta forma, qualquer pessoa que não seja o usuário, mesmo o próprio servidor de banco de dados, não tem poder de rastrear o conteúdo de pesquisa do usuário.

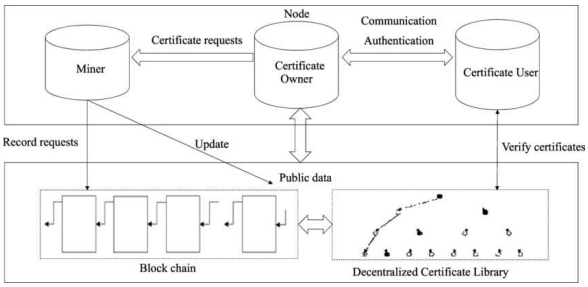
Foi relatado neste trabalho que em uma realização de consulta em PTS, existem **k-1** resultados extras de consulta inúteis retornados, sucedendo-se em um grande desperdício de largura de banda da rede, o que implica em um célere aumento do custo computacional e o custo de comunicação em função do aumento do valor de **k**.

O custo de comunicação entre as partes interessadas nos dois esquemas é insignificante. Já os custos computacionais de thin-client em EPTS é bem próximo ao do PTS. A Figura 3.2b explora bem este cenário relativo aos custos de comunicação.

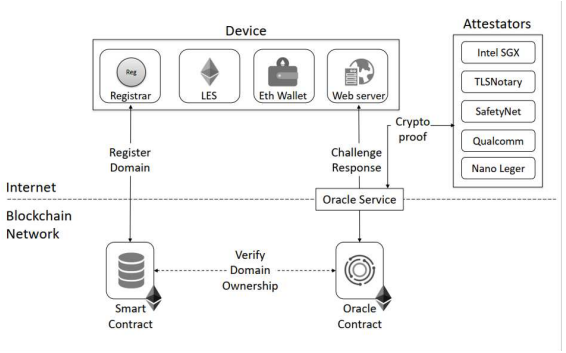
Conforme ilustrado na Figura 3.2b, é possível perceber que os custos computacionais de usuários de nó completo e custos de comunicação em EPTS são inferiores aos custos de comunicação em PTS. Deste fato, os autores inferiram que o EPTS pode ter uma melhora a sua eficiência.

Embora os autores descreveram que adaptaram o modelo de CertCoin para o forneci-

mento de um esboço de uma ICP baseada em *Blockchain*, nada foi mencionado sobre a tecnologia de *Blockchain* adotada no experimento demonstrado na Figura 3.2b



(a) O modelo do sistema Cecoin (QIN *et al.*, 2020)



(b) Arquitetura do sistema proposto (KFOURY *et al.*, 2020)

FIGURA 3.3 – Arquiteturas propostas

3.2.5 *Cecoin: A decentralized pki mitigating mitm attacks*

(QIN *et al.*, 2020) propõem uma arquitetura de ICP fornecendo os serviços de operação de certificados, como registro, revogação, renovação e verificação de certificados, onde estes podem ser tratados como moedas e registrados na cadeia de blocos. Fornece o serviço de certificados múltiplos para garantir a premissa de consistência da identidade e prevenir contra falsos certificados, busca satisfazer o requisito de que uma identidade pode ser usada para registrar vários certificados para diferentes cenários.

Cecoin realiza a recuperação e verificação e operações rápidas de certificados por meio de uma árvore Patricia Merkle modificada, implementando assim uma Biblioteca de Certificados distribuída. Para uma melhor vizualização desta arquitetura recomenda-se observar o ilustrado na Figura 3.3a

Esta arquitetura também permite a transferência de posse de identidade através de um protocolo de justa troca online e, seu protótipo é inspirado no sistema Bitcoin.

O resultado comparativo com outras ICPs, como Instant Karma PKI (IKP), Certcoin, Authcoin e Cecoin, apontou uma eficiência desejável em função do tempo de execução, mas carece de melhorias em relação ao espaço de armazenamento, sobretudo dos nós clientes. Para um protótipo inicial foi utilizada a *Blockchain* do Bitcoin.

3.2.6 *A Blockchain-based Method for Decentralizing the ACME Protocol to Enhance Trust in PKI*

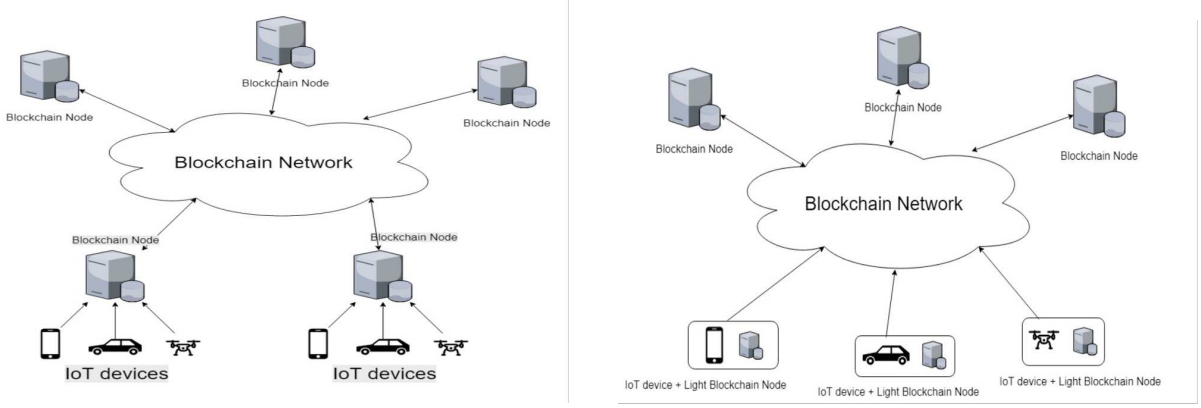
(KFOURY *et al.*, 2020) têm por proposta criar um mecanismo que integre o protocolo de ambiente de gerenciamento de certificado automatizado, *Automated Certificate Management Environment* (ACME), com a tecnologia *Blockchain* no intuito de descentralizar a verificação de propriedade de domínio, eliminando-se assim a necessidade de utilização de uma AC confiável para esta razão. Destaca-se que o padrão ACME é o padrão usado pela AC do *"Let's Encrypt"* para automatizar a emissão de certificados validados por domínio.

A proposta tem por objetivo automatizar a emissão de certificados, minimizar as preocupações de confiança e ainda solucionar indiretamente os percalços atinentes aos certificados obrigatórios, visto que as verificações não são mais realizadas manualmente por AC. Para uma melhor vizualização da arquitetura do mecanismo proposto e seus componentes, recomenda-se observar o ilustrado na Figura 3.3b

O componente ***Device*** é o ativo que necessita de registro de nome de domínio na *Blockchain*, comumente representado como o servidor Web. Este componente é composto por um servidor que precisa registrar seu nome de domínio na *Blockchain*, comumente representado pelo servidor Web. Além deste, o dispositivo é composto por um módulo de software registrador, uma versão do cliente *Ethereum Light Sync* e uma carteira *Ethereum*.

O componente ***Smart Contract*** representa os contratos inteligentes implantados na *Blockchain*, possuindo uma disposição que permite o mapeamento dos nomes de domínio dos dispositivos para seus respectivos registros. O componente ***Oracle Service and Attestators***, situados entre Internet pública e rede *Blockchain*, colaboram com os contratos inteligentes interfaceando com a Internet os pedidos de coleta e postagem de informações dos contratos inteligentes mediante a uma prova de autenticidade, que sintetiza a não adulteração de dados fundamentada em base criptográfica.

Para consecução da implementação da solução e dos testes avaliativos foram utilizadas a rede de teste da *Blockchain Ethereum* denominada Ropsten e a linguagem Solidity. Já a Interface de Programação de Aplicativos (API) Web3j foi empregada para conectar aplicativos que rodam na Máquina Virtual Java (JVM) a *Blockchains Ethereum*, sem onerar o processamento. Por fim, o trabalho realizou sua análise de desempenho baseada na realização das transações em função do *gas* e do custo em dólar, mostrando-se apropriada para estes fins.



(a) (b)

FIGURA 3.4 – Arquiteturas das soluções de ICPs (SINGLA; BERTINO, 2018)

3.2.7 *Blockchain-Based PKI Solutions for IoT*

(SINGLA; BERTINO, 2018) é um artigo que apresenta três soluções de ICPs para dispositivos voltados à Internet das Coisas, *Internet of Things (IoT)*, baseadas na tecnologia *Blockchain* e as compara com uma solução de ICP tradicional baseada em AC.

Os autores na apresentação de suas soluções as agrupam em duas categorias. Na primeira, considera-se uma arquitetura IoT que necessita de um nó remoto, confiável e completo, ou seja, que mantenha uma cópia atualizada dos dados da *Blockchain*, e que esteja integrado ao ecossistema IoT associado. Ressalta-se que esta categoria sintetiza duas das três soluções propostas, onde respectivamente são implementadas por meio do *Name Value Service* (NVS) da Emercoin (KONASHEVYCH, 2018) e que utiliza os contratos inteligentes da *Blockchain Ethereum*. Para uma melhor visualização desta solução proposta recomenda-se observar o ilustrado na Figura 3.4a.

Na segunda categoria não se faz necessário utilizar um nó remoto que faça o papel de uma Terceira Parte Confiável, *Trusted Third Party* (TTP), para integração do ecossistema IoT à *Blockchain*, pois sua proposta consiste em integrar no dispositivo IoT uma versão do cliente *Ethereum Light Sync*, permitindo desta forma que dispositivos com recursos de hardware limitados iterajam com a *Blockchain Ethereum*, conforme ilustrado na Figura 3.4b.

Em relação a análise de desempenho e comparação, foram estabelecidos como parâmetro de avaliação o tempo de execução, o espaço de armazenamento, o requisito de confiança, o custo para adicionar um certificado, a flexibilidade de armazenamento e o tempo para emissão de certificados. Em relação ao tempo de execução no lado do servidor a abordagem em AC é superada pelas abordagens em *Blockchain* e o mesmo acontece no lado dos dispositivos IoT.

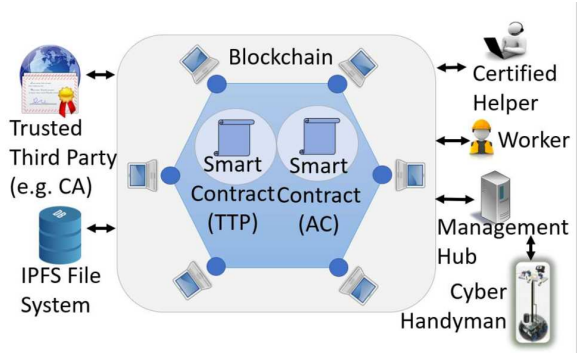
O espaço de armazenamento das ICPs baseadas em AC pode variar em função da

dependência do uso da LCR. No que tange às ICPs baseadas em *Blockchain*, a abordagem fundamentada sobre a NVS Emercoin possui o menor espaço, seguido da abordagem *Ethereum Light Sync* e, por último, da *Ethereum Fast Sync*. Já nos dispositivos IoT o espaçamento em disco tem outro comportamento, a abordagem Ethereum "Light" Sync ocupou o mesmo espaço que ocupou na *Blockchain*, enquanto a abordagem de nó remoto Emercoin e Ethereum não ocupam espaço.

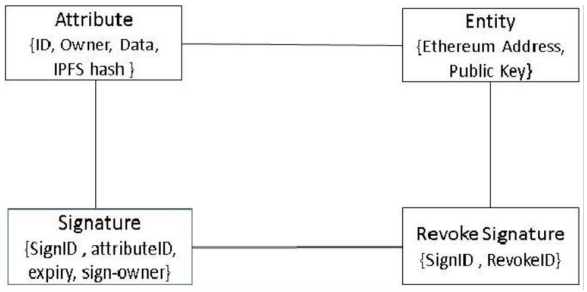
O requisito de confiança em uma ICP baseada em AC está na confiança da própria AC. No caso das ICPs baseadas em *Blockchain* não há confiança em um elemento central que se deve confiar. Mas em relação ao ecossistema de IoT, os nós remotos que hospedam a *Blockchain* devem ser fidedignos para fornecer os dados corretos. Cabe ressaltar que na abordagem Ethereum "Light" Sync não existe a necessidade de TTP.

Quanto ao tempo para emissão de certificados e ao custo para adicionar um certificado, as ICPs baseadas em *Blockchain* mostraram-se substancialmente mais vantajosas do que na abordagem de ICPs baseadas em AC. Por fim, a flexibilidade de armazenamento em ICP baseada em AC está condicionada à regulação do padrão X.509. Em relação às ICPs baseadas em *Blockchain*, existem possibilidades para armazenamento de informações adicionais como a guarda de endereço do proprietário do dispositivo na *Blockchain*, ou até mesmo a concessão de criação de estruturas de armazemento de dados complexas via contratos inteligentes, utilizando-se a Ethereum para tal.

Cabe ressaltar que o trabalho aborda a temática da garantia da retenção de identidade situando a ICP baseada em *Blockchain* no avançar de mais um desafio não resolvido pelas ICP baseada em AC e WoT.



(a) Arquitetura (WIDICK *et al.*, 2019)



(b) Diagrama de funções de contratos (SIVAKUMAR *et al.*, 2017)

FIGURA 3.5 – Arquiteturas das soluções

3.2.8 *Blockchain Based Authentication and Authorization Framework for Remote Collaboration Systems*

(WIDICK *et al.*, 2019) apresentaram um trabalho com a proposta de criação de uma estrutura de colaboração entre os processos de autenticação e autorização suportados pela tecnologia *Blockchain* para o controle de acesso aos recursos de dispositivos IoT.

A estrutura proposta contempla um sistema de colaboração remota de baixo custo, chamado *"Cyber Handyman"*, que consiste em um dispositivo inteligente que visa facilitar as tarefas diárias das pessoas, e este utiliza um dispositivo IoT para a obtenção de assistência remota. O auxiliar remoto pode possuir o acesso a todos os recursos via Internet, tais como webcam de alta definição, luzes de LED, rodas, alto-falante e microfone para a efetiva orientação do trabalhador que tem de realizar as tarefas manuais.

No intuito de não permitir a violação das regras de privacidade do trabalhador faz-se necessário a aplicação de mecanismos de autenticação e autorização nesta estrutura. Cabe ressaltar que o módulo proposto de autorização para sistemas de colaboração remota foi baseado em OAuth, contemplando a substituição do servidor de autorização centralizado por dois contratos inteligentes que executem estas demandas. Para uma melhor vizualização desta arquitetura e seus componente recomenda-se observar o ilustrado na Figura 3.5a.

O componente ***Worker*** representa as pessoas que carecem de assistência, sendo identificado pelo endereço público de sua conta Ethereum. Destaca-se que cada ***Worker*** pode registrar apenas um dispositivo ***Cyber Handyman***.

O componente ***Helper*** representa os especialistas de cada área de assistência. Apenas ajudantes certificados podem se registrar como ajudantes. Os ajudantes são igualmente identificados pelo endereço público de sua conta Ethereum.

O componente ***Smart Contract*** é composto por dois contrato inteligente, onde um trata com as operações de certificados digitais, representado em **TTP *Smart Contract*** e o outro lida com o controle de acesso, representado em **AC *Smart Contract***.

O componente ***Trusted Third Party*** *(TTP)* é uma entidade que emite certificados digitais que identificam e qualificam o componente ***Helper***.

O componente ***Management Hub*** representa uma interface entre a rede *Blockchain* e o componente ***Cyber Handyman***, e este componente por sua vez é um dispositivo com múltiplos recursos que viabiliza a colaboração do componente ***Helper*** com o componente ***Worker*** por meio da consciência situacional obtida por esse aos componentes ***Helper*** e ***Worker***.

O componente ***InterPlanetary File System*** *(IPFS)* é um ativo usado para arma-

zenar e compartilhar dados em um sistema de arquivos distribuído. Na proposta deste trabalho o IPFS fica responsável por armazenar a maioria dos dados, ou seja, armazenamento de dados fora da cadeia de blocos, cabendo a este último, somente o armazenamento de metadados. Assim, preserva-se a característica transparente da *Ethereum Blockchain*, em busca de uma eficiência financeira para este arquétipo.

Ressalta-se que, além dos componentes presentes na figura, o artigo também aborda o compoente ***Reviewer*** que avalia as evidências ofertadas pelo componente ***Helper*** durante a etapa de registro.

Para consecução da implementação da solução e os testes avaliativos foi utilizado a rede de teste da *Ethereum Blockchain* denominada Ropsten e linguagem Solidity. Foram realizados testes de escalabilidade utilizando-se como parâmetro a mudança no número de usuários, na quantidade de campos exigidos nos certificados e na quantidade de evidências simultâneas que podem ser enviadas. Ao todo forma cridas 106 contas. Ressalta-se que os resultados em função do número de usuários demonstraram um consumo de *gas* que varia de uma ordem linear para uma quadrática.

Outro resultado relevante apontado, foi que esta aplicação está limitada a lidar com cerca de 25 ***Helpers*** antes de uma escala de preços exponencial do *gas*. Para tanto, os autores mediram quanto os custos de uso de seus contratos escalariam à medida que o número de ***Helpers*** simultâneos aumentassem.

3.2.9 *Privacy based decentralized Public Key Infrastructure (PKI) implementation using Smart contract in Blockchain*

(SIVAKUMAR *et al.*, 2017) propuseram uma solução de ICP baseada em *Blockchain* que prevê o tratamento de privacidade por meio da tradução de endereço de chaves públicas e da ofuscação de contratos inteligentes.

A tradução de endereços de chaves públicas consiste em gerar um conjunto de chaves públicas distintas originadas da mesma chave privada assegurando a cada transação ICP uma nova chave pública relacionada à chave privada do nó Ethereum. Tal medida impõe anonimato e privacidade, à medida que o controle dos dados armazenados na *Blockchain* é realizado através de eventos de contratos inteligentes, sendo estes então, uma caixa preta para possíveis invasores.

Já a ofuscação de contratos inteligentes é pautada na encriptação do armazenamento dos dados na *Blockchain* e no ofuscamento dos métodos e eventos dos contratos inteligentes. Para uma melhor vizualização desta solução de ICP e seu respectivo contrato inteligente, recomenda-se observar o ilustrado na Figura 3.5b.

A arquitetura prevê que as operações de ICPs sejam implementadas na plataforma Ethereum como funções de contratos inteligentes escritas na linguagem Solidity. Conforme ilustrado na Figura 3.5b, as entidades são retratadas por meio de chave pública e endereço ethereum. Sendo que cada entidade pode ter vários atributos para identificar o proprietário da entidade. De igual modo, estabelece que os eventos e funções associados às operações de ICPs são definidos no contrato inteligente que invocam as operações de ICPs específicas. A proposta contempla as seguintes operações:

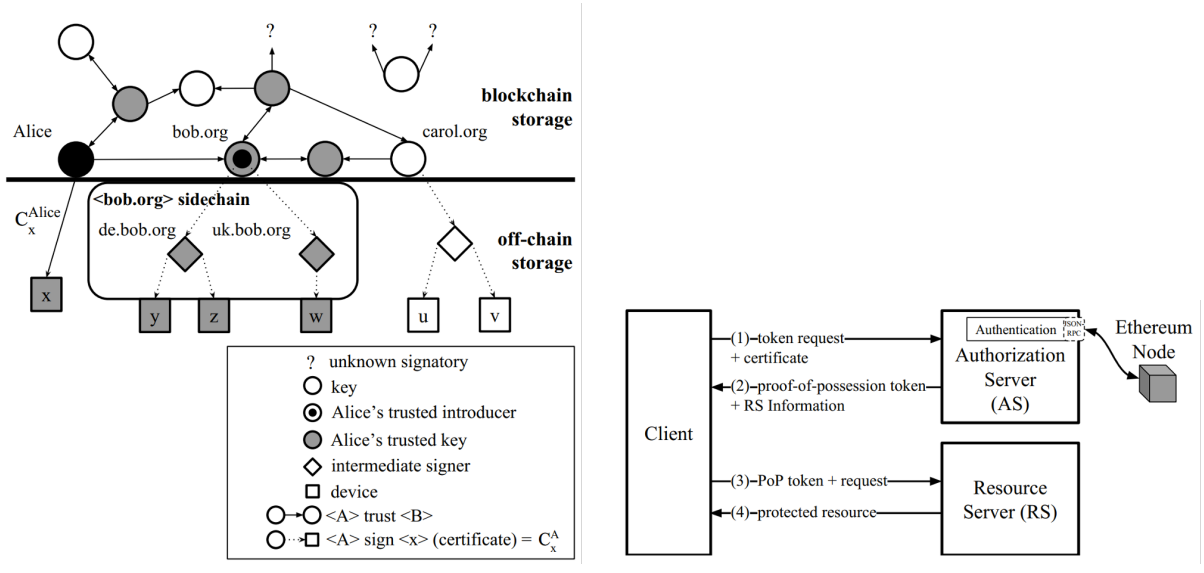
Registration of Entity – Proporciona a inclusão dos usuários no sistema de ICP invocando o evento de registro do contrato inteligente, este por sua vez coleta os detalhes pertencentes à entidade e os encaminha no formato de transação para a *Blockchain*. Cada entidade é mapeada com um id gerado automaticamente emitido por contrato inteligente.

Signing of Attributes – Proporciona a assinatura dos atributos coletados na operação de *Registration of Entity* pelo sistema por meio de contrato subjacente e por meio de uma transação. Ressalta-se que após esta assinatura as entidades podem ser disponibilizadas como fidedignas por outras entidades ou usuários.

Retrieval of Attributes – Proporciona a consulta na *Blockchain* por meio dos Ids gerados pelos contratos inteligentes.

Revoke Signature – Proporciona a revogação das assinaturas em atributos ou entidades assinadas anteriormente assinadas por meio de contrato inteligente que invoca o evento de revogação e publica tal ato como uma transação.

Embora este trabalho relata possuir uma implementação em Ethereum, não foi encontrada a efetiva disponibilização do contrato inteligente a ela associada.



(a) (b)

FIGURA 3.6 – Modelo de rede de confiança descentralizada e fluxo de autenticação (DURAND *et al.*, 2017)

3.2.10 *Decentralized Web of Trust and Authentication for the Internet of Things*

(DURAND *et al.*, 2017) propuseram a construção de um sistema escalonável para IoT baseado em *Blockchain*, visando igualmente compatibilizar uma estrutura de autorização e autenticação emergente da web para ambientes com restrições.

A arquitetura proposta contempla a colaboração de uma rede de confiança juntamente com elementos hierárquicos, a fim de prover a escalabilidade de guarda dos certificados na *Blockchain*. Para uma melhor vizualização da arquitetura do sistema proposto e seus componentes, recomenda-se observar o ilustrado na Figura 3.6a.

Nesta arquitetura as chaves públicas são expostas por signatários que podem ser indivíduos, (como Alice) ou organizações (como bob: org, carol: org). Tais chaves são utilizadas para realizar a assinatura dos certificados dos dispositivos, podendo ser executada por quaisquer membros da cadeia (pessoas) ou por agentes / AC intermediários. A publicação é efetivada em uma rede *Blockchain* não permissionada ou pública. Cabe ressaltar que, para melhor otimizar o espaçamento em disco na cadeia de blocos, apenas as relações de confiança são salvas na *Blockchain*, prevendo-se como opção de transparência para empresas maiores a publicação de certificados nas chamadas sidechain.

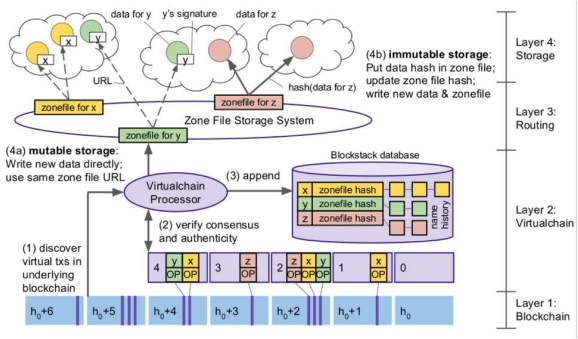
Para as etapas de autorização e autenticação, esta arquitetura fomenta para uso em IoT a adoção de ferramentas que utilizem o padrão *Authentication and Authorization for Constrained Environments* (ACE) em estudo. A concessão de aprovação de acesso a um recurso proposta nesta arquitetura utiliza o fluxo de autenticação constante na Figura 3.6b.

Com esta arquitetura é possível autenticar dispositivos de organizações externas fundamentado no conceito Web of Trust e, para isto, criou-se um protótipo na rede *Ethereum*.

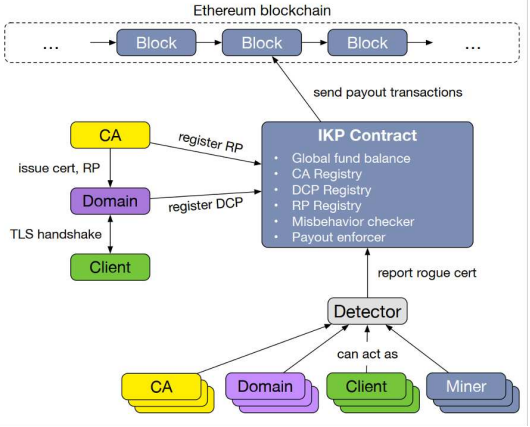
Embora este trabalho tenha mencionado uma implementação em seu protótipo não relatou a realização de experimentos.

3.2.11 *Blockstack: A Global Naming and Storage System Secured by Blockchains*

(ALI *et al.*, 2016) propõem a implementação de uma solução de sistemas de nomeação e armazenamento baseado na tecnologia *Blockchain* do Bitcoin. Tem por diferencial em relação aos sistemas baseados em *Blockchain* antecessores, considerar a separação entre controle e plano de dados, tendo como premissa manter somente os metadados mínimos, ou seja, hashes de dados e transições de estado na *Blockchain*. Além disso, usa armazenamento de dados externos para salvar dados em massa.



(a) Visão geral da arquitetura do Blockstack (ALI *et al.*, 2016)



(b) Visão geral das entidades e funções no IKP (MATSUMOTO; REISCHUK, 2017)

FIGURA 3.7 – Visão geral das soluções (DURAND *et al.*, 2017)

Outra característica que o diferencia de outros sistemas que utilizam a tecnologia *Blockchain* do Bitcoin, se deve ao fato de não fazer uso de bifurcação da *Blockchain*. No entanto, utiliza esta *Blockchain* subjacente para vinculação de nomes a registros de dados via mecanismo de consenso, comunicando as mudanças de estado nos novos blocos da *Blockchain*.

Ressalta-se que a Blockstack originalmente foi concebida sobre a *Blockchain* da Namecoin, no entanto devido a questões de segurança envolvendo a desproporcionalidade de poder computacional em determinados mineradores, ou a insuficiência de poder computacional para suporte a várias *Blockchains* seguras, optou-se por migrar da Namecoin para a Bitcoin.

Sua arquitetura está fundamentada em 4 camadas, são elas: *Blockchain*, virtualchain, rotemaneto e armazenamento. Para uma melhor vizualização desta solução, recomenda-se observar o ilustrado na Figura 3.7a.

A camada de *Blockchain* codifica as operações da Blockstack em transações de *Blockchain*, forncendo consenso e ordem das operções deste sistema. A camada virtualcahin define as novas operações que são codificadas como metadados adicionais em transações de *Blockchain* válidas, desta forma separa-se a visão dos dados, cabendo a virtualchain a visualização da lógica de processameno das operações e aos nós as transações brutas, comuns as tecnologias Blockchcains. A camada de roteamento cria uma sepação entre a solicitação de roteamento do armazenamento real dos dados permitindo assim coexistência de vários sistemas de amarzenamento conforme a necessidade. Já a camada de armazenamento é responsável por hospedar os dados reais vindos da Blockstack, sendo estes assinados pela chave do respectivo proprietário no Sistema de nomeação. Desta forma pode-se guardar valores de tamanho arbitrário e proveniente de distintas plataformas de

armazenamento.

Com esta arquitetura, Blockstack tem uma inicialização mais rápida de novos nós e preserva as atualizações de mais lentas de dados fora da rede *Blockchain*, condições desejávies para construção de serviços descentralizados que utilizam uma infraestrurura publicamente disponível.

Quanto a seu desempenho mostrou-se compatível com o serviço de armazenamento e sendo necessário algum incremento de sobrecarga de CPU.

Blokcstack é um emblemático estudo de caso, onde visualiza-se a necessidade de observar as reais capacidades da tecnologia *Blockchain* subjacente para criação adequada de DApps.

3.2.12 *IKP: Turning a PKI Around with Decentralized Automated Incentives*

(MATSUMOTO; REISCHUK, 2017) propõem uma plataforma que expande a arquitetura de ICP baseada em AC permitindo a automatizações do registro comportamental de um certificado, sendo um notável caso de uso voltado a melhoria de transparência de certificados, onde se tem o registro de certificados não autorizados, incentivos ao bom comportamento de AC e ao relato de certificados potencialmente não autorizados.

Objetiva também caracterizar a emissão incorreta de certificados por meio das chamadas Política de Certificados de Domínio e possibilitar a detecção de relatos de certificados falsos. Para atingir este objetivo faz-se necessário o uso de contratos inteligentes via *Blockchain* Ethereum, permitindo que os proprietários de domínio definam as ACs de sua confiança. Para uma melhor vizualização desta plataforma recomenda-se observar o ilustrado na Figura 3.7b.

Ao receber um certificado como entrada provido por um detector, componente responsável por reportar a presença de certificados suspeitos à autoridade IKP, este examina sua conformidade junto à Política de Certificados de Domínio que define a lista de ACs aprovadas para emissão de certificados para este domínio específico. Caso o certificado seja emitido por uma AC ausente na lista supracitada, executa-se a Política de Reação que efetua uma operação de garantia (uma espécie de seguro) de transferência de Ether da AC, cujo comportamento foi inadequado para o usuário impactado, juntamente com o detector que relatou a infração.

Neste trabalho foram realizadas avaliações relacionadas ao custo de implantação da IKP na *Blockchain* Ethereum. Enfatizou-se a viabilidade técnica e econômica da plataforma, de modo que não havendo grandes oscilações no preço do *gas*, no limite de *gas* ou

no preço do Ether, a solução julga-se factível. No entanto, cabe ressaltar que o link para acesso da referida implementação não está disponível. Este trabalho contribui sobremodo na transparência dos certificados, contudo ainda contempla ICPs baseadas em ACs, um dos desafios impostos ao ambiente de ICP que merece ser vencido.

3.2.13 *A Decentralized Public Key Infrastructure with Identity Retention*

(FROMKNECHT *et al.*, 2014) propõem uma solução completa de ICP desenvolvida na *Blockchain* da Namecoin ofertando a garantia de retenção de identidade. Para tal, é encaminhada uma transação contendo a tupla (chave, IDpublic) juntamente com a função a ser executada para a *Blockchain*, por exemplo registro, atualização, busca, verificação e revogação. Deste ponto, seguem-se os processos de mineração e validação da *Blockchain* que culminarão no resultado da transação.

O Certcoin possui 3 versões, cada uma com suas especificidades de complexidade e tempo de processamento. A primeira versão (version 0), conhecida por ser conceitualmente a mais simples, envolve um número vultuoso de operações, sendo a única versão que contempla as cinco funções supracitadas.

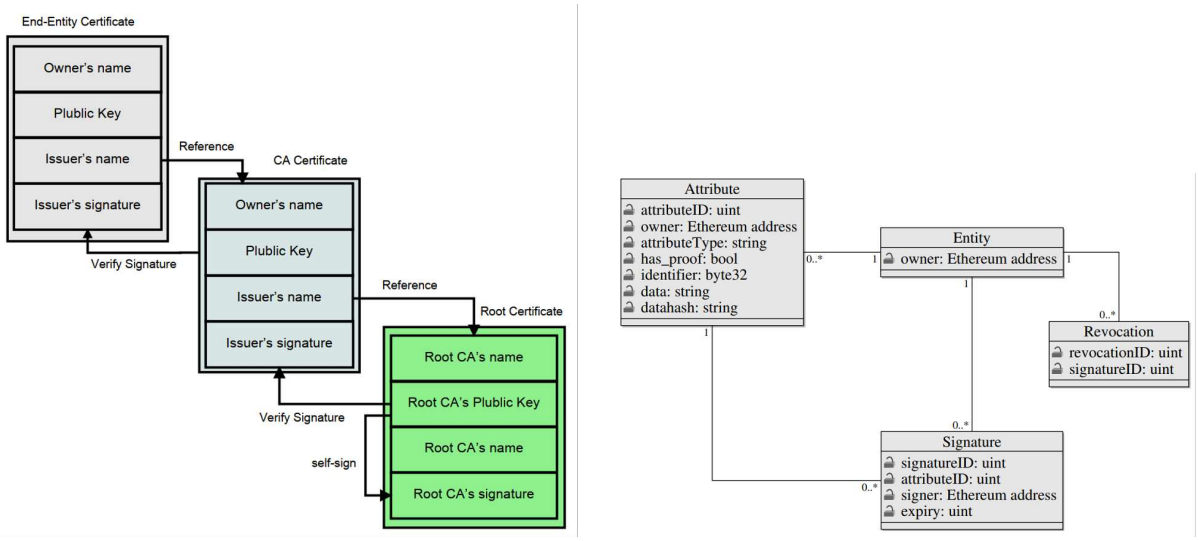
As versões 1 e 2 investem na simplificação das funções de atualização, busca e verificação por meio de acumuladores criptográficos e DHTs respectivamente, proporcionando uma melhora no processamento e diminuição de seu tempo. Logo, com a utilização de acumuladores criptográficos, tem-se a redução do tempo e da complexidade envolvidos no processo de verificação. De igual forma, a utilização DHTs traz como valor agregado a realização de pesquisas rápidas para as consultas de chaves públicas.

3.3 Artigos relacionados com Provas de Conceito de ICP Descentralizada

Nesta Seção, apresentam-se dois trabalhos cujos estudos de ICPs baseadas em *Blockchain* propuseram e disponibilizaram suas soluções em linguagem Solidity.

3.3.1 *A Blockchain-Based PKI Management Framework*

(YAKUBOV *et al.*, 2018) propõem um framework para gerência de ICP baseada em *Blockchain* por meio da criação de um certificado híbrido, que faz a integração de metadados de *Blockchain* nos campos de extensão do certificado padrão X.509 versão 3,



(a) Cadeia de confiança (YAKUBOV *et al.*, 2018)

(b) Diagrama entidade-relacionamento so SCPKI (AL-BASSAM, 2017)

FIGURA 3.8 – Visão geral das soluções

permitindo assim a validação de certificados pela tradicional cadeia de confiança, ou por uma estrutura de ICP baseada em *Blockchain*. Para uma melhor vizualização deste framework, recomenda-se observar o ilustrado na Figura 3.8a.

Quanto a arquitetura, este artigo estabelece certificados para ACs Raiz, ACs Intermediárias e para o usuário final. Igualmente, estabelece um contrato inteligente para cada AC objetivando o gerenciamento dos certificados emitidos, o mapeamento das informações de revogação e a validação da cadeia de confiança.

Já a sua proposta de implementação resume-se em três pilares, um serviço Restful, a validação de certificados e uma interface básica de serviço Web usada para teste.

O Serviço Restful foi desenvolvido em Golang, como sendo um servidor web autônomo que fornece acesso ao *Blockchain* público Ethereum para a execução das funções de emissão, revogação e validação dos certificados e fornece como funcionalidades principais, o registro de usuários, o gerenciamento dos certificados revogados, a criação e preenchimento dos contratos inteligentes associados as ACs e a validação da cadeia de confiança.

A validação de certificados se dá por meio de um contrato inteligente que valida o caminho de certificação de um determinado certificado, ou seja, do certificado até a AC que o emitiu. Já a interface Web permite a realização de testes como a adição de certificados de ACs e do usuário final em diferentes níveis da cadeia de certificação e em diferentes contas de ACs, além da realização de revogação de certificados.

Cabe ressaltar que a solução trabalha com dois métodos distintos para a realização da validação. A validação da cadeia de confiança pode ser realizada pelo serviço Restful ou separadamente por meio da chamada de contrato inteligente instruída no parágrafo

anterior.

A avaliação de desempenho foi realizada comparando os dois distintos métodos de validação da cadeia de confiança. Para a consecução desta avaliação no ambiente *Blockchain*, foi utilizada a Rinkeby, rede de teste pública da Ethereum que utiliza o mecanismo de consenso *Proof of Authority* (PoA). Os resultados desta avaliação apontaram para a relevância de construir sistemas robusto de ICP em *Blockchain*, visto que esta abordagem tem desempenho plausível e custos de manutenção atrativos. Por exemplo para se obter a validação de grandes cadeias de confiança o tempo despendido na solução em Golang foi maior do que na solução em Ethereum e o custo associado à ICP baseada em *Blockchain* também mostrara-se menor do que de uma ICP baseada em AC.

Ressalta-se que, embora este trabalho apresente codificação em Solidity, o mesmo não foi reaproveitado na presente pesquisa para auxiliar as medições acerca das hipóteses, visto a dissimetria das funções entre os trabalhos.

3.3.2 *SCPKI: A Smart Contract-based PKI and Identity System*

(AL-BASSAM, 2017) propõe o desenvolvimento de um protótipo simplificado com funcionalidade para a operação de ICP e sistema de gerenciamento de identidade, chamado *Smart Contract-based PKI and Identity System* (SCPKI). O sistema é baseado em dois componentes principais, a saber, um contrato inteligente que rege o protocolo do sistema e atua como uma interface para a *Blockchain* quanto ao gerenciamento de identidades e atributos e, de um cliente que possibilita a interação do contrato inteligente com outros sistemas, como por exemplo o IPFS. Desta forma é possível a realização de pesquisas e filtragem de atributos por parte de usuário.

Para uma melhor visualização deste protótipo, recomenda-se observar o diagrama entidade-relacionamento referente ao contrato inteligente do SCPKI na Figura 3.8b.

Conforme pode-se observar na Figura 3.8a, o contrato inteligente publica um conjunto de atributos, assinaturas e revogações na *Blockchain*.

O SCPKI apresenta-se com duas versões de contratos inteligentes, sendo uma versão projetada para ser programaticamente utilizada por outro contrato e uma outra versão, onde tal feito não é possível.

Foi discutido o custo de *gas* em função do tamanho das entradas de dados, bem como em função das operações dos contratos inteligentes. Os resultados indicaram que o SCPKI é viável segundo os parâmetros aqui citados. O código do SCPKI foi reaproveitado no trabalho para auxiliar em medições acerca das hipóteses.

3.4 Artigo relacionado ao desempenho de redes de testes da Ethereum

Esta Seção apresenta um trabalho que realiza uma avaliação experimental de desempenho do ambiente *Ethereum*, utilizando três redes de testes públicas que apresentam forte semelhança com a rede principal da *Ethereum*.

(ZHANG *et al.*, 2019) propõem avaliar a performance da plataforma *Ethereum* através da análise dos seguintes parâmetros:

1. Latência de transação de consulta em saldo em conta;
2. Tempo de Geração do bloco; e
3. Latência de aceitação da transação desde a origem até o destino.

Para a realização dos experimentos propostos foram estabelecidos quatro níveis de cargas de transações: com apenas uma transação, com 25 transações, com 50 transações e com 100 transações. Cada teste realizado foi repetido 10 vezes para garantir a confiabilidade dos resultados.

Esse trabalho faz uma breve contextualização da rede *Ethereum* e de seus elementos, como blocos, transações, limite de *gas* e os aspectos temporais relacionados ao bloco.

De forma sucinta elenca as características das redes de testes utilizadas nos experimentos e, de igual modo, explica os mecanismos de consenso a estas relacionados.

Embora tenha apresentado fielmente os resultados conforme previsto em seu escopo e conforme pode ser visto na Figura 3.9, por meio de seu experimento de maior carga, com a finalidade de se obter os resultados mais conclusos, as redes de experimentos carecem de uma análise mais abrangente e aprofundada. Outro fator de relevância que poderia ser considerado nos testes que trariam mais realismo, seria realizar experimentos com transações que exigem mais do ambiente de testes, tais como transações de ICP.

Por fim, para aumentar o nível de factibilidade da experimentação, seria desejável a realização de testes com nível concorrêncial maior, a fim de obter medições mais próximas dos garaglos desse ambiente.

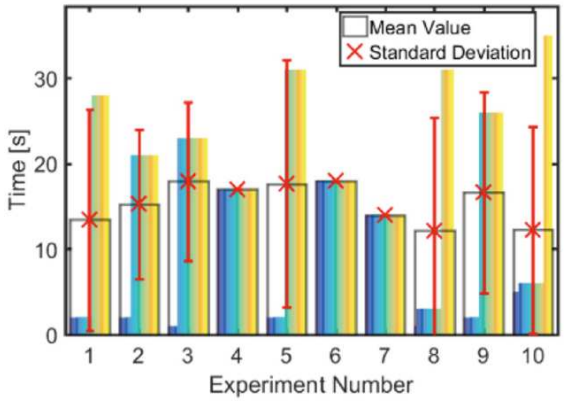


FIGURA 3.9 – Rinkeby com 100 Transações concorrentes (ZHANG *et al.*, 2019)

3.5 Comparativo do Trabalho com o Estado da Arte

Esta Seção tem por objetivo apresentar um panorama dos requisitos que têm influenciado os caminhos de pesquisa na temática de ICPs baseadas em *Blockchain*. Os requisitos escolhidos foram extraídos do conjunto inicial pesquisado na fase do levantamento sistemático e reunidos, de forma matricial, na Tabela 3.2 para facilitar o entendimento.

Para tanto, convém explicar cada um desses requisitos:

1. ***Blockchain*** - Refere-se à tecnologia *Blockchain* escolhida para a consecução dos trabalhos relacionados;

2. **Codificação disponível** - Embora boa parte destes trabalhos tenham desenvolvido pelo menos um protótipo, apenas três disponibilizaram e mantiveram disponíveis a fonte de suas implementações. Para este trabalho, este requisito não foi apenas um recurso de acessibilidade, mas sim um critério de seleção por meio do qual pôde-se observar a real pertinência dos algoritmos que serão comparados no Capítulo 5. Outro fator de grande importância está na possibilidade de realizar uma verificação de segurança no código disponibilizado;

3. **Escalabilidade** - Dos trabalhos apresentados ainda se tem um número pouco expressivo sobre estudos de escalabilidade em torno de ICPs baseadas em *Blockchain*. No entanto, tal característica é de fundamental importância para projetos de DApps;

4. ***Testbed*** - Prezando pelo rigor científico observado na condução das pesquisas experimentais na área de computação e para que se constate um ambiente transparente e replicável nos estudos de novas tecnologias, foi estabelecido como característica a ser observada nos trabalhos relacionados observado a construção de *Testbed*; e

5. **Desempenho** - A análise de desempenho é a característica mais presente nos trabalhos apresentados. Por meio desta análise, torna-se possível obter uma melhor caracterização do ambiente em função dos serviços realizados, de seus clientes e suas respectivas requisições, por exemplo.

TABELA 3.2 – Tabela Comparativa de Requisitos

Trabalhos Relacionados	Requisitos				
	Blockchain	Codificação Disponível	Escalabilidade	*Testbed*	Desempenho
(AXON, 2015)	Namecoin				
(WANG et al., 2018)	*Blockchain* Privada				
(LEIDING et al., 2016)	*Blockchain* Independente				
(JIANG et al., 2018)	CertCoin				X
(QIN et al., 2020)	Bitcoin				
(KFOURY et al., 2020)	Ethereum				X
(SINGLA; BERTINO, 2018)	Emercoin e Ethereum				X
(WIDICK et al., 2019)	Ethereum		X		X
(SIVAKUMAR et al., 2017)	Ethereum				
(DURAND et al., 2017)	Ethereum				
(ALI et al., 2016)	Bitcoin e Namecoin		X		X
(MATSUMOTO; REISCHUK, 2017)	Ethereum				
(FROMKNECHT et al., 2014)	Namecoim				X
(YAKUBOV et al., 2018)	Ethereum	X			X
(AL-BASSAM, 2017)	Ethereum	X			X
(ZHANG et al., 2019)	Ethereum				X
Esta Pesquisa	Ethereum	X	X	X	X

Conforme disposto na Tabela 3.2, de forma geral, os trabalhos relacionados investem em análise de desempenho, mas ainda assumem uma postura inexpressiva em relação à construção deste tema para as próximas gerações de pesquisadores. De todos os traba-

lhos, apenas este trabalho de pesquisa construiu uma *testbed*, onde se torna possível a perpetuação de testes de forma a ampliar os horizontes de pesquisa para realização de novas aplicações e arquiteturas da forma mais realística possível.

3.6 Síntese do Capítulo 3

Neste Capítulo, foi realizada uma revisão sistemática sobre o estado da arte da literatura de modo a categorizar os conteúdos que deram suporte a esta pesquisa. Esta revisão resultou em um levantamento bibliográfico composto por 16 trabalhos compreendendo arquiteturas, métodos, soluções computacionais, mecanismos, plataformas, codificação de contratos inteligentes, inclusive na linguagem Solidity e a realização de avaliação de desempenho.

Em uma investigação mais refinada foi possível constatar uma lacuna (*gap*) no estudo de análise de desempenho mais expressivo e a falta de construção de *testbeds* que possibilitem perenizar estudos experimentais neste tema com arquiteturas mais realísticas, como a que será apresentada no próximo Capítulo.

4 Metodologia Aplicada e Caracterização do Ambiente

Como foi explicado na Introdução, o núcleo deste trabalho tem como objetivo investigar a capacidade do uso de *Blockchain* no atendimento aos requisitos de entidades certificadoras digitais, garantindo resiliência, segurança e escalabilidade, utilizando-se desta nova plataforma. Para tanto, foi examinado de maneira minuciosa o potencial na perspectiva de desempenho dos contratos inteligentes de ICP em *Blockchain*, por meio de testes de desempenho de contratos inteligentes em um ambiente experimental factível e fidedigno para esta avaliação.

Tal avaliação foi realizada em várias fases e será descrita ao longo deste Capítulo. Em primeiro lugar, são descritas as justificativas na escolha da plataforma empírica e experimental *Rinkeby*, a escolhida para este estudo de desempenho. Em seguida, explica-se o desenvolvimento do ambiente automatizado de testes, bem como todas as ferramentas utilizadas no trabalho e também a metodologia e sistemática dos experimentos realizados. Dada a necessidade de se trabalhar com um ambiente, ao mesmo tempo realístico emulando a rede Ethereum real, mas que pudesse produzir pouco ruído nos testes de desempenho, foi realizada uma ampla caracterização da rede *Rinkeby* em termos de número de nós ativos e tamanho, distribuição geográfica dos nós e volume de transações.

Em seguida, conclui-se a parte principal deste estudo realizando-se uma sequência de experimentos repetitivos, com quantidades variadas de usuários concorrentes escalonados, para medir o desempenho de contratos inteligentes de ICP. Estas medidas permitem obter a perspectiva apropriada sobre a capacidade da *Blockchain* em atender as demandas de ICP de maneira decentralizada no mundo real. Para avaliar o comportamento de DApps de ICP, foram utilizados e adaptados três contratos inteligentes codificados na linguagem Solidity, oriundos da literatura revisada, que guardam similitude de funções, de modo a realizar uma comparação justa entre eles. E sob a perspectiva de desempenho, focou-se na utilização de métricas que pudessem evidenciar a influência dos níveis de concorrências sobre o desempenho, bem como medidas sobre o consumo de recursos computacionais das máquinas clientes, o consumo de *gas* e, em especial, o impacto de escolhas de algoritmos

versus tempo de consenso de rede, no tempo de resposta das transações na *Rinkeby*.

Finalmente, para concluir, foram apresentadas as propostas de melhoria de desempenho e outros impactos como criptografia não medidos na campanha. Com estas propostas de melhorias, foi mostrado que é possível atingir um número mais escalável de clientes, com custos reduzidos e com boa estabilidade.

4.1 Escolha da Rede Experimental

Partindo-se da premissa de ser necessário um ambiente realístico que pudesse capturar a escalabilidade e o desempenho ao realizar operações de ICP na plataforma da *Blockchain Ethereum*, foi adotada neste trabalho de pesquisa uma das redes de teste da *Ethereum*. Embora, o realismo mais fidedigno possa ser obtido pela implantação de contratos inteligentes diretamente na rede principal da *Ethereum*, os elevados e imprevisíveis custos com esta rede impediram o seu uso neste trabalho.

Para se ter uma noção, o valor do Ether variou de $133,76 a $1.287,48, desde o início desta pesquisa (março de 2020 à janeiro de 2021). Portanto, na execução de testes com centenas de clientes concorrentes realizados neste trabalho, seria necessário um gasto real na rede principal, de mais de 100 Ethers, desde aprendizado até a coleta dos resultados. Um montante de custo médio total de $89.500,00, considerando-se o preço médio do Ether no período de $447,50.

Após verificar a necessidade de trabalhar com uma *testnet* da rede ethereum, foram então considerados certos requisitos mínimos a serem seguidos para a escolha da melhor rede para os experimentos, a saber: estabilidade para medições de desempenho, tamanho de cadeia de dados pequena devido a necessidade de grande número de clientes concorrentes, tempo de bloco pequeno, suporte a cliente programável e automatizável como o Geth.

De acordo com a Tabela 4.1 que elenca os requisitos deste estudo, cabe destacar que a rede *Kovan* não suporta o cliente Geth. Desta forma, em decorrência da escolha do cliente Geth, o mais completo para automatização e seguindo critérios estabelecidos na Seção 2.2.8, tem-se como opção de rede de testes a rede *Ropsten* e a *Rinkeby* e *Görli*.

Outro aspecto importante é que a opção da rede *Ropsten* parece ser muito atrativa para uma campanha de medições, devido a boa reproducibilidade comparada com o ambiente de rede principal *Ethereum*. Porém, na literatura (ZHANG *et al.*, 2019) e em testes preliminares, esta rede apresentou uma grande flutuabilidade de latência das transações e da geração do bloco. Assim sendo, verifica-se que ela também é inadequada para este estudo. De acordo com (HU *et al.*, 2018), a instabilidade da rede *Ropsten* é provocada pelas

TABELA 4.1 – *Tabela Comparativa de Testnets Ethereum*

Rede	**A Favor**	**Contra**
Rinkeby	Imune a ataques de spam e DDoS	Não pode minerar
	Suporte a geth	
	Chaindata de 6 GB	
	Tempo do Bloco: 15 segundos	
Ropsten	Boa reproducibilidade do ambiente da mainnet	Muita instabilidade
	Pode ser usado com qualquer cliente ethereum	Famoso ataque de DDoS em 2017
	É possível a mineração	Ataques frequentes de spam
		Chaindata de 15GB
		Tempo do Bloco: 30 segundos
Kovan	Imune a ataques de spam e DDoS	Não suporta geth
		Não pode minerar
		Chaindata de 13GB
Görli	Suporte a multiplos clientes	Proposta recente
		Poucos usuários

cadeias de PoW, no entanto, as cadeias da Proof of Authority (PoA) são mais robustas ao manter a gerência em nós confiáveis. Igualmente, há relatos de intensa atividade de *spam* e de ataque de negação de serviço distribuída *(DDoS)*, sendo famoso um ataque de *DDoS* em 2017 que poluiu a *Blockchain*.

As últimas redes *Görli* e *Rinkeby* apresentam um ambiente menos ruidoso, também com uma característica crucial do tamanho do bloco de dados da *Blockchain* que precisa ser armazenada na máquina. Como os experimentos são feitos utilizando-se de múltiplos clientes concorrentes, qualquer economia em termos de armazenamento de dados da *Blockchain* economiza espaço em disco para armazenar os resultados.

Entre as duas redes que sobraram, a *Rinkeby* é a mais antiga, livre de *spam e DDoS*, bem gerenciada e possui estabilidade inclusive nos tempos de bloco, com um valor quase-fixo de 15 segundos. Por outro lado, a *Görli* é mais recente e portanto, menos conhecida e com um menor número de usuários e tem um tempo entre blocos mais variável. Portanto, a *Rinkeby* foi escolhida.

Convém destacar que, embora seja uma rede de testes, a *Rinkeby* cumpre prontamente outros requisitos também impostos à rede principal como: operar com a gerência de contas *Ehtereum*; realizar operações de criação, importação e desbloqueio das contas; e criar e gerir os *Ethers* (fundos) para realização das transações, entre outros requisitos, permitindo testar de maneira abundante os passos nativamente utilizados na rede de operação.

Outro aspecto relevante a ser destacado é que diferentemente de outras plataformas de simulação de redes *Ethereum*, citadas no Capítulo 2, a *Rinkeby* é uma rede pública e, portanto, sujeita a um ambiente empiricamente realístico, produzindo um ruído de fundo da própria Internet, pois os clientes estão distribuídos geograficamente na Internet.

4.1.1 Aspectos Gerenciais da Rede Rinkeby

Em geral, conforme especificações da rede de teste *Rinkeby*, é efetuada a adição de blocos de dados à *Blockchain* a cada 15 segundos. O algoritmo de consenso é diferente da rede principal, mas isto facilita experimentos de desempenho, pois o PoA atinge de maneira mais rápida e consistente o estado das transações.

Nesse algoritmo de consenso, existe a exigência de nós confiáveis. Portanto, é necessária a prova de existência antes de obter Ether (de teste) para a realização das transações na rede. Para este fim, os desenvolvedores criaram o conceito de "Faucets" (torneiras de dinheiro) que consistem em *sites* que transferem para experimentos, pequenas quantias de Ether, para os usuários em troca de publicação em mídia social e sua respectiva comprovação (SOLOMON, 2019).

Para exemplificar este cenário, foi utilizada a requisição de Ether a partir de um, de cerca de mil endereços de contas que foram criadas e manipuladas no ambiente de experimentação desse estudo. A Figura 4.1 mostra uma das maneiras de se obter *faucet Ether via Twitter*. Para tanto, um *tweet* é criado solicitando fundos no endereço da conta desejada. A conta de *Twitter* é de usuário legitimo e o *tweet* precisa ser criado dentro de um período recente de tempo.

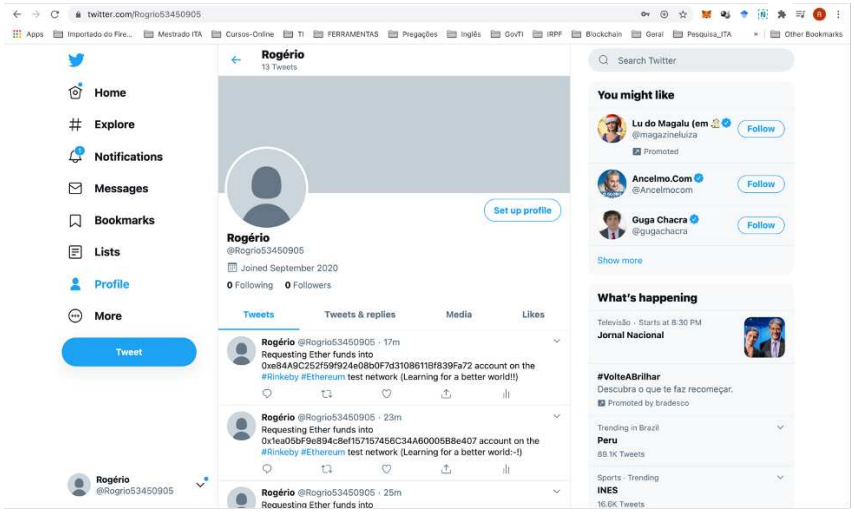


FIGURA 4.1 – Publicação em mídia social para requisição de fundos

.

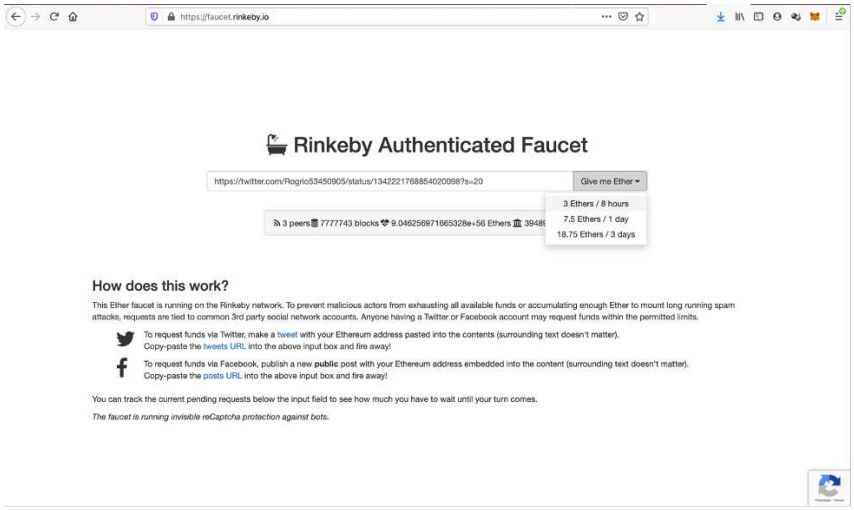


FIGURA 4.2 – Prova de existência e seleção da quantia de fundos pretendida

Posteriormente, é preciso acessar o *site* oficial de "Faucets" da Rinkeby para apontar o endereço de URL do seu *tweet* atestando a prova de existência da conta para a requisição de fundos, conforme mostrado na Figura 4.2. Com estas etapas cumpridas, o usuário pode então selecionar a quantidade de Ether a ser requisitada, até um certo limite, de acordo com o tempo que o usuário concorda fazer uma próxima requisição. Por exemplo, 3 Ethers em 8 horas é uma das possibilidades listadas. Isto evita ataques aos recursos desta rede. Observe que esta página também é protegida por *captcha* e, portanto, um processo que só pode ser feito manualmente.

Uma vez feita a requisição de financiamento listada no *Twitter*, cujo endereço criado é $e84A9C2$ (Figura 4.1), é possível acompanhar o processamento da transferência dos recursos no *website* da *Rinkeby* (Figura 4.3 b). Para demonstrar a interação entre os processos de uma transação, seu mapeamento no *Dashboard* da *Rinkeby* e no explorador de Blocos da *Rinkeby*, executou-se a transação unityTest.js, resultando nos dados ilustrados na Figura 4.3 c. Observe que o bloco de transação tem o número 7788549.

Quanto ao mapeamento no *Dashboard* da *Rinkeby*, é possível verificar a criação do bloco 7788549 (Figura 4.3 b) instantes após o acionamento da transação (Figura 4.3 c). Também é possível realizar várias buscas de interesse destas transações excutadas com base na informação do número do bloco, com o *hash* da transação e com o endereço do remente (Figura 4.3 c), bastando inserí-los no explorador de blocos (Figura 4.3 b).

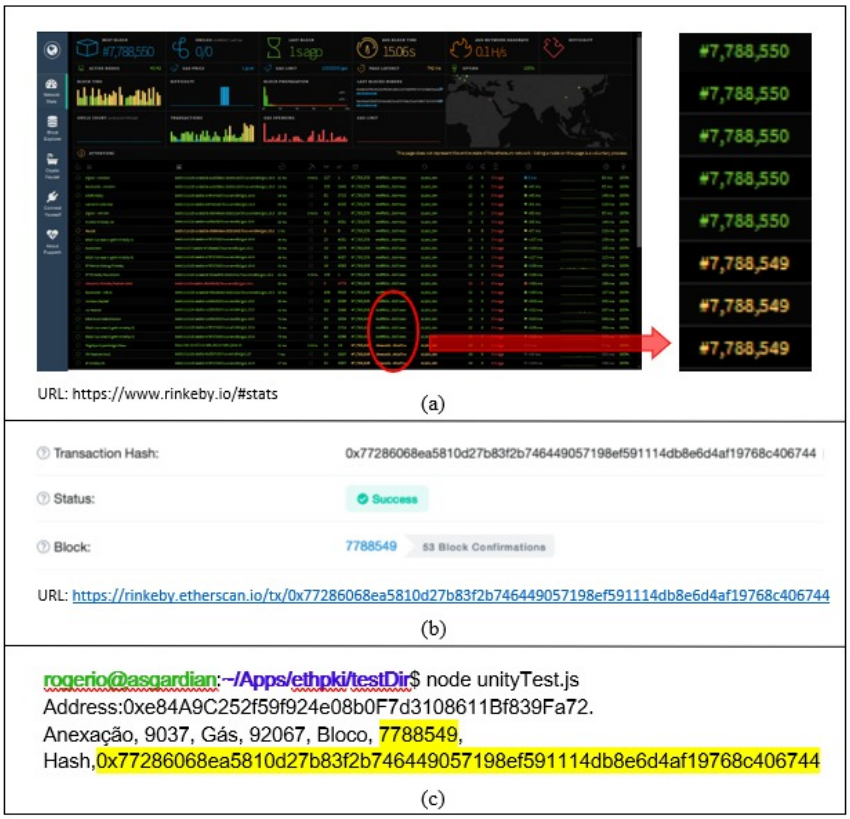


FIGURA 4.3 – *Dashboard* da rede *Rinkeby* - Transação 1

Em seguida, após um conjunto de validações (confirmações) dos nós ativos da rede, outras transações são realizadas através dos números $0x410ee$ e $0x9544a6$, que poderão ser acompanhadas com a mesma dinâmica estabelecida no parágrafo anterior. Desta forma, é possível rastrear a transação do nó financiador enviando recursos solicitados para a conta de interesse.

Também é possível observar nas Figuras 4.3 e 4.4 que o *gas* é um parâmetro que possui um comportamento parecido com o da rede principal. Por exemplo, as transações $0x410ee$ e $0x9544a6$ possuem *gas* 22577 e 22206 respectivamente. O *gas* na rede Rinkeby possui um aspecto de controle de abuso, uma vez que pode prevenir a violação do uso da rede (TELES, 2018). Porém, convém destacar que o mesmo parâmetro de taxas (de forma não exclusiva) também é considerado e aplicado na rede principal *Ethereum* (WOOD *et al.*, 2014) e (CHEN *et al.*, 2020).

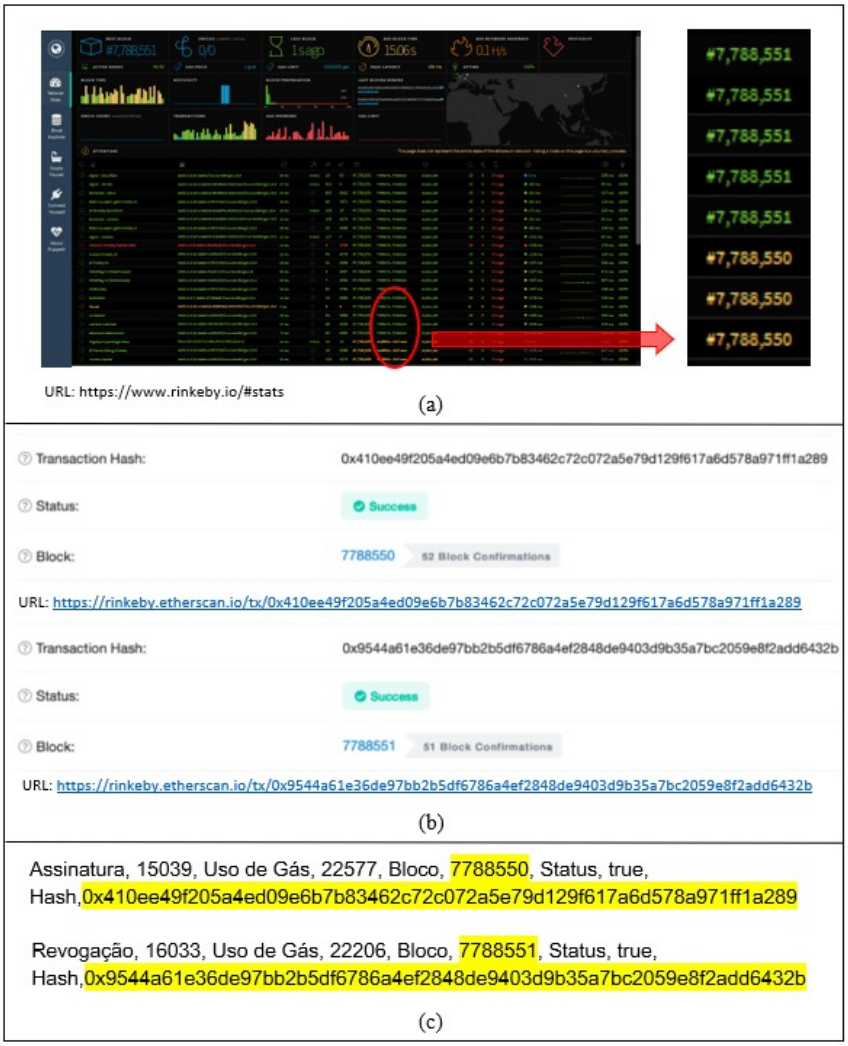


FIGURA 4.4 – *Dashboard* da rede *Rinkeby* - Transação 2

Quanto aos casos referentes às redes de testes da Ethereum, cabe observar que o Ether não tem valor nenhum, mas como ele é transferido de perfis conhecidos no *Twitter*, o *gas* é meramente utilizado como uma medida para prevenir abuso na rede. Em outras palavras, muitas transações podem acabar zerando os recursos providos em caso de abuso. Nos testes realizados nesta pesquisa, embora tenha sido feito um grande número de transações concorrentes, entretanto, não se chegou a zerar recursos de teste.

4.2 Ambiente de Experimentação e Metodologia dos Testes

Conduzir experimentos repetitivos em um ambiente "não-simulado" de uma rede *testnet Ethereum* na própria Internet, e com usuários reais ao fundo, conduzindo outros tipos de experimentos e transações, pode ser desafiador. Portanto, para os objetivos deste trabalho, foi desenvolvido um ferramental para orquestrar e instanciar os experimentos com uma carga de repetição e com a monitoração de todas as variáveis pertinentes como

CPU, disco e tempos de resposta.

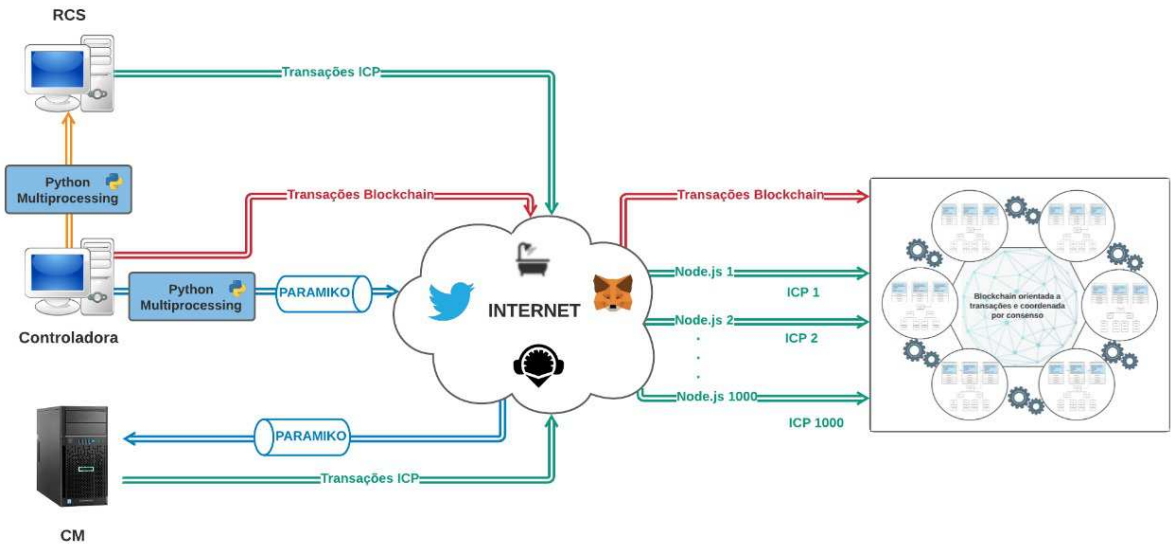


FIGURA 4.5 – Arquitetura do Projeto de Medições

A arquitetura apresentada na Figura 4.5 ilustra a visão geral da arquitetura do Projeto de Medições. O fluxo de trabalho descrito na referida figura tem posicionados seus principais componentes e mostra as diversas fases até a coleta dos resultados. Entre os componentes contidos na Figura 4.5, destaca-se a máquina controladora que tem como objetivo orquestrar o experimento, o número de clientes concorrentes, número de repetições e verificar se existem recursos de Ether e se esses tenham sido previamente alocados, disparar remotamente os clientes Geth com os parâmetros necessários e coletar os resultados dos experimentos.

Por outro lado, as máquinas RCS e CM são máquinas relativamente potentes que realizam as transações propriamente ditas na *Blockchain* e coletam os resultados. Devido ao *setup* físico não era possível realizar experimentos com número maior que 400 clientes concorrentes devido a limitações de memória e uso de CPU. Outro ponto importante é que enquanto a máquina controladora e RCS ficam localizadas em uma mesma sub-rede local, por outro lado, a máquina CM ficava localizada remotamente. Portanto, foi utilizado a biblioteca *Paramiko* do *Python* que permitiu o controle de sessão remoto e condução dos aplicativos com segurança SSH.

Nos fluxos em vermelho da Figura 4.5, destacam-se operações diretas na *Blockchain* como criação de contas, e instalação de *smartcontracts*, os fluxos em verde representam chamadas remotas de procedimento via o aplicativo Geth, os fluxos em azul representam chamadas remotas de procedimento via biblioteca *Paramiko* do *Python* e, o fluxo em laranja representa chamada entre as máquinas Controladora e RCS, que ficam localizadas em uma mesma sub-rede. Para este desenvolvimento foi criado um arquivo na linguagem

Node.js.

Em resumo, esta arquitetura possui 5 fases principais no Projeto de Medições. Tais fases acontecem consecutivamente e a paralelização no controle acontece por meio do uso do multiprocessamento no *Python*. Maiores detalhes desta implementação encontram-se disponíveis no repositório público do github[1]. Uma descrição sucinta das fases se segue e maiores detalhes sobre o ferramental são detalhados na próxima Seção:

- Processo de criação de contas - o controlador dispara transações diretamente na *Blockchain* para criar contas e vincular as mesmas na ferramenta *metamask*;
- Processo de requisição de Ether - o controlador dispara a transação *Blockchain* para adquirir Ether junto ao *twitter*. Existe uma etapa manual nessa fase, por conta de *re-captcha* de comprovação da requisição de recursos junto à *"rinkeby authenticated faucet"* e posterimente o controlador verifica se os fundos (Ether) foram liberados;
- Processo de implantação (*deploy*) do *smartcontract*. O controlador dispara a transação *Blockchain* para envio do contrato para a rede. Para tanto, ele orquestra a ferramenta Remix via estabelecimento de conexão com a biblioteca Web3 e por outro lado conecta com a ferramenta Metamask, para passar a conta Ethereum com fundos, para que ela realize a implantação do mesmo;
- Processo de importação e desbloqueio de contas - o controlador executa comandos de maneira distribuída (via *Paramiko*) para que cada máquina CM e RCS possa atrelar contas criadas aos processos concorrentes do cliente Geth e também desbloquear o uso dessas contas, cujos recursos foram transferidos em passo anterior; e
- Processo da execução concorrente de transações na rede de testes Rinkeby - Trata-se da execução final do processo de medição. Executa certas funções das implementações de ICP baseados em *Blockchain* da literatura e captura as métricas de interesse.

Com relação a metodologia, o controlador aciona as execuções concorrentes de transações e são coletados conjuntos de 30 experimentos com a mesma configuração. Este valor foi adotado para se ter maior grau de confiança estatística nos dados. Foram variados os tamanho dos *clusters* de clientes concorrentes, também foram variados os algoritmos das ICP e, finalmente, os testes foram divididos pelas funções específicas para se ter um retrato completo do desempenho e escalabilidade das execuções das ICPs baseadas em *Blockchain*.

Para fins de reproducibilidade, apresenta-se na Tabela 4.2 as especificações técnicas das máquinas utilizadas nessa arquitetura de teste, bem como as versões de software utilizadas.

[1]https://github.com/rogerioita/ICPChain

TABELA 4.2 – *Setup* das Máquinas participantes

	CM	**RCS**
Processador	4.2 Ghz Quad-Core Intel Core i7	2.2 Ghz Quad-Core Intel Core i7
RAM	64 GB	16 GB
Disco	SSD 1,8 TB	SSD 250 GB
Sis. Oper.	Ubuntu 5.4.0-62-generic	MAC Big Sur 11.1
Python	2.7.17	Python 2.7.16
Geth	1.9.24-stable	1.9.23-stable
Node.js	v10.23.0	v10.21.0

4.2.1 Ferramental Utilizado

Para se obter um melhor detalhamento, no contexto da arquitetura de testes, esta Seção apresenta as diversas ferramentas, bibliotecas e tecnologias no ecossistema Rinkeby que deram suporte ao ambiente de experimentação. Algumas das ferramentas foram parametrizadas para os experimentos e outras tiveram que ser desenvolvidas. Isto será destacado nas próximas Sub-seções, como um resumo, o uso de cada uma delas.

4.2.1.1 Desenvolvimento e Implantação de *smartcontracts*

O ambiente Remix é basicamente um *plugin* de ambiente de desenvolvimento integrado (IDE) que facilita a codificação de contratos inteligentes diretamente do browser. Os contratos são desenvolvidos diretamente em linguagem Solidity. Entre os recursos desse *plugin* ele permite a compilação do código em *bytecode* da máquina virtual *Ethereum.* Também cria automaticamente o ABI (Application Binary Interface) que é uma espécie de API para aplicações javascript ou python se conectarem aos contratos inteligentes.

Em termos de opções, o Remix possui uma ampla variedade de configurações. Por exemplo, ele permite selecionar a versão do compilador Solidity a ser usado, como uma forma de dar a possibilidade de compilar código legado de versões anteriores de Solidity. Além disso, o *plugin* permite realizar a implantação em tempo real dos contratos inteligentes não somente na *Rikeby*, mas também *Ropsten* e outros (WU *et al.*, 2019). Finalmente, sua facilidade permite realizar experimentos de invocação de funções dos contratos inteligentes sem sair do navegador, apenas atrelando uma conta *Ethereum* com recursos (obtidos via "Faucets").

No contexto deste trabalho de pesquisa, utilizou-se o *Remix* para a fase de desenvolvimento e adequação dos contratos inteligentes escolhidos da literatura para avaliação de desempenho. Por exemplo, foram reestruturadas as funções dos *smartcontracts* para se tornarem similares e facilmente comparáveis e eventualmente, também foi aumentado a complexidade por meio de funções criptográficas. Adicionalmente ao trabalho de desenvolvimento, o *Remix* serviu também para a submissão e implantação desses contratos na

rede *Rinkeby*.

Outro uso relevante foi no cálculo para determinar a quantidade de *gas* necessária para o *deploy* dos três contratos inteligentes, conforme pode ser ilustrado na Figura 4.6. Nesta figura, pode-se notar duas referências de custos relacionados ao *gas* (*transaction cost* e *execution cost*). Como a plataforma da Remix realiza a implantação do *smart contracts* em VM Javascript (no próprio ambiente de sandbox), dessa forma, ela consegue distinguir os custos de execução, referentes ao processamento empregado na máquina virtual, dos custos de transação que englobam conjuntamente custo de execução e custo de envio de dados e armazenagem na *Blockchain*.

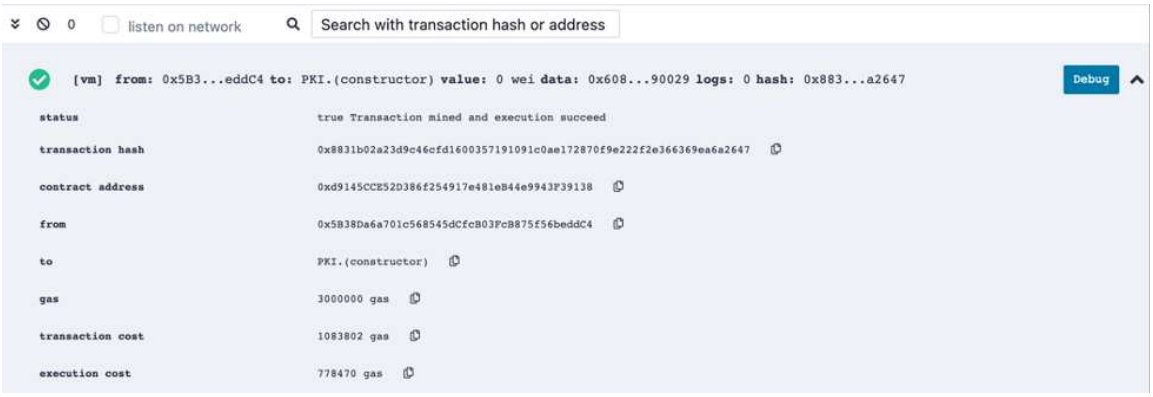


FIGURA 4.6 – Custo de *gas* na sandbox do *Remix*

4.2.1.2 Criação de Contas Ethereum, Interface Web e Testes de Execução de Funções dos *smartcontracts*

Os scripts de criação de endereços Ethereum e de testes dos contratos inteligentes empregados neste ambiente de experimentação foram criados em Javascript. A Figura 4.7 ilustra o script de criação das contas. O restante dos códigos foram criados usando Node.js, que é uma plataforma de execução de eventos em Javascript, normalmente usado no *back-end* de sitios dinâmicos. Em especial, utilizou-se um conjunto de pacotes gerenciados pelo Node Package Manager (NPM) para dar todas as funcionalidades aos nossos *scripts* (INFANTE, 2019). Em especial, destaca-se o Web3.js e shelljs que permitem realizar as requisições para os *smart contracts* por meio de chamada remota de procedimento (RPC) através do cliente Geth e, portanto, habilitando a interação com os contratos inteligentes na Rinkeby.

```
let Web3 = require('web3');
var shell = require('shelljs');
let web3 = new Web3();
web3.setProvider(new web3.providers.HttpProvider('http://<IP>:<Porta>'));

var k =0;


while (k <= 999) {
    // Cria 1000 novas contas.
    shell.exec(`geth --rinkeby account new --password pass`)
    k++
}
```

FIGURA 4.7 – Script de criação de contas

4.2.1.3 Cliente Ethereum Principal

Para que a aplicação dos experimentos ficasse completa, foi necessário um cliente Ethereum principal. Conforme citado na Seção 2.2.7, optou-se por um cliente automatizável e completo para o ambiente de experimentação. Neste caso, o Geth é uma das implementações originais e com maior quantidade de contribuições e usos do protocolo Ethereum, o G do Geth significa que ele é desenvolvido na linguagem Go promovida pela Google, conhecida pelo seu forte gerenciamento de memória, processamento em paralelo e velocidade próxima a de C no binário.

O cliente Geth é a implementação usada por 82% dos clientes Ethereum, sejam mineradores ou aplicativos. Ele possibilita todas as interações necessárias com a rede *Rinkeby*, tais como a criação e desbloqueio de contas, transferência de Ethers, realização da comunicação com os *smart contracts*, dentre outras.

No ambiente de experimentação, a conexão com a rede *Rinkeby* foi realizada utilizando o modo de sincronização *"full"*, ou seja, armazenado e processando todas as transações da *Blockchain*. Inicialmente, foram posicionados os aplicativos Geth em máquinas virtuais, dentro da máquina principal de experimentação, mas cada VM consumia uma quantidade considerável de disco virtual devido a este critério de armazenamento. Por outro lado, isto facilitava a busca de transações e dos endereços e a validação que está toda em *cache*. Eventualmente, optou-se por rodar instâncias do programa geth na mesma máquina, reaproveitando os dados da *Blockchain* completa. Adicionalmente, o Geth permite requisições por RPC, conforme mostrado na parametrização apresentada na Figura 4.8.

```
geth --rinkeby --rpc --rpcaddr <IP> \
--datadir=/<endereço_keystore> \
--port <porta> --rpcport <porta_rpc> \
--rpcapi="eth,net,web3,personal,txpool" \
--allow-insecure-unlock console
```

FIGURA 4.8 – Conexão com a Rinkeby.

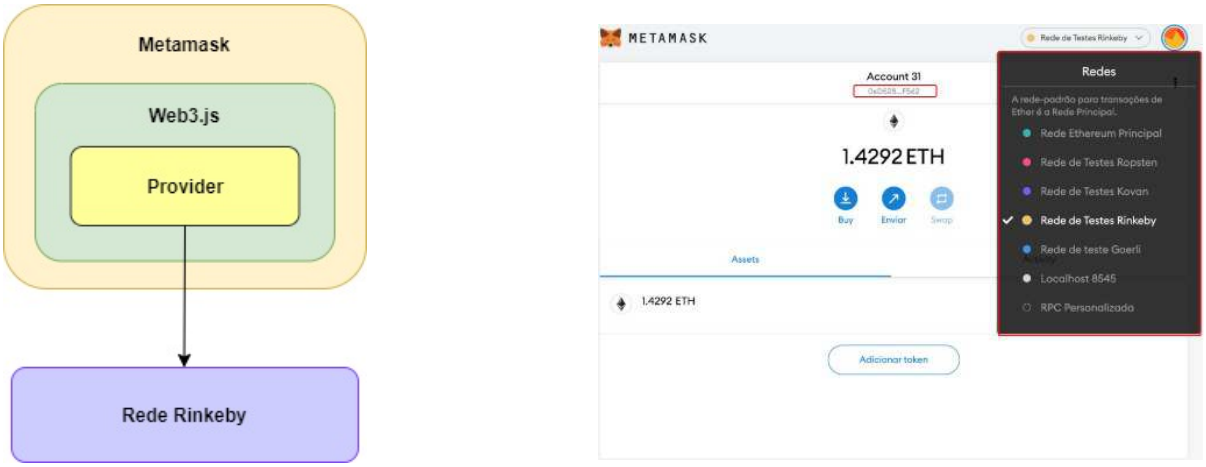
Outros parâmetros pertinentes usados e apresentados na Figura 4.8 são relativos a indicação de APIs utilizadas em conjunto com esse cliente *Ethereum*. No caso foi especificado as APIs "eth, net, web3, personal e txpool". Algumas destas APIs permitem a utilização de um mesmo endereço IP público para gerenciar várias contas *Ethereum*, o que se mostrou conveniente nos experimentos. E também se ressalta que o IP público exposto precisa incluir o parâmetro "–allow-insecure-unlock", como artifício para desbloqueio das contas, pois por padrão estes acessos de contas estariam bloqueados.

4.2.1.4 Gerência das Carteiras e dos Recursos de Criptomoedas Ether

No que tange aos aspectos de usabilidade das aplicações descentralizadas (DApps), é preciso realizar corretamente a gerência das carteiras de contas. Também é preciso realizar transferências entre estas contas e gerenciar os recursos de criptomoeda *Ethereum*. Deste modo, neste trabalho de pesquisa utilizou-se do *plug-in de navegador Metamask*. Trata-se de um software concebido para servir como: carteira *Ethereum* e propiciar interação com DApps. Por exemplo, com relação a este último, permite a implantação de *smart contracts* sem a necessidade de cliente *Ethereum* completo. Internamente, este possui os elementos para interagir com a rede. A Figura 4.9a ilustra os componentes que realizam a conexão anteriormente citada.

A estrutura de API Web3.js do *plugin Metamask* é descarregada no navegador e, a partir dela, é possível estabelecer conexões HTTP, RPC ou websockets [2] com a *Blockchain*. Em particular, o protocolo *Ethereum* trabalha dentro do núcleo do que é chamado *provider* equivalente a um pequeno cliente. Os *providers* tem conexão com diversas redes de testes da *Etherem* como Rinkeby, Kovan, e até mesmo a rede principal, como está ilustrado no rol de opções do *Metamask* na Figura 4.9b.

[2] https://www.html5rocks.com/pt/tutorials/websockets/basics/



(a) Componentes do Metamask (b) Lista de *Providers* do Metamask

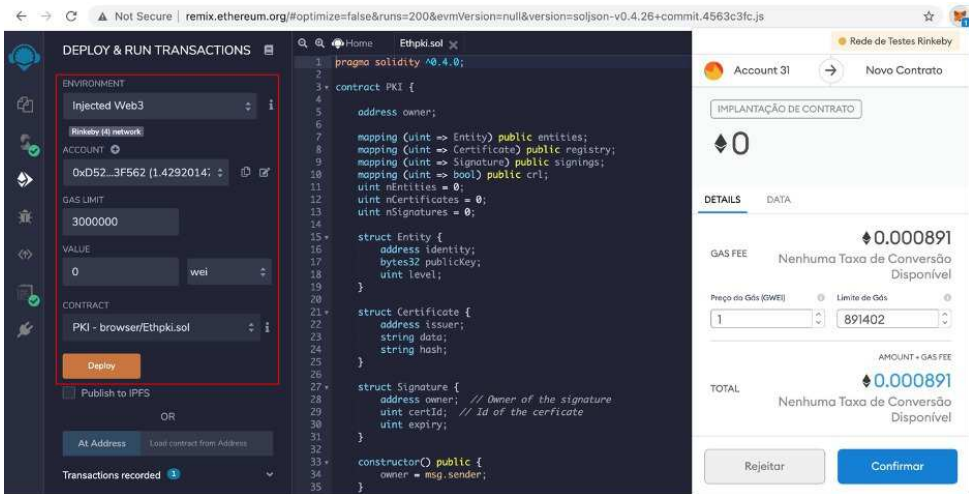
FIGURA 4.9 – Componentes e provedores do Metamask

No ambiente de experimentação, ao ser realizada a implantação dos contratos inteligentes, o *software remix* é responsável pela comunicação com o *metamask* de modo a ser utilizado como meio de pagamento em *gas* pelo esforço computacional dos mineradores, como pode ser visualizado na Figura 4.10a. Após a confirmação da transação de implantação de contrato inteligente na Rinkeby, o *Remix* faz a recepção da confirmação da implantação do *smart contract*, constando mimamente da URL de validação e rastreio para Etherscan. A partir da mesma, é possível obter todas as outras informações da transação, a quantidade de *gas* usado e o hash da transação e, sua origem e destino. A Figura 4.10b ilustra este processo.

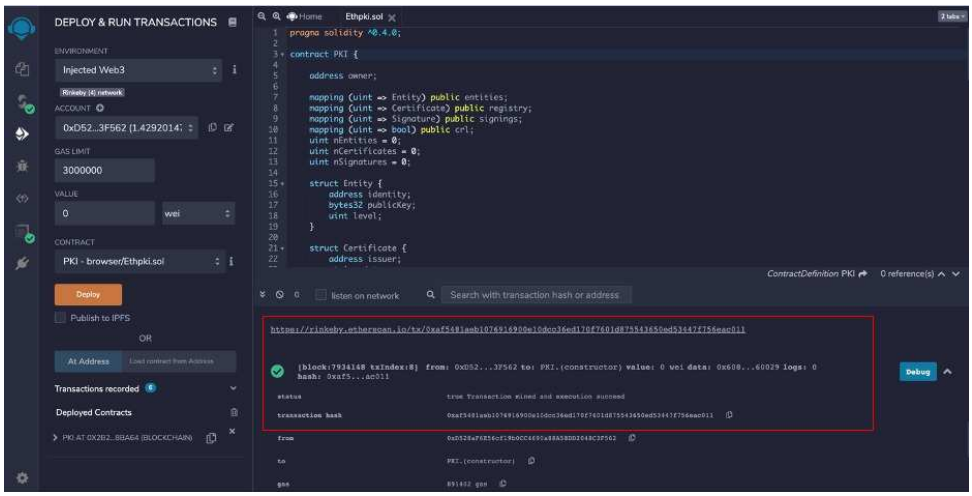
4.3 Caracterização da Rede Rinkeby

Antes de realizar os experimentos, faz-se necessário se certificar que o ambiente da *Rinkeby* é capaz de suportar a realização de transações de ICP em uma *Blockchain* de forma realística com clientes reais. Além disto, contemplando a interação em uma cadeia de blocos pública com pouco ruído. Portanto, foi realizada uma caraterização para entender fatores fora do controle acerca dos experimentos realizados neste ambiente de execução.

Ressalta-se que o objetivo desta abordagem pragmática de realização de experimentos é permitir observar várias nuances reais do ecossistema da *Blockchain* que necessitam ser mapeadas para a identificação de possíveis gargalos ou pontos de melhorias no desenvolvimento das DApps. Entre as características de interesse, pode-se destacar: o tamanho da rede em termos de nós; os tempos de atrasos dos clientes e potencial impacto dessa distribuição geográfica nos tempos de processamento dos *smart contracts*; a rotina de conexão com pares na rede ao longo do tempo; e, finalmente, o volume de transações bem



(a) Componentes do *Metamask*



(b) Lista de *providers* do *Mestamask*

FIGURA 4.10 – Combinação de Remix e Metamask para financiar as contas de teste

como sua variabilidade.

Os experimentos realizados nesta análise exploratória de dados foram realizados com um único cliente obtendo informações da rede. Também foram obtidos dados diretamente da plataforma web da *Rinkeby*, via processo de *scraping* das páginas. Todos os dados dos experimentos foram executados 30 vezes para obter médias e desvio e traçar um perfil estatístico robusto para cada medida.

4.3.1 Número de Nós Vizinhos

Uma das questões pertinentes em uma campanha de medição é a de se tentar estimar a quantidade de nós concorrendo com o experimento. Por exemplo, na rede principal do *Ethereum* estima-se que existam mais de 100 mil mineradores ativos. Isso torna a rede vasta do ponto de vista de recursos. Mas exatamente, por causa desse volume enorme

de clientes, a variabilidade para medir tempos de processamento de *smartcontracts* pode deixar a desejar.

Outro aspecto relevante é a chamada taxa de rotatividade (em inglês, *churn rate*. Pois uma razão grande de clientes pode simplesmente desconectar e oportunamente religar, como toda a rede Par-a-Par. Portanto, é desejável também que a estabilidade com vizinhos aconteça em curto período de tempo desde o estabelecimento da comunicação.

Sob essa perspectiva, foi realizada a coleta de dados de um período de 1 hora, por meio de *scrapping* de atualizações de clientes na plataforma de estatística *web* do Rinkeby [3]. Deste modo, a ideia era capturar quantos vizinhos cada cliente dizia ter conhecido, em outras palavras, se o nó apresenta o número de *peers* de referência a ele conectado ao longo de sua história. E assim estabelecer uma noção de como esta rede é formada. Ou seja, contando com *peers* mais recentes, com poucos vizinhos e outros *super-peers* estáveis que estiveram conectados na rede Rinkeby há muito tempo. Os valores capturados no *site* de estatística são os *Peers* e representam justamente essa relação.

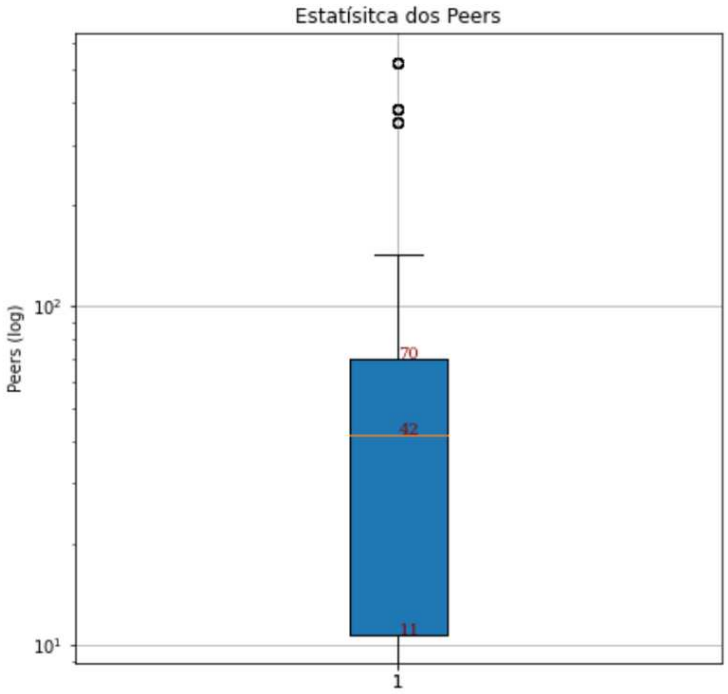


FIGURA 4.11 – Estatística dos *peers*

Os dados coletados são mostrados na Figura 4.11. Note-se que a figura boxplot está em escala logarítmica pois, em boa parte da distribuição situa-se dentro de uma faixa inter-quartil (Q1 e Q3) entre 11 e 70 (área retangular do gráfico) com mediana em 42. Entretanto, existe um número de pontos fora da curva (*outliers*). Estes super-nós são facilmente reconhecidos por seus nomes e pertencem à Infura e Akasha, fornecedores de

[3] https://stats.rinkeby.io

infraestrutura da Ethereum, como podemos verificar nos nomes *bootnode e signer infura e akasha*. E seu padrão de conexão é maior que 200 *peers*.

Como a maioria dos nós entra e sai, especula-se que estes nós de infraestrutura da *Rinkeby* estão atuando por meses ou até anos. Portanto, têm um registro mais longo de nós, dentro de um período, com os quais teve contato. Desta forma, combinando estas informações de *outliers* da Figura 4.11, pode-se supor que o ambiente Rinkeby em termos de conectividade segue uma distribuição exponencial, por conta destes super-nós, algo parecido com o que acontece na rede principal da Ethereum (KIM *et al.*, 2018).

4.3.2 Distribuição Geográfica dos Nós

Outro ponto de interesse é a distribuição geográfica dos nós. Com essa perspectiva, busca-se obter como resultado o panorama da disposição dos nós e seus diferentes atrasos na Internet. O produto desta análise auxilia a desvendar a existência de problemas de congestionamento na rede. Também revela se existe alguma possível concentração de nós em determinadas regiões, mitigando assim o risco de invalidação de algum resultado que porventura pudesse impactar nos testes de desempenho. Como os tempos de validação de blocos de transações na *Blockchain* são da ordem de 15 segundos, segundo especificações da *Rinkeby*, esses atrasos não parecem ser significantes para impactar as medições.

De maneira semelhante aos resultados do número de nós, foi realizada outra extração de dados, por meio de *web scrapping* do site de estatísticas. Desta vez, o dado capturado foi o tempo médio de propagação (*average propagation time*) em um período de poucos minutos. Desta forma, do ponto de vista do nó central coletor das estatísticas, esses dados registram os tempos de propagação do nós ao longo do tempo.

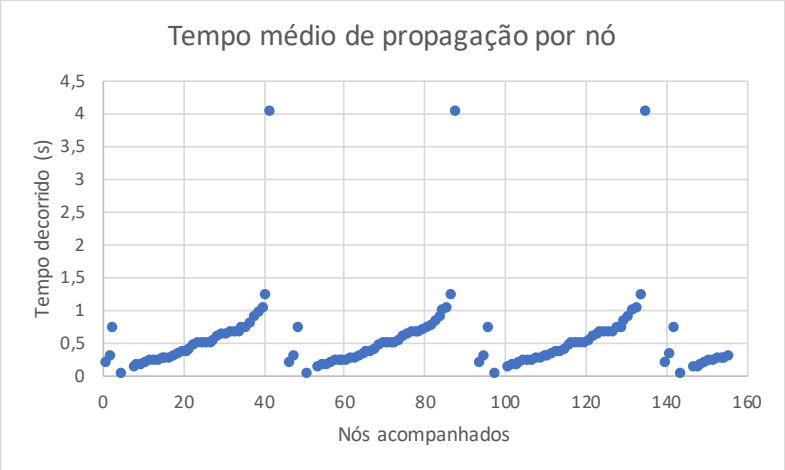


FIGURA 4.12 – *Polling* de tempo de propagação por nós

A Figura 4.12 representa a distribuição dos tempos de propagação de determinados *peers* indexados pelos eventos gerados dos nós acompanhados. Verifica-se então uma certa

ordenação desses eventos. Portanto, em cada ciclo uma espécie de *polling* parece estar em efeito. De modo que o nó mais próximo ao *site* de estatística responde primeiro e depois nós mais distantes sucessivamente. Portanto, os tempos de propagação apresentam esse comportamento de poucos milisegundos até 1 segundo, na maioria dos nós. Isto explica os sucessivos movimentos de subida do gráfico. Em outras palavras, os eventos aparecem em função da propagação do nó para o *site*.

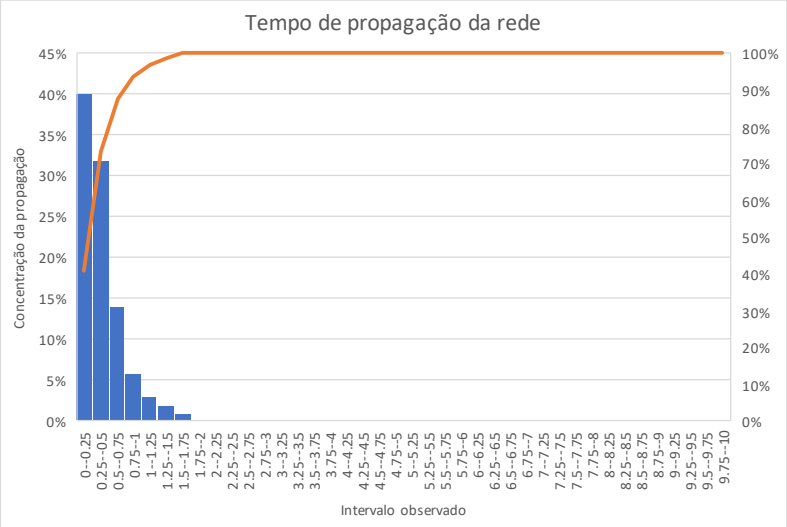


FIGURA 4.13 – Distribuição de tempos de propagação

Note-se também que, na Figura 4.13, o histograma da distribuição desses atrasos de propagação apresenta concentração dos tempos nas categorias 0-0.25 segundos (até 250 ms), 0.25-0.5 segundos (até 500 ms) e 0.75-1 segundos (inferior a 1 segundo). A reta em laranja é a probabilidade cumulativa. Portanto quase todos os nós estão com atraso inferior a 1,5 segundo.

Esta distribuição é normal, pois atrasos de propagação de rede na Terra giram em torno de 250 a 750 ms dependendo de fatores como enlaces transoceânicos e também atrasos de processamento em máquinas lentas, não dedicadas, que executam vários processos podendo ter uma numerosa carga de processamento. Um aspecto peculiar que merece destaque é a existência de um ponto que leva 4 segundos de processamento, sendo que este evento pode ser considerado como um "outlier" ou o gerente de evento do *polling* que dispara um *timeout* do sistema. Isto pode ser justificado pela Fig. 4.12 apresentar uma repetição na coleta das informações após este ponto.

O fato dos tempos de propagação estarem dispostos como uma reta na Figura 4.12, parece induzir a conclusão de que os tempos de propagação estão uniformemente distribuídos geograficamente, ou seja, sem concentrações aparentes. No *site* de estatísticas também é possível verificar um mapa com os nós conectados e percebe-se que embora a maioria esteja distribuído nos EUA e Europa, tem-se em um número maior na América do Sul, Ásia e África. E enquanto nos EUA, tem-se uma distribuição equilibrada entre

costa leste, oeste e centro.

4.3.3 Número de nós ao longo do tempo e estabilidade das transações

A coleta de dados dos *peers* apresentada anteriormente não apresenta de maneira consistente como é a evolução da taxa de rotatividade (em inglês, *churn rate* ao longo do tempo. Deste modo, foram realizados testes de longa duração contemplando 24 horas e 48 horas utilizando o cliente Geth conectado a rede Rinkeby e conduzindo verificações sobre os *peers* ativos e transações ao longo desse tempo.

O objetivo deste experimento era entender essa relação de fatores de estabilidade da rede experimental Rinkeby, para que os experimentos de desempenho posteriores não apresentassem resultados com grande variação ao longo do tempo.

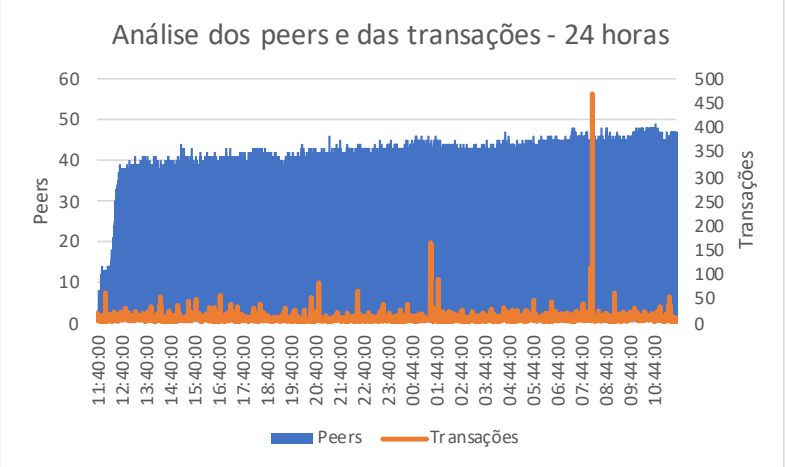
Os resultados desses experimentos de caracterização encontram-se apresentados nas Figuras 4.14a e 4.14b para os períodos de 24 e 48 horas respectivamente. Na Figura 4.14b, o eixo X apresenta os períodos em questão do ponto de vista do relógio da máquina CM, e é mostrado dois eixos Y. O primeiro eixo Y1 é a apresentação de *peers* ao longo do tempo, e a segundo eixo Y2 é a quantidade de transações por período de tempo. Os resultados foram coletados a cada 15 segundos ao longo das horas, mas no gráfico é exibido a média por hora para facilitar a visualização.

Note-se que as Figuras 4.14a e 4.14b mostram aspectos que validam a observação anterior, de que os *peers* descobertos ao longo do tempo, vão entrando na lista de *peers ativos*. Assim, a rede parece que está sempre em crescimento. Realizando-se uma análise cruzada destes dados com a Figura 4.11, onde 68 peers em média era o número de vizinhos (note-se que 42 é a mediana), percebe-se que ainda assim, tanto o período de um ou dois dias não é o suficiente para se conhecer a "maioria" dos nós ativos da rede.

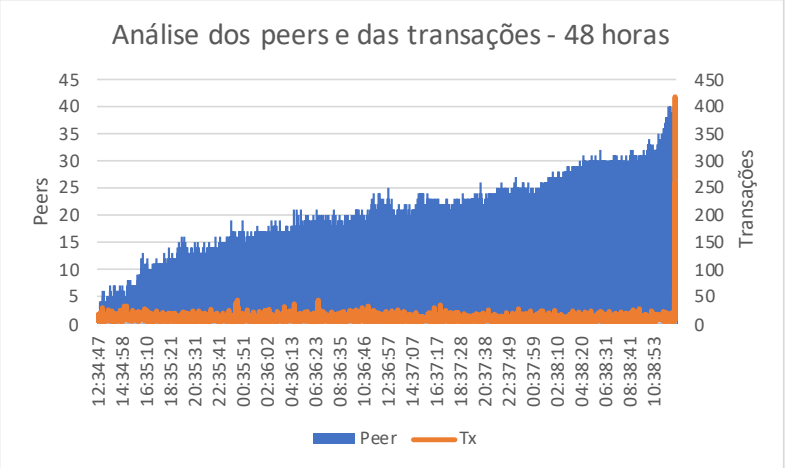
No caso da Figura 4.14a, em questão de poucas horas, provavelmente em um dia ocupado, foi possível atingir um patamar de 40 *peers* conhecidos e este número foi subindo lentamente, mas sempre flutuando um ou outro nó, com nós entrando e saindo. Do ponto de vista (PoV) de transações na Figura 4.14a, percebe-se que o volume de transações é bem estável ao longo das 24 horas, tendo somente um pico de 450 transações as 7:50AM, como mostra as barras em laranja.

Para aprofundar o aspecto de estabilidade das transações, pode-se verificar utilizando-se de *boxplot* sobre os dados das transações, esses gráficos foram podados em 100 transações. Conforme Figura 4.15a, o dia mais ocupado (24 horas) estão distribuídas entre 5 e 12 transações por amostra e, de maneira semelhante, o período maior de 48 horas (Figura 4.15b) também apresentou estas medidas nos quartis 1 e 3.

Embora, a configuração deste detalhamento no Tableau, utilizando-se os dados das



(a) Análise dos *peers* e das transações - 24 horas



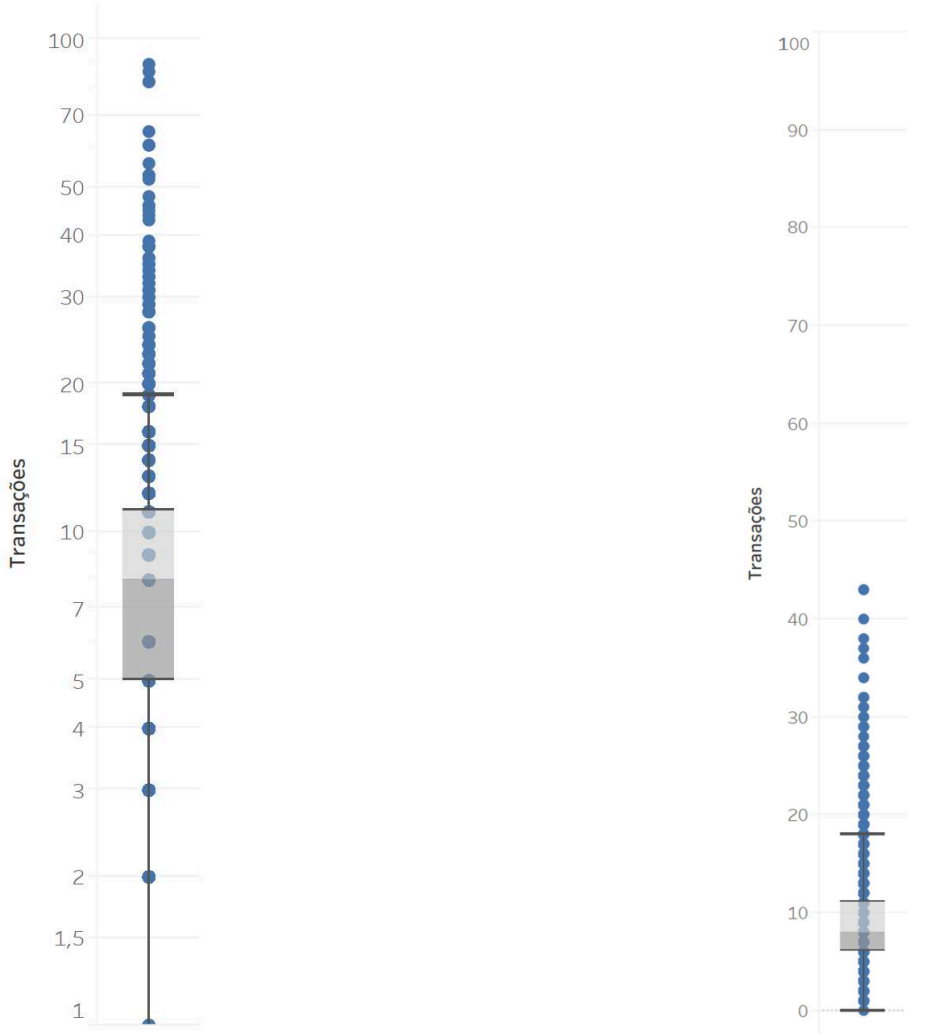
(b) Análise dos *peers* e das transações - 48 horas

FIGURA 4.14 – Visualização de dados na perspectiva do Número de nós vizinhos

Figuras 4.14a e 4.14b, tenha puxado os *boxes* para posições diferentes, eles se encontram no mesmo patamar de valor. A diferença entre 24 horas e 48 horas são os *outliers* que são mais presentes no teste de 24 horas, com vários superiores a 12 até 100. Estes resultam na estabilidade no volume de transações.

4.3.4 Lições aprendidas sobre a rede Rinkeby

A etapa de caracterização da rede Rinkeby mostrou que a mesma é apropriada para experimentos de desempenho, no sentido que o número de usuários é estável, com pelo menos 68 *peers* em média para comunicação. As transações também em períodos curtos de 15 segundos apresentam estar situadas em um patamar de 5 a 12 para cada bloco escrito na *Blockchain*. O volume de transações acontece também de maneira quase constante, permitindo boa previsibilidade. E, finalmente, a dispersão geográfica dos nós parece estar uniformemente distribuída, do ponto de vista de atrasos de propagação, como mostra a



(a) Transações PoV nós - 24h.

(b) Transações PoV nós - 48h.

FIGURA 4.15 – Visualização de dados na perspectiva da estabilidade das transações

reta linear de tempos de dispersão apresentada. Em outras palavras, com baixa chance de congestionamentos e outros artifícios de flutuação de tempo. Todos estes atributos, justificam e validam a escolha feita pela rede *Rinkeby*, como sendo a mais apropriada.

4.4 Síntese do Capítulo 4

Este Capítulo descreveu como foi realizada a campanha de medições que visam investigar o desempenho de ICPs baseadas em *Blockchain* e o ambiente experimental utilizado para tal, que envolve a criação de contas, a requisição de Ether, a implantação de contratos inteligentes, a importação e desbloqueio de contas e a execução concorrente de transações na rede de testes.

Igualmente, detalha o processo de automatização e sistematização dos testes realizados e seu ferramental associado, através de uma arquitetura orquestrada por uma máquina

controladora que gerenciou as etapas da experimentação através da realização de chamadas sucessivas dos *clusters* concorrenciais, do ajuste do número de repetições em 30 (para obtenção de maior grau de confiança estatística nos dados) e da captura de métricas de interesse.

Com o objetivo de se certificar que a rede de testes escolhida, a *Rinkeby*, suporta transações de ICP em uma *Blockchain* realística, foi realizada a caracterização desta rede sob a perspectiva do número de nós ativos e tamanho da rede, da distribuição geográfica dos nós e do volume de transações. Assim, depreende-se que a *Rinkeby* possui capacidade para suprir as demandas de ICPs, de maneira decentralizada no mundo real. Os resultados da experimentação serão pormenorizados no próximo Capítulo.

5 Avaliação de Desempenho de ICPs Baseadas em *Blockchain*

A próxima fase desta pesquisa é a avaliação de desempenho em si. Para tanto, na Seção 5.1, é justificada e detalhada a escolha de 3 algoritmos de contratos inteligentes que vão dar suporte à ICP. Em seguida, na Seção 5.2, são lançados 1200 experimentos em média, com gastos substanciais de Ether no ambiente experimental e tempos de resposta e variância incremental com a carga oferecida. Para ilustrar estatisticamente as diferenças dos algoritmos, é realizada uma comparação entre eles, por meio de regressão linear, e análise gráfica dos blocos de *Blockchain* utilizados na Seção 5.3.

Em resumo, a avaliação de desempenho sugere que dependendo do aumento de carga, os tempos podem ser relativamente grandes para os piores casos. Porém, os algoritmos da literatura são bastante simplificados. Deste modo, foram realizados, na Seção 5.4, testes com modificação do algoritmo mais intenso utilizando-se de criptografia com *hash*, para um teste ainda mais realístico. As escolhas (*tradeoffs*) entre criptografia, compressão e tempo de resposta dão uma indicação de como é possível melhorar o gasto por um lado, e quanto isto aumenta em termos de tempo de resposta.

Finalmente, com base nas lições aprendidas, na Seção 5.5, também foi realizado um teste exploratório de melhoria, onde os experimentos pudessem ser executados de maneira dispersa no tempo. De modo a aliviar o uso concorrente dos blocos. O resultado é bastante encorajador em como sistemas de ICPs baseados em *Blockchain* podem ser projetados no futuro, baseados nas limitações das redes.

5.1 Escolha das Implementações ICP

As escolhas dos contratos inteligentes, objetos dos experimentos realizados neste trabalho, foram norteadas pelo estado da arte apresentado no Capítulo 3. Entre os critérios para a escolha dos mesmos, tem-se a seguinte lista:

- ICP em *Blockchain* que tenham sido codificados na linguagem Solidity - o argumento

importante é que muitas vezes, propostas da literatura, e em sítios e blogs, tem integrações não-triviais entre um aplicativo *front-end* e o *back-end* de uma DApp. Desse modo, alguns desses trabalhos desenvolvem parte importante da lógica de ICP em *Blockchain* fora da *Blockchain*. Trabalhar diretamente com os *smart contracts* permite eliminar do desempenho, o que é artificio do sistema integrado, do que é artificio da rede de *Blockchain*;

- ICP em *Blockchain* exclusivamente relacionados a trabalhos acadêmicos oficialmente reconhecidos - trabalhar com algoritmos descritos na literatura é mais reproduzível, tem resultados prévios avaliados por pares e publicados e permitem um estudo mais sobre o núcleo das funcionalidades;

- ICP em *Blockchain* disponíveis no github - embora na literatura, em função da necessidade de reproducibilidade, os autores detalham os algoritmos com extremo detalhe, ainda assim, trabalhar com o código original do autor permite eliminar ambiguidades na implementação das ideias; e

- ICP em *Blockchain* que possuam simetria funcional - a estrutura das implementações de ICP em *Blockchain* em Solidity, por vezes, também não é homogênea, algumas tem uma função para tudo, outras são mais modulares. Nesse quesito, foram filtradas implementações que tivessem uma funcionalidade simétrica, ou seja, que cada função de um *smartcontract* pudesse ter função semelhante em outro.

Os contratos inteligentes escolhidos foram EthPKI, SCPKI e Trustery e estão armazenados no github desse projeto [1]. Em especial, os contratos SCPKI e Trustery são respectivamente versões mais elaborada e mais simples da mesma proposta (AL-BASSAM, 2017). Maiores detalhes dos contratos inteligentes eleitos são apresentados a seguir.

5.1.1 EthPKI

A Ethereum PKI (EthPKI) se propõe a implementar funções de ICP, tais como registro de entidades, anexação de certificados, realização de assinatura com certificados anexados e definição do tempo de expiração, revogação de certificados e cenários de testes de validação com certificados válidos e inválidos.

Esta proposta ratifica os esforços e iniciativas do trabalho de (SIVAKUMAR et al., 2017) e também de um tutorial de funções de ICP baseadas em *Blockchain* intitulado "PKI Tools in EVM-based Blockchains", também disponibilizado no github. O autor deste último tem um significativo protagonismo no meio acadêmico relacionado às áreas de *Blockchain*, Matemática, Softwares e Finanças.

[1] https://github.com/rogerioita/ICPChain

Além das funções supracitadas, este contrato possui um método construtor que inicializa a variável *"owner"* com o endereço de conta de quem implantou o contrato inteligente. Dessa forma, a implementação restringe, somente ao possuidor deste endereço, a realização de determinadas atividades neste contrato, como é o caso da função register(). Esse cadastro prévio do endereço do dono do contrato, na sua criação, é ilustrada na Figura 5.1.

```
constructor() public {
    owner = msg.sender;
}
```

FIGURA 5.1 – Método construtor do EthPKI

As principais funções de ICP implementadas pelo EthPKI são as seguintes:

- *register()*

  Esta função realiza o registro de certificados por meio da verificação de propriedade do responsável pela entrada dos dados. Caso o responsável, seja o mesmo que implantou o contrato inteligente, será permitido o registro deste certificado, caso contrário tal registro será negado. O proprietário é como se fosse a AC.

  A função faz uso de dados do tipo "structs". Este tipo de dados, definido pelo usuário, pode conter vários tipos de dados primitivos, tipo *mapping*, estrutura de dados que armazena dados no formato "chave":valor, possibilitando o mapeamento da variável declarada aos tipos primitivos utilizados na formação desta estrutura. Em especial mapeia o tipo "address", tipo primitivo de dados não numérico, de 20 bytes e exclusivo da linguagem Solidity, sendo próprio para o armazenamento dos endereços de contratos inteligentes.

  Sobretudo destaca-se nesta função a validação realizada sobre a entrada dos endereços que possam se candidatar como "sender". Essa checagem na realização do registro evita a execução de uma transação inválida de modo inadvertido ou proposital. A linguagem Solidity possui alguns validadores para este tipo de situação e neste caso foi empregado o validador *require()*.

  Ao usar a validação de entrada *require()*, protege-se do risco de passagem de parâmetros inválidos e igualmente uma transição de estado do contrato de forma não autorizada. Tal validação atua na rejeição das transações e reversão de quaisquer recursos de *gas* ainda não usados pela transação. A Figura 5.2 ilustra a forma como EthPKI trata as validações da entrada sendo enviada para o *smart contract*.

```
// Add a trusted entity. Owner of the PKI only
function register(address trustedEntity, bytes32 publicKey)
    public returns (uint entityId) {
    if (msg.sender == owner) {
        entityId = nEntities++;
        entities[entityId] = Entity(trustedEntity, publicKey, 1);
    } else {
        bool found = false;
        for (uint i=0; i < nEntities; i++) {
            if (msg.sender == entities[i].identity) {
                entityId = nEntities++;
                found = true;
                entities[entityId] =
                    Entity(trustedEntity, publicKey, entities[i].level + 1);
            }
        }
        require(found);
    }
}
```

FIGURA 5.2 – Validação de input utilizando a require()

- *append()*

  Esta função permite realizar a anexação de certificados. Do ponto de vista de implentação em Solidity não traz novidades em relação o que já foi apresentado. É uma função muito simples, faz operações de incrementos e atribuição condicionada ao objeto "msg.sender"usando um vetor por chave/valor.

- *sign()*

  A função *sign()* realiza em teoria a assinatura digital via certificado. De maneira semelhante a função *append()* não apresenta aspectos relevantes e específicos da implementação em Solidity. Na prática, essa função faz uso de mappings dos valores de chave/valor adicionados anteriormente em *append()* para uma validação booleana associada à atividade de assinatura. Também faz uma atualização das listas de certificados revogados.

- *revoke()*

  Esta função *revoke()* realiza a revogação dos certificados, empregando a mesma estrutura de dados chave/valor com atribuição de valores booleanos em uma lista de certificados revogados.

Existem outras funções utilitárias menos relevantes. Portanto, delimitou-se o escopo da apresentação das funções do EthPKI a somente estas apresentadas. Haja vista que são estas que possuem paridade com os demais contratos a serem apresentados. Além disto, fez-se pequenas modificações no código para ficar com a simetria funcional que se estabelecem nos critérios iniciais.

5.1.2 SCPKI

O próximo *smart contract* SCPKI é a versão mais elaborada de (AL-BASSAM, 2017). Este trabalho se propõe a implementar funções de ICPs, tais como registro de atributos, assinatura de atributos com definição do tempo de expiração e revogação de assinatura.

Diferente do EthPKI, o SCPKI não faz uso de método construtor para inicialização de variáveis. Além disto, (AL-BASSAM, 2017) considerou em sua arquitetura o desenvolvimento de uma aplicação cliente para interação com o contrato inteligente em tela. Para realizar esta tarefa, no que tange a implementação do contrato, o autor utilizou-se da funcionalidade de gerência de eventos na linguagem Solidity.

O gerenciamento de eventos na plataforma Ethereum possui as seguintes características:

- a inscrição em evento é realizada por cliente;
- os eventos são armazenados em *log* de transação na Blockchain, tornando-se um registro permanente desde a criação do bloco;
- contratos não podem assinar eventos;
- permite o armazenamento de dados históricos; e
- viabiliza fator de economia para o cliente, visto que nesta perspectiva, o armazenamento na *Blockchain* é mais barato que manter em memória.

Dessa forma, no SCPKI, a aplicação cliente recebe notificações quando são executadas as transações do SCPKI na *blockchain* e as trata segundo a especificidade de cada função.

Em geral, o evento associado a cada função é então registrado. Esse registro inclui todos os parâmetros de entrada da chamada de função juntamente com o identificador chave associado a esta função e o "sender" relacionado. Desse modo, somente os clientes SCPKI atendem a tais requisitos.

No SCPKI, a função *addAttribute()* (Figura 5.3) armazena os dados, respectivamente, de todas as adições de atributos, assinaturas e revogações em uma série de atributos, assinaturas e revogações. Esses atributos podem ser acessados por outros contratos inteligentes por meio do identificador pertinente em uma matriz. Cada ID serviria como índice para cada função na matriz. Funções semelhantes de Anexação (*append*), Assinatura (*sign*) e Revogação (*revoke*) também estão presentes no SCPKI. Algumas pequenas adequações também foram realizadas para completar a simetria funcional, coisas como renomear as funções, mudar algumas estruturas de dados de lugar, desmembrar a função, entre outras.

Devido a essa estrutura de eventos, é possível constatar que o SCPKI possui uma quantidade menor de funções que o EthPKI. Em comparação funcional, as funções do EthPKI são mais simples que no SCPKI. Por exemplo, SCPKI não faz uso de *mappings* e possui a mesma quantidade de *structs* que o EthPKI. Igualmente, não traz nenhuma novidade adicional, do ponto de vista de implentação em Solidity, com exceção da abordagem com gerenciamentos de eventos. A Figura 5.3 ilustra como utiliza-se o gerenciamento de eventos no SCPKI.

```
...
...
event AttributeAdded (uint indexed attributeID, address indexed owner, string attributeType, bool has_proof,
    bytes32 indexed identifier, string data, string datahash );
...
...
function addAttribute (string attributeType, bool has_proof, bytes32 identifier, string data, string datahash)
returns (uint attributeID) {
    attributeID = attributes.length ++;
    Attribute attribute = attributes[attributeID];
    attribute.owner = msg.sender ;
    attribute.attributeType = attributeType ;
    attribute.has_proof = has_proof ;
    attribute.identifier = identifier ;
    attribute.data = data ;
    attribute.datahash = datahash ;
    AttributeAdded (attributeID, msg.sender, attributeType, has_proof, identifier, data, datahash);
}
```

FIGURA 5.3 – Emprego do gerenciamento de eventos em transações de blockchain

5.1.3 Trustery

A última implementação de *smartcontract* foi a Trustery. Esta é a versão mais simples do SCPKI, também proposta por (AL-BASSAM, 2017). A Trustery possui estrutura de funções semelhante à SCPKI e a diferença consiste na não-utilização da matriz de organização de eventos, para salvar os dados provenientes das funções. Neste caso, as funções tendem a ficar mais parecidas com o EthPKI, e apenas incrementa-se o identificador mais recente associado a função em execução.

5.2 Análise dos Experimentos

A fase de experimentos foi conduzida entre 01/10/2020 e 25/01/2021. Primeiramente, consistiu em uma fase de desenvolvimento e testes no primeiro mês. Em seguida, houve uma fase de coleta dos resultados finais para serem relatados nesta pesquisa.

Conforme discutido no Capítulo 4, repetindo de maneira resumida, a metodologia utilizada consistia em vários passos. Em primeiro lugar, na ativação da *Blockchain* realizada pela máquina controladora após a verificação das principais condicionantes para realização dos testes, tais como a criação de contas na quantidade suficiente para a realização

dos testes e, a verificação e balanceamento da quantidade de Ether para as contas.

Um detalhe não apresentado anteriormente, é que devido a necessidade de se executar tais experimentos em mais de uma máquina, foi desenvolvido um script para obtenção de saldo de cada ambiente, conforme mostrado na Figura 5.5. Embora no início, havia uma expectativa de gastos virtuais pequenos, com a intensidade dos testes, por exemplo, de 400 transações concorrentes repetidas 30 vezes, mostrou que o gasto por experimento foi superior a 20 Ethers por dia de experimento. Entretanto, devido a restrições de tempo, não realizamos um estudo mais aprofundado do consumo desses recursos virtuais.

```
i=0
while [  $i -lt 2610 ]; do
a="personal.listAccounts["$i"]"
r="web3.fromWei(eth.getBalance("
g=')'
conta=`geth attach /home/rogerio/.ethereum/rinkeby/geth.ipc --exec $a`
saldo=`geth attach /home/rogerio/.ethereum/rinkeby/geth.ipc --exec ${r}${a}$g$g`

echo "Conta de índice" $i "-" $conta. "Saldo:" $saldo "ETH">> ./saldoEnxuto.txt
let i=i+1
done
```

FIGURA 5.4 – Script para obtenção de saldo de contas

Seguindo o fluxo metodológico, a máquina controladora executa o script de desbloqueio, por exemplo, para 800 contas. E, uma vez que o contrato já se encontrava implantado na *Blockchain* ativa e as contas estavam munidas de Ethers e desbloqueadas, começava a fase de execução.

Esta fase se inicia com script desenvolvido em Python que aciona, de maneira remota usando paramiko, os Node.js relativo a cada conta. Assim, executa-se em ordem serial, o contrato EthPKI com (200, 400, 600 e 800) níveis de concorrência. Este número de transações simultâneas máxima, estabelecido de acordo com a capacidade computacional, e dobrando-se a concorrência, ajuda a ressaltar tendências de alta nos experimentos.

Logo após, executa-se o contrato SCPKI com as mesmas premissas e procede-se de igual forma com o contrato Trustery. Itera-se tais passos em 30 vezes.

Segue abaixo a descrição dos os fatores e variáveis envolvidos na experimentação:

- as Máquinas empregadas são CM e RCS;
- os algoritmos utilizados são EthPKI, SCPKI e Trustery;
- as Funções usadas são Anexação, Assinatura e Revogação; e
- os *clusters* de concorrência são de 200, 400, 600, 800 e 1000 transações.

Multiplicando-se os fatores executados, houve um volume total superior a 1200 experimentos. Embora, não tenha sido o foco do trabalho, mas uma estimativa de tempo real na condução de cada experimento é a seguinte: para 200 transações concorrentes, com 600000 *gas* e algoritmo EthPKI rodando todas as funções, o tempo gasto foi em torno

de 5 minutos. Repetindo o experimento por 30 vezes com esses parâmetros fixos, para a obtenção de média robusta e desvio, o total vai para 2.5 horas para cada combinação.

5.2.1 Métricas de Interesse e Tratamento dos Dados

A realização de um experimento na Blockchain pode trazer vários resultados importantes sobre o objeto de estudo, como mostrado na Figura 5.5, onde se traz informações referentes ao bloco de apenas uma transação realizada no EthPKI.

```
Será realizado o experimento com o
Address:0x1ea05bf9e894c8ef157157456c34a60005b8e407
Anexação, 15054 { blockHash:
   '0xf4b7059d09871b3e261e993afba66d0d63b7230ee8a78613cb12881c2ca91e9a',
  blockNumber: 7957992,
  contractAddress: null,
  cumulativeGasUsed: 457737,
  from: '0x1ea05bf9e894c8ef157157456c34a60005b8e407',
  gasUsed: 92067,
  logsBloom:
'0x000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000',
  status: true,
  to: '0x86b66debf5c264829e9b85a16beb1b58367fbe87',
  transactionHash:
   '0xb04beac715f40553ffe21be21e992568c92b3e8bb33ac469f681c488e00babff',
  transactionIndex: 6,
  events: {} }
Done!
```

FIGURA 5.5 – Informações de uma transação

No entanto, para o propósito deste trabalho de pesquisa, faz-se necessário reunir medidas quantificáveis e objetivas que permitam analisar criteriosamente o desempenho. Sendo assim, foram estabelecidas as seguintes métricas:

1. Tempo de resposta - Gerado automaticamente na execução do experimento, captura o tempo de início e fim de cada transação em milissegundos. Utilizou-se biblioteca do Node.js para coleta dos tempos com o máximo de precisão. Não havia outras aplicações rodando nas máquinas no momento da execução das aplicações; e

2. Gas Utilizado - Gerado automaticamente na execução do experimento, esse parâmetro varia em função da quantidade de execução que acontece na *Ethereum Virtual Machine* (EVM) e também em condições de tamanho de mensagem transferida e armazenada na rede. Foram utilizados valores padronizados como a quantidade de gas ofertada por experimento.

Após o estabelecimento das métricas e a execução dos experimentos, projetou-se um conjunto de dados (*dataset*) contendo somente as médias e desvios para uma análise macro, com 36 entradas para permitir uma melhor exploração, análise e visualização dos resultados.

Nesta etapa, com o auxílio da biblioteca pandas do *Python Data Stack*[2] foi feito tratamento dos dados advindos do passo anterior, por meio de Agregações e seleção de *subsets* de dados, processo de de *Data Muniging / Wrangling* [3], sumariamente consistindo no processo de converter e mapear dados de um estado "bruto", realiza o tratamento de dados de *missing*, bem como o merge e outras operações relacionais com SQL. E em seguida, utilizada da plataforma Jupyter e Excel para a criação dos gráficos multi-dimensionais.

A estrutura final do de *dataset* ficou com o seguinte formato, conforme descreve a Tabela 5.1. Este é um extrato do *dataset* utilizado na experimentação, contemplando apenas uma das 3 transações do *dataset* original.

TABELA 5.1 – Tabela com extrato simplificado do *dadaset*

Transação	Contrato	Concorrência	Máquina	MdTempo	MdGas	DvTempo
Anexação	EthPKI	200	CM	21750.50	92132.00	8619.43
Anexação	EthPKI	400	CM	38119.63	92132.00	18357.61
Anexação	EthPKI	600	CM	54345.72	92132.00	28241.80
Anexação	EthPKI	800	CM-RCS	95392.06	92132.00	47282.42
Anexação	SCPKI	200	CM	33803.29	142629.00	19270.96
Anexação	SCPKI	400	CM	65049.09	142629.00	40913.40
Anexação	SCPKI	600	CM	89067.76	142629.00	47723.63
Anexação	SCPKI	800	CM-RCS	132254.38	142629.00	63796.77
Anexação	Trustery	200	CM	21475.16	34873.00	16543.92
Anexação	Trustery	400	CM	20166.18	34873.00	8236.60
Anexação	Trustery	600	CM	29511.36	34873.00	13290.16
Anexação	Trustery	800	CM-RCS	41354.06	34873.00	20093.20

5.2.2 Visão Holística dos Resultados

A Figura 5.6 apresenta os resultados, utilizando o máximo de variáveis para uma visão holística dos resultados. Optou-se pelo uso da técnica de *facet* que particiona os resultados como uma matriz de painéis. Cada painel mostra um diferente subconjunto de dados. Neste caso, apresenta-se os algoritmos (como uma combinação de suas funções) e observando-se os parâmetros de cargas concorrente, gas utilizado (em Wei, uma fração de Ether) e tempo de resposta médio e seu respectivo desvio padrão (STD).

[2]https://hub.packtpub.com/python-data-stack/
[3]https://medium.com/dados-de-cientista/data-wrangling-x-data-munging-b47bd3a4d555

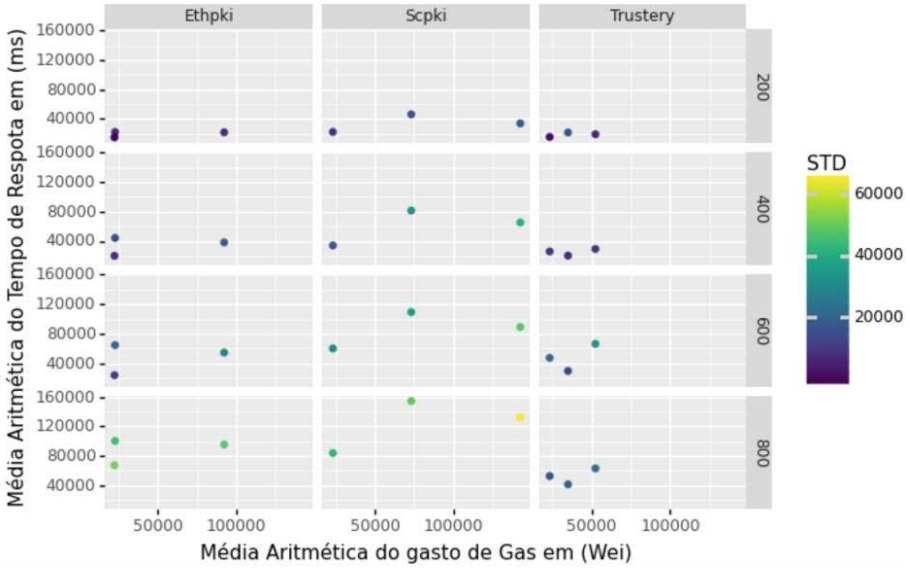


FIGURA 5.6 – Tempo de Resposta por quantidades de Usuários Concorrentes e Contrato Inteligente.

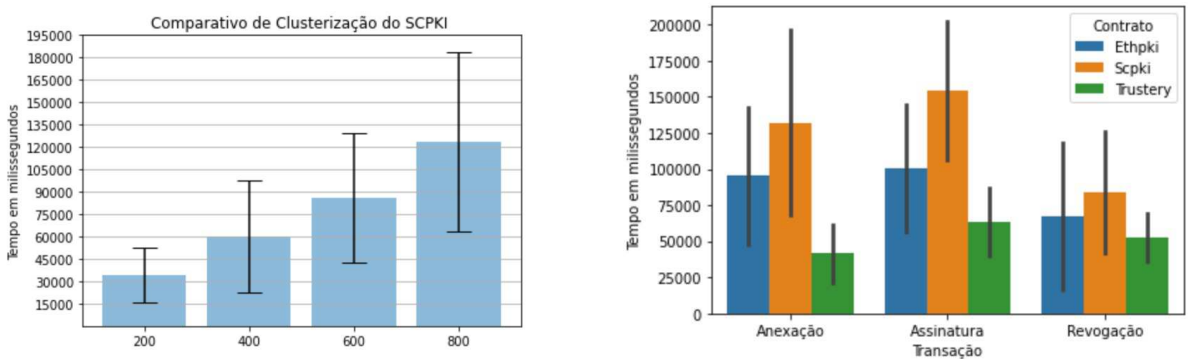
Conforme foi utilizado os tempos médios, as milhares de experimentações ficam reduzidas na figura em único valor médio e, por consequência, único desvio padrão dos dados. Note-se, pelos resultados da Figura 5.6, que os tempos de resposta médios variam ao longo dos experimentos entre minimamente inferior a 40 mil milisegundos (40 segundos) e alguns resultados destoam para 160 segundos. Como era esperado, os tempos de resposta são menores para cargas menores de 200 e 400 transações concorrentes versus cargas maiores.

Com carga baixa de 200 transações concorrentes, o EthPKI e o Trustery possuem resultados de tempo de resposta inferiores a 40 segundos e baixo desvio padrão. No caso do SCPKI, os tempos médios estão próximos a 40 segundos com diferentes valores de gas consumidos, o que indica que esse é o algoritmo mais intenso.

Ao passar para cargas mais altas como 400 em diante, as médias de SCPKI sobem rapidamente, e a variação também é a mais elevada. O algoritmo de Trustery para 800 transações concorrentes têm os menores tempos de resposta e a menor variação. Mesmo em termos de gas consumido, o valor não varia muito. Portanto, apresenta-se bastante uniforme, de baixo custo e consistente.

5.2.3 Detalhamento por Carga e Função

Ao detalhar os resultados e ir separando por categoria, pode-se observar outros *trade-offs* nos resultados. Na Figura 5.7a tem-se uma comparação em separado do SCPKI para diferentes tamanhos de carga. A barra em azul representa a média dos tempos de resposta de todas as funções, e a linha em preto representa o desvio médio. De forma similar ao que havia sido observado na Figura 5.6, os tempos de resposta médios são crescentes com



(a) Aumento da carga no SCPKI (b) Comparativo de funcionalidade simétrica

FIGURA 5.7 – Detalhamento dos resultados de carga e função

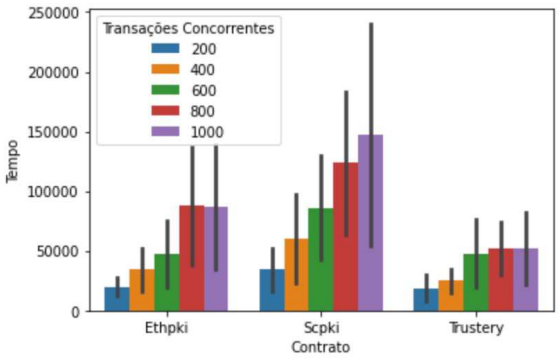
o aumento da carga e o desvio também parece aumentar proporcionalmente com maior carga. Em outras palavras, dobrando a carga, dobra-se a média e, igualmente, dobra-se o desvio.

Em seguida, são observados os dados separando individualmente por funções na Figura 5.7b. A ideia é ter um comparativo sobre a intensidade na execução de funções que foram adequadas para serem simétricas. No caso, todos os algoritmos EthPKI, SCPKI e Trustery possuem 3 funções simétricas de Anexação, Assinatura e Revogação. Fazendo a comparação, nota-se que as funções de Anexação e Assinatura são mais intensa para todos os algoritmos. Enquanto a função de revogação é a mais simples e estável. Entre os algoritmos, conforme já havia sido observado, o SCPKI é o mais intenso de todos, sendo que sua função de assinatura tende a ser o dobro da mesma função no Trustery. A utilização de recursos avançados de gerenciamento de eventos parece ser um fator importante nesse desempenho.

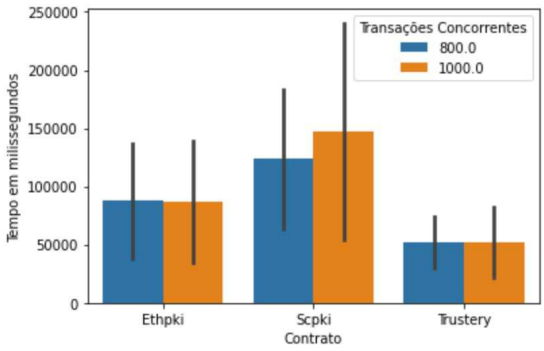
5.2.4 Comparativo com Cargas Extremas

Na medida em que a carga vai aumentando, os algoritmos vão aumentando seus tempos de resposta e desvios também, mas não de uma maneira uniforme entre eles. Conforme mostrado na Figura 5.8a, para os algoritmos de EthPKI e SCPKI, este comportamento de dobro de média do tempo de resposta e dobro de desvio se mantém. Porém, para o Trustery, o comportamento é muito mais suave, com baixo desvio. Neste gráfico da Figura 5.8a, também foram incluídos testes com 1000 transações concorrentes. Este último foi um experimento introduzido tarde na pesquisa, e que necessitava de duas máquinas ao mesmo tempo, devido à quantidade de CPU, memória e disco que 600 a 1000 usuários concorrentes utiliza.

Os resultados para cargas extremas como 1000 transações concorrentes, para melhor



(a) Aumento de carga entre algoritmos



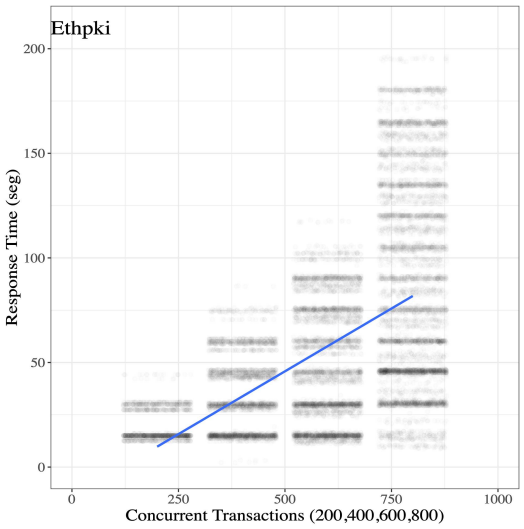
(b) Comparando 800 e 1000 transações concorrentes

FIGURA 5.8 – Detalhamento de resultados com cargas extremas

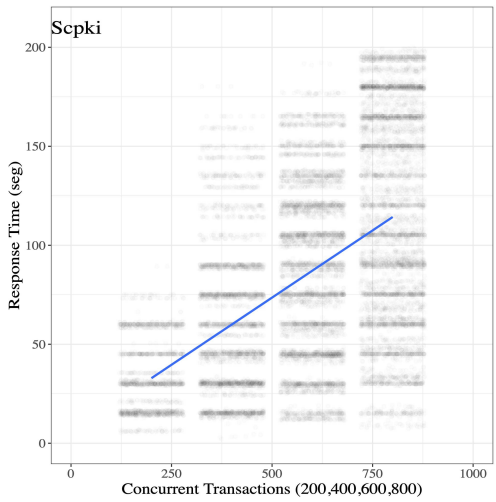
visualização, podem ser comparados com a carga 800 transações. Colocando lado a lado os algoritmos e medindo os tempos de resposta médios e os desvios temos a Figura 5.8b. Novamente, constata-se o mesmo comportamento de 800 transações, ou seja, não afetando tanto a média e desvio de EthPKI e Trustery mas impactando mais o SCPKI.

5.3 Análise Comparativa por Regressão Linear

Embora o estudo com os valores médios possam dar uma dimensão sobre os dados, observar mais a fundo e estimar modelos de regressão permitem uma comparação algorítmica mais adequada. Nos resultados apresentados nesta Seção, tem-se os tempos de respostas plotados para todos, literalmente todos os experimentos. Com isto, usou-se da robustez dos dados para traçar retas de regressão linear, a partir dos experimentos.



(a) Regressão Linear EthPKI



(b) Regressão Linear SCPKI

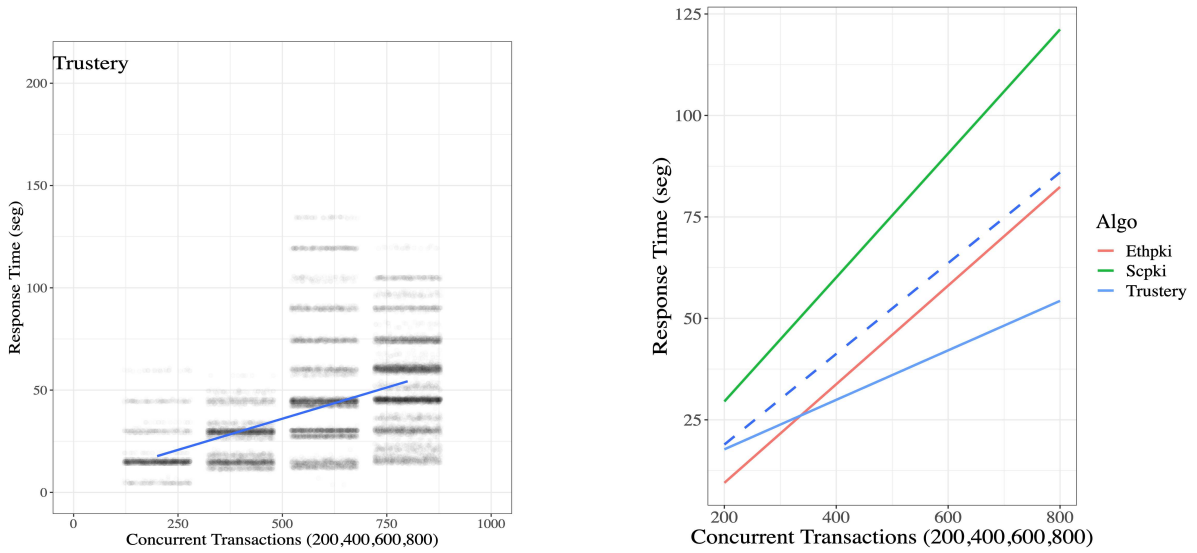
FIGURA 5.9 – Comparação das Regressões Lineares da EthPKI e SCPKI

As Figuras 5.9a e 5.9b apresentam os algoritmos EthPKI e SCPKI. Para estes gráficos, os pontos encontram-se espalhados no eixo-x, em torno da carga fixa de transações concorrentes, para melhorar a visualização.

O efeito disso na visualização é chamado de "jitter" nos gráficos e espalha pontos que estariam sobrepostos, permitindo ter outras perspectivas sobre os dados. Deste modo, todos os pontos que se encontram espalhados entre 150 e 250, na verdade foram executados com 200 transações concorrentes.

Por outro lado, os pontos de tempo de resposta estão clusterizados, em 15 segundos e 30 segundos. Esta informação faz muito sentido, pois a *Blockchain* da Rinkeby opera escrevendo blocos a cada 15 segundos. Então, se algumas das 200 transações concorrentes não são escritas na primeira vez que aparece o bloco, vão aparecer no segundo bloco, após 30 segundos.

A Figura 5.9a, em especial, possui outra característica interessante no sentido que os pontos, mesmo espalhados, ainda tem um certo nível de sobreposição. Portanto, ao calibrar o nível de transparência dos pontos, percebe-se que certos "patamares" de tempo de resposta são mais "frequentes" que outros. Ou seja, quanto mais escuros a zona de pontos, mais pontos estão nessa área. Em outras palavras, na Figura 5.9a, parece emergir um aspecto de código de barra em degradê, o que pode indicar que mais transações são processadas primeiro em termos de blocos, e depois menos e menos transações são processadas em blocos seguintes na *Blockchain*. Estes resultados são mais bem detalhados na Seção 5.5 onde mostramos a distribuição das transações por bloco da *Blockchain*.



(a) Regressão Linear Trustery (b) Comparando com Regressões Lineares

FIGURA 5.10 – Comparação da Regressão Linear da Trustery e Visão Geral

Dando continuidade ao estudo, para o algoritmo Trustery apresentado na Figura 5.10a, o resultado é que tem melhor distribuição das requisições nos blocos de tempo de resposta

e melhor desempenho em geral. Colocando todos as regressões lineares juntas temos a Figura 5.10b.

Nessa comparação lado a lado, é possível verificar que SCPKI e Trustery são quase antagônicos, do ponto de vista de desempenho. Enquanto EthPKI se parece mais com a regressão linear de todos os pontos, independente de algoritmos, representada pela linha azul tracejada.

Convertendo essas curvas de regressão linear em suas formas matemáticas tem-se as Equações 5.1, 5.2 e 5.3.

$$EthPKI = -14829.94 + 121.51 * TransacoesConcorrentes + e \tag{5.1}$$

$$SCPKI = -1038.6 + 152.79 * TransacoesConcorrentes + e \tag{5.2}$$

$$Trustery = 5535.88 + 60.97 * TransacoesConcorrentes + e \tag{5.3}$$

Note-se, que o termo multiplicativo das equações é o fator de escalabilidade delas, quanto menor o fator, mais escalável e com melhor tempo de resposta. No caso da SCPKI, como foi verificado, ao longo desta pesquisa, trata-se do algoritmo mais custoso e lento, e o fator multiplicativo é 152.79 multiplicado pelo número de transações concorrentes. No futuro, outras equações podem ser derivadas também para o pior caso, pois o desvio é grande para todos os algoritmos.

5.4 ICP em *Blockchain* com Criptografia Embarcada

Em face dos resultados de desempenho obtidos, dada a necessidade de se aguardar talvez períodos de até 100 segundos para validar ou revogar um contrato digital, parece razoável que este tipo de abordagem tenha espaço como solução para ICPs de maneira geral.

Entretanto, um aspecto inerente dos algoritmos analisados é que eles não fazem uso intenso de funções criptográficas internas ao *smart contract*. Em outras palavras, os certificados digitais são alimentados de fora da rede, e as revogações acontecem em listas de certificados pré-cadastrados. Nota-se uma clara falta de funções de criação de chaves pública e privada e mesmo criação de *hashes* de validação.

As motivações para essa falta de uso do que se chama de criptografia embarcada vão desde custos, porque funções mais intensas podem ter custo para rodar em EVM em *gas*

TABELA 5.2 – Comparativo entre experimentos com 200 e 400 usuários concorrentes, com e sem criptografia

Comparativo entre experimentos com 200 e 400 usuários concorrentes, com e sem criptografia. Tempo medido em milissegundos.	
Experimento com 200 transações concorrentes sem criptografia.	
Média de atraso	Desvio
20.279	8.438
Experimento com 200 transações concorrentes com criptografia.	
Média de atraso	Desvio
20.748	8.184

Diferença entre médias: 469 ms

Experimento com 400 transações concorrentes sem criptografia.	
Média de atraso	Desvio
30.394	16.389
Experimento com 400 transações concorrentes com criptografia.	
Média de atraso	Desvio
32.076	16.341

Diferença entre médias: 1.682 ms

mais elevados. Até limitações de mineradores no poder dos algoritmos de computação que podem rodar na rede experimental ou mesmo na rede principal *Ethereum*.

Para demonstrar esse aspecto e como ele pode ser influenciador no desempenho da ICP foram executados 4 experimentos, variando-se o número de transações concorrentes e introduzindo-se ou não uma função de *hash256* diretamente no código das funções do contrato inteligente. Os resultados podem ser verificados na Tabela 5.2.

No caso dos experimentos com 200 usuários concorrentes quando adicionada a criptografia, por meio da função *hash256* nota-se que o tempo médio de atraso aumenta em 469 milissegundos. O desvio médio não muda muito inferior a 200 milissegundos. Então, dessa forma, é possível quantificar que a execução da função mais intensa tem um revés não-trivial na execução do contrato. Entretanto, considerando que o tempo total médio é de 20 segundos, trata-se de 2 a 3% de aumento.

Por outro lado, ao dobrar a quantidade de transações para 400 concorrentes, a expectativa é que o tempo de resposta (média de atraso, na Tabela 5.2) dobrasse. Porém, é importante recordar que o contrato inteligente é implantado em uma única EVM na rede Rinkeby, e dependendo da carga e necessidade de processamento da função criptográfica pode gerar um gargalo nesse contrato.

Os gargalos tendem a ter um comportamento de atraso diferente do crescimento linear, com o aumento de carga. Portanto, no caso do experimento com 400 transações concorrente, pode-se verificar que a diferença entre médias com e sem a função *hash256* aumenta para 1682 milissegundos. Em outras palavras, um aumento de 3.58 vezes, demonstrando que o gargalo já começa a aparecer nesse caso. Portanto, ao trabalhar com a ICP em

Blockchain com Criptografia Embarcada no próprio *smart contract* pode gerar gargalos de difícil observação quando comparados com ICP mais simples, como as que executamos nos testes de desempenho.

Um aspecto que não se encontra mostrado na Tabela 5.2, mas que precisa de maior investigação, é referente ao *gas* consumido nestes testes com a criptografia embarcada. Ele foi menor na média, nos testes com criptografia, comparado com os sem criptografia. A justificativa é que a função de retorno no caso da função sem *hash* retornam um valor binário muito grande do contrato inteiro. Enquanto a função de retorno, no caso da função com *hash*, retorna o próprio *hash* que é um valor compactado. Deste modo, o custo de rede em termos de bytes transferidos caiu e, portanto, o gas seguiu este movimento.

5.5 ICP em *Blockchain* com Escalonamento Aleatório

Esta Seção apresenta um segundo experimento exploratório de proposta de melhoria para o desempenho da ICP baseada em *Blockchain*. Conforme foi notado na Seção 5.3, com as transações se concentrando em tempos de resposta espaçados de 15 em 15 segundos, parece sugerir que a rede Rinkeby realiza algum tipo de degradação à medida que as transações vão sendo colocadas nos blocos. Porém será notado nos resultados que as degradações são no processamento das transações e não necessariamente em qual bloco elas são inseridas.

Para ilustrar esse aspecto e trazer os resultados de melhoria, como as transações concorrentes muitas vezes tem quantidades maiores de transações do que o bloco suporta. Então, muitas delas tendem a ficar em algum tipo de fila de espera. No caso da Rinkeby, é calculado que em função da quantidade de *gas* e dos mineradores, que a quantidade máxima de transações por bloco é de cerca de 200 [4]. O tráfego de fundo, e outras condições da rede fazem com que as transações dos experimentos ocupem parte disso.

Então, foi realizada uma modificação do código para suportar a dispersão das transações de maneira puramente aleatória. Basicamente, a ideia é que os experimentos possam iniciar em tempos aleatórios módulo de 15 segundos, em outras palavras, joga um dado virtual e caindo 1 multiplica por *sleep* 15 segundos, 2 aciona um *sleep* 30 segundos, e assim sucessivamente.

A justificativa dessa modificação de Escalonamento Aleatório é que o programa de execução da carga não deveria sobrecarregar um bloco de 15 segundos com mais de 200 transações, pois este é o máximo possível do ponto de vista da Rede Experimental Rinkeby. Assim, muitas das transações não ficam no *pool* de transação não-confirmadas, que

[4]https://medium.com/ethereum-grid/ethereum-101-how-are-transactions-included-in-a-block-9ae5f491853f

consomem recursos da rede experimental.

TABELA 5.3 – Tabela de Experimentos com Carregamento de Transações - Comparação Aleatório e Fixo

Experimentos com Carregamento Fixo e com Carregamento Aleatório					
Cluster com 200 transações com Carregamento Aleatório					
Tempo Médio de Transações 15449			Média de gas usado 45616		
	Blocos	Transações	Anexação	Assinatura	Revogação
	Bloco 1	2	2	0	0
	Bloco 2	89	87	2	0
	Bloco 3	153	64	87	2
	Bloco 4	198	47	64	87
	Bloco 5	111	0	47	64
	Bloco 6	47	0	0	47
Total	6	600	200	200	200
Tempo Médio de Transações 21024			Média de gas usado 45688		
	Blocos	Transações	Anexação	Assinatura	Revogação
	Bloco 1	92	92	0	0
	Bloco 2	82	82	0	0
	Bloco 3	200	26	174	0
	Bloco 4	200	0	26	174
	Bloco 5	26	0	0	26
Total	5	600	200	200	200

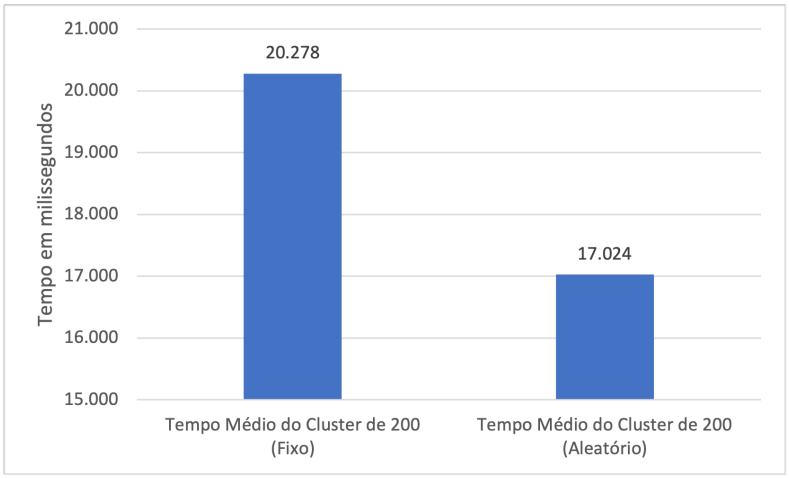
Para apresentar os resultados, será utilizada a Tabela 5.3. Primeiramente, para esta tabela apresentam-se os resultados do esquema de carregamento de execuções em modo aleatório para uma carga de 200 transações concorrentes. Os experimentos sempre são executados com o *smart contract* completo, sendo que as funções executadas são Anexação, Assinatura e Revogação (200 de cada, ou seja, 200 usuários executando a mesma transação de uma só vez). Cada função é executada em série, uma após a outra.

As transações, dependendo da aleatoriedade do começo da execução do programa, podem pegar o final de um tempo de bloco ou o começo do mesmo. Depois em função do escalonamento das operações por função, elas são processadas em cada bloco da *Blockchain*. Esse valor total das transações realizados nos experimentos e que entraram em um bloco está descrito no campo “Transações”.

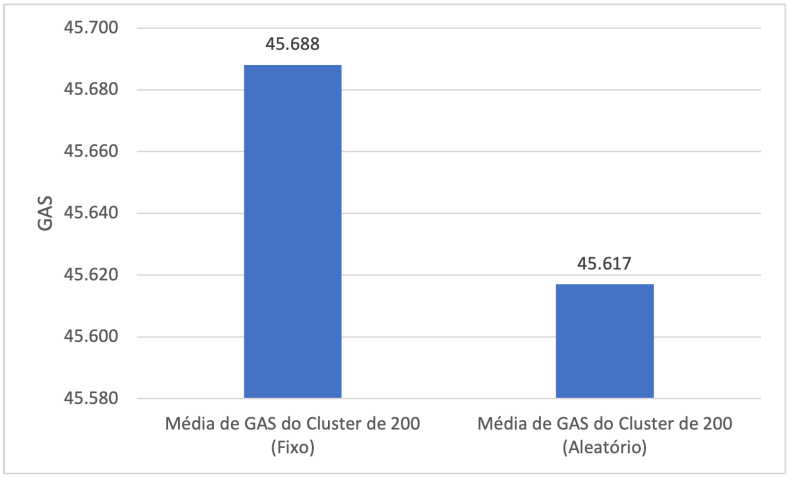
É possível se verificar também que existe algum tráfego de fundo no momento dos experimentos, pois as transações por bloco em alguns casos representam menos que 200

por bloco. Outro aspecto interessante é que apesar das transações estarem distribuídas entre múltiplos blocos, o tempo médio da transação é contado do ponto em que ela é colocada para executar até quando finaliza. Portanto, os tempos médios de transações nesta **amostra** da Tabela 5.3 mostra que o esquema aleatório tem melhor desempenho com 15.449 segundos em média, versus 21.024 segundos em média para as transações desta amostra em si. O *gas* quando utilizando o esquema de escalonamento aleatório também é menor.

Esse resultado da **amostra** da Tabela 5.3, se trata de 1 única execução. Tal padrão, de transações e espalhamento por bloco, também se repete nos resultados das 30 execuções.



(a) Análise em função do tempo



(b) Análise em função do gasto de *gas*

FIGURA 5.11 – *Clusters* com carregamento fixo e com carregamento aleatório em função do *gas* e do tempo.

Para ilustrar essa relação entre transações com menor tempo quando escalonadas e espaçadas no tempo, foi feito o cálculo médio de todos os tempos médios dos 30 experimentos com 200 transações usando Cluster Fixo (ou seja, rajada sem dispersão) e Cluster Aleatório (com dispersão aleatória), conforme mostram Figura 5.11a e Figura 5.11b. Os

tempos de resposta, em milissegundos, são 17% menores quando utilizando-se de um escalonamento mais espalhado no tempo e, portanto, com menor uso de recursos de *pool* de transações não-confirmadas.

Por outro lado, o *gas* consumido também é menor no escalonamento aleatório, porém, bem menor a economia, menor que 0.01%. Ou seja, os custos e intensidade das mensagens são equivalentes, mas o processo de escalonamento alivia a rede da *Blockchain* e as transações não-confirmadas, melhorando o desempenho em termos de tempo, de maneira geral.

5.6 Síntese do Capítulo 5

Este Capítulo apresentou a avaliação de desempenho de 3 algoritmos de contratos inteligentes de ICPs baseadas na tecnologia *Blockchain*, utilizando-se da arquitetura desenvolvida e descrita no Capítulo 4. Esta avaliação teve por objetivo o estudo de desempenho frente a escalabilidade em rede realística e adotou-se, como métricas de interesse, o tempo de resposta e o gas, ambos capturados automaticamente.

Inicialmente, a apresentação dos resultados foi disposta em uma visão Holística, utilizando o máximo de variáveis capturadas nos testes, considerando o tempo de resposta médio e seu respectivo desvio padrão, a média de gas gasto, os algoritmos com suas respectivas funções e os *cluters* de concorrência.

Em seguida os resultados foram detalhados por categorias, a fim de se observar outros *tradeoffs* decorrentes. Assim, projetou-se os resultados em função de seus *clusters* e pelas funções dos contratos inteligentes. Por fim, foram comparados os *clusters* de cargas mais extremas, de 800 e 1000 transações concorrentes.

Além disso, foi realizada a análise comparativa dos 3 algoritmos, por meio de regressão linear utilizando-se todos os dados automaticamente capturados, a fim de se obter a demonstração estatística entre as diferenças destes algoritmos. Cabe ressaltar que os resultados do desempenho holístico convergem com o resultado da análise por regressão linear, pricipalmente, no que se refere ao comprotamento dos algoritmos em função das cargas submetidas.

Por fim, foram realizados 2 experimentos extra, explorando a inserção de função criptográfica e a mudanças no escalonamento do início das transações. O próximo Capítulo condensa as conclusões, contribuições, recomendações e sugestões para trabalhos futuros em destaque.

6 Conclusão

Com a crescente expansão de serviços ofertados em ambiente envolvendo a tecnologia de *Blockchain*, reestruturando modelos de negócio, permitindo maior liberdade de pagamento e aumento de segurança e, sobretudo, ressignificando os serviços de certificação digital, o desenvolvimento das Aplicações Descentralizadas (DApps) para Infraestruturas de Chaves Públicas (ICPs) tem uma boa perspectiva no futuro. Entretanto, existem muitas questões de pesquisa ainda não resolvidas no uso real destas aplicações para assinar documentos de usuários.

Por um lado, tem-se que lidar com as garantias inerentes aos modelos de *Blockchain* como imutabilidade dos dados, remoção de entidades de terceiros no processo e remoção do ponto único de falha. Porém, imaginando cenários, por exemplo de um hospital, onde milhares de usuários precisam assinar documentos, receitas médicas, comprovantes de injeção de medicamentos, fica o questionamento se este tipo de sistema pode ter bom desempenho.

Nesse contexto, esta pesquisa teve por objetivo principal investigar o desempenho e a escalabilidade de aplicações descentralizadas de ICPs, utilizando a tecnologia de *Blockchain* em uma rede experimental empírica Rinkeby na Internet.

6.1 Contribuições

Esta Seção descreve, de forma resumida e quantitativa, as principais contribuições deste trabalho de pesquisa.

Primeiramente, foi realizado um levantamento bibliográfico, usando uma revisão sistemática sobre o estado da arte da literatura, onde os principais artigos foram resumidos, organizados e classificados como suporte às ideias desta pesquisa. Durante esta investigação, ao se constatar que a curva de aprendizado de ambientes de desenvolvimento e experimentação envolvendo contratos inteligentes e utilizando a tecnologia de *Blockchain* não era trivial para se aumentar produtividade, foi desenvolvido um ambiente de teste (*testbed*) e um ferramental que permitisse executar experimentos automatica e repetitiva-

mente com pouca parametrização.

O ferramental desenvolvido envolveu o financiamento automático de contas vinculadas à tecnologia de *Blockchain* Ethereum e os saldos foram utilizados em experimentos. Além disto, entre várias opções de redes de experimentação baseadas em *Blockchain* Ethereum, optou-se pela rede Rinkeby, por sua pequena escala, cobertura global e possibilidade de introduzir poucos ruídos em experimentos.

Uma vez escolhida a rede de experimentação Rinkeby, foram conduzidos experimentos iniciais para caracterização de seu desempenho, para se ter uma noção mais precisa sobre a rede e melhor interpretar seus resultados de desempenho. Os principais experimentos desta caracterização focaram na medição de aspectos como distribuição de atraso, quantidade média de transações e número de nós ativos.

A contribuição principal desta pesquisa consistiu na realização de mais de 1200 experimentos de geração de testes com cargas variadas, utilizando-se 3 algoritmos de *smart-contract*, codificados na linguagem Solidity de ICPs utilizando a tecnologia de *Blockchain*. Estes algoritmos foram escolhidos por critérios científicos de praticidade, disponibilidade de código e equivalência funcional entre eles. Os resultados dos experimentos mostraram que o crescimento do tempo de resposta é função da quantidade de carga concorrente e quando se dobra a carga, dobra-se linearmente também o tempo médio de transações e o desvio dos tempos de resposta.

Durante esta investigação encontrou-se diferenças importantes de desempenho entre os algoritmos utilizados. O Trustery se mostrou substancialmente melhor do que o EthPKI e o SCPKI, em todos os cenários. Ao ser realizada uma modelagem por regressão linear dos tempos de resposta, observou-se intrinsicamente que as aplicações de ICPs baseadas na tecnologia de *Blockchain*, em termos de escalabilidade, são limitadas pela quantidade máxima de transações por bloco da *Blockchain*. Valor este que varia em função da remuneração de mineradores e constantes atualizações de protocolo.

Após concluídos os experimentos principais para se medir desempenho, foram propostos dois experimentos adicionais e exploratórios, modificando-se aspectos dos algoritmos para melhor entender sua influência no desempenho. Inicialmente, no 1º experimento foi adicionada uma função criptográfica embarcada no próprio contrato inteligente (*smart-contract solidity*). A ideia é que as ICPs, se utilizando da tecnologia de *Blockchain*, se tornem mais sofisticadas no futuro. O resultado obtido foi um aumento no tempo de resposta não-linear, o que parece indicar um gargalo em termos de desempenho. Ao mesmo tempo, constatou-se uma redução em taxas de gas, devido a compressão de dados na rede.

O 2º experimento baseou-se nas observações dos experimentos de desempenho, onde transações foram distribuídas entre *slots* de tempo, a partir de uma temporização de 15 segundos na *Blockchain* da rede Rinkeby. Neste caso, certas transações puderam

ficar pendentes, por longo tempo, ao serem disparadas juntas, no *pool* de transações não-confirmadas. A partir daí, desenvolveu-se um algoritmo para escalonar os *clusters* de transações em tempos aleatórios para suavizar os esforços dos mineradores quando submetidos a um grande volume de transações. Ao comparar-se esta abordagem com a que encaminhava todas as transações ao mesmo tempo, constatou-se uma melhoria nos tempos de execução. Verificou-se uma diferença entre o tempo médio de execução destes experimentos que ultrapassava em 14% a média dos resultados obtidos.

Em resumo, os resultados apontam que ainda existe muito a se fazer para que ICPs utilizando a tecnologia de *Blockchain* se tornem uma realidade. Porém, ao se considerar intervalos de tempos múltiplos de 15 segundos, apesar de ainda não serem realísticos, podem ser justificados ou considerados como aceitáveis para operações de longa duração, como por exemplo, na criação de certificados, na checagem e revogação de assinaturas, entre outras operações com características similares.

6.2 Trabalhos Futuros

Muitos aspectos não puderam ser abordados e aprofundados nesta pesquisa, devido à necessidade de redução do seu escopo e ao seu reduzido tempo de desenvolvimento.

Embora a rede Rinkeby tenha sido escolhida por sua consistência e baixo nível de ruído, ainda assim, ela consegue operar satisfatoriamente com um mecanismo de consenso diferente da *Blockchain* Ethereum. Na Ethereum, usa-se Prova de Trabalho (Proof of Work - PoW) como mecanismo de consenso para realizar a escolha da autoridade habilitada a incluir o bloco, enquanto na rede Rinkeby usa-se Prova de Autoridade (Proof of Authority - POA) para tornar o processo de geração de blocos mais seguro.

Essa diferença em relação a criação dos blocos gera mais variabilidade na rede Ethereum. Portanto, se por um lado seria mais realístico, por outro, não seria simples separar os tempos de desempenho desta variabilidade. Deste modo, para trabalhos futuros, com base nos resultados desta pesquisa, sugere-se realizar novos experimentos com uma rede experimental ainda mais próxima da Ethereum, como a rede Ropsten.

Outra sugestão de investigação em trabalhos futuros seria a exploração com maior profundidade de experimentos com escalonamentos aleatórios. Mais especificamente, para se propiciar sistemas de *Blockchain* obter estimativas de números de transações concorrentes e oferecer janelas de escalonamentos adaptativas, ao invés de fixas, como utilizado nos experimentos. Outra possível linha de investigação sugerida para trabalhos futuros seria como um *pool* de transações pendentes conseguiria operar no baixo nível, com quais regras de prioridade e como poderia ser melhorado.

A elaboração dos algoritmos de ICPs baseados em *Blockchain* deixaram muito a desejar, do ponto de vista de características realísticas. Sugere-se combinar as melhores características dos algoritmos envolvidos nesta pesquisa em um novo algoritmo, escalável, para ser colocado em produção, em um estudo de caso real utilizando-se um novo algoritmo com usuários reais, em novas campanhas de medições.

6.3 Limitações deste Trabalho de Pesquisa

Apesar de alcançar todos os objetivos e realizar as contribuições para a área de ICPs baseadas na tecnologia *Blockchain* anteriormente citados na Seção 6.1, entretanto, algumas limitações foram observadas, ao longo deste estudo.

Iniciando com o aspecto do financiamento dos experimentos na Rinkeby. A princípio, quando os experimentos foram idealizados, imaginou-se que os gastos virtuais seriam baixos, pois as transações eram simples e a rede experimental também era simples e pequena. Porém, o aumento da quantidade de experimentos e repetições conduziram a gastos virtuais equivalentes a dezenas de milhares de dólares.

Portanto, tornou-se limitante, ter que se consultar continuamente o *Faucet Rinkeby* para se recarregar as carteiras e, deixando-se de lado a exploração de aspectos importantes como a mudança de gas oferecido, para otimizar as transações, mesmo porque a rede Rinkeby não propicia muita variação, mas na Ethereum, este seria um aspecto relevante.

A quantidade de CPUs e de memória dos computadores utilizados, apesar de substancial (8 cores, 64 GB RAM) são muito exigidas pelo aplicativo Geth e isto limitou que testes com carga superiores a 1.000 transações tivessem que ser conduzidos com 2 máquinas, ao invés de uma só. Mas isto, não se mostrou uma limitação grave, porque os resultados foram considerados consistentes para as cargas utilizadas.

Referências

AAS, J.; BARNES, R.; CASE, B.; DURUMERIC, Z.; ECKERSLEY, P.; FLORES-LóPEZ, A.; HALDERMAN, J. A.; HOFFMAN-ANDREWS, J.; KASTEN, J.; RESCORLA, E.; SCHOEN, S.; WARREN, B. Let's encrypt: An automated certificate authority to encrypt the entire web. In: **Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security**. New York, NY, USA: Association for Computing Machinery, 2019. (CCS '19), p. 2473–2487. ISBN 9781450367479. Disponível em: <https://doi.org/10.1145/3319535.3363192>.

ADAMS, C.; LLOYD, S. **Understanding PKI: concepts, standards, and deployment considerations**. [S.l.]: Addison-Wesley Professional, 2003.

AL-BASSAM, M. Scpki: a smart contract-based pki and identity system. In: **Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts**. [S.l.: s.n.], 2017. p. 35–40.

ALI, M.; NELSON, J.; SHEA, R.; FREEDMAN, M. J. Blockstack: A global naming and storage system secured by blockchains. In: **2016 {USENIX} Annual Technical Conference ({USENIX}{ATC} 16)**. [S.l.: s.n.], 2016. p. 181–194.

AXON, L. Privacy-awareness in blockchain-based pki. **Cdt technical paper series**, 2015.

BADR, B.; HORROCKS, R.; WU, X. B. **Blockchain By Example: A developer's guide to creating decentralized applications using Bitcoin, Ethereum, and Hyperledger**. [S.l.]: Packt Publishing Ltd, 2018.

BASHIR, I. **Mastering Blockchain: A deep dive into distributed ledgers, consensus protocols, smart contracts, DApps, cryptocurrencies, Ethereum, and more, 3rd Edition**. Packt Publishing, 2020. ISBN 9781839211379. Disponível em: <https://books.google.com.br/books?id=ZZ_6DwAAQBAJ>.

BRENNAN, C.; LUNN, W. Blockchain: the trust disrupter. **Credit Suisse Securities (Europe) Ltd.: London, UK**, 2016.

CHEN, H.; PENDLETON, M.; NJILLA, L.; XU, S. A survey on ethereum systems security: Vulnerabilities, attacks, and defenses. **ACM Computing Surveys (CSUR)**, ACM New York, NY, USA, v. 53, n. 3, p. 1–43, 2020.

CHOKHANI, D. S.; FORD, D. W. S. **Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework**. RFC Editor, mar. 1999.

RFC 2527. (Request for Comments, 2527). Disponível em: <https://rfc-editor.org/rfc/rfc2527.txt>.

DANNEN, C. **Introducing Ethereum and solidity**. [S.l.]: Springer, 2017.

DONG, Y.; KIM, W.; BOUTABA, R. Conifer: centrally-managed pki with blockchain-rooted trust. In: IEEE. **2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)**. [S.l.], 2018. p. 1092–1099.

DU, W. **Computer Security: A Hands-on Approach**. CreateSpace Independent Publishing Platform, 2017. ISBN 9781548367947. Disponível em: <https://books.google.com.br/books?id=YPWlswEACAAJ>.

DURAND, A.; GREMAUD, P.; PASQUIER, J. Decentralized web of trust and authentication for the internet of things. In: **Proceedings of the Seventh International Conference on the Internet of Things**. [S.l.: s.n.], 2017. p. 1–2.

FROMKNECHT, C.; VELICANU, D.; YAKOUBOV, S. A decentralized public key infrastructure with identity retention. **IACR Cryptology ePrint Archive**, v. 2014, p. 803, 2014.

HOMESTEAD, E. **Ethereum Homestead Documentation**. 2016.

HU, Y.-C.; LEE, T.-T.; CHATZOPOULOS, D.; HUI, P. Hierarchical interactions between ethereum smart contracts across testnets. In: **Proceedings of the 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems**. [S.l.: s.n.], 2018. p. 7–12.

INFANTE, R. **Building Ethereum DApps: Decentralized Applications on the Ethereum Blockchain**. Manning Publications, 2019. ISBN 9781617295157. Disponível em: <https://books.google.com.br/books?id=wdh2tgEACAAJ>.

IYER, K. **Building Games with Ethereum Smart Contracts**. [S.l.]: Springer, 2018.

JIANG, W.; LI, H.; XU, G.; WEN, M.; DONG, G.; LIN, X. A privacy-preserving thin-client scheme in blockchain-based pki. In: IEEE. **2018 IEEE Global Communications Conference (GLOBECOM)**. [S.l.], 2018. p. 1–6.

KFOURY, E. F.; KHOURY, D.; ALSABEH, A.; GOMEZ, J.; CRICHIGNO, J.; BOU-HARB, E. A blockchain-based method for decentralizing the acme protocol to enhance trust in pki. In: IEEE. **2020 43rd International Conference on Telecommunications and Signal Processing (TSP)**. [S.l.], 2020. p. 461–465.

KIM, S. K.; MA, Z.; MURALI, S.; MASON, J.; MILLER, A.; BAILEY, M. Measuring ethereum network peers. In: **Proceedings of the Internet Measurement Conference 2018**. New York, NY, USA: Association for Computing Machinery, 2018. (IMC '18), p. 91–104. ISBN 9781450356190. Disponível em: <https://doi.org/10.1145/3278532.3278542>.

KONASHEVYCH, O. Emercoin blockchain anchoring as a way of singing contracts. In: **TERECOM@ JURIX**. [S.l.: s.n.], 2018. p. 17–29.

LEIDING, B.; CAP, C. H.; MUNDT, T.; RASHIDIBAJGAN, S. Authcoin: validation and authentication in decentralized networks. **arXiv preprint arXiv:1609.04955**, 2016.

MATSUMOTO, S.; REISCHUK, R. M. Ikp: Turning a pki around with decentralized automated incentives. In: IEEE. **2017 IEEE Symposium on Security and Privacy (SP)**. [S.l.], 2017. p. 410–426.

MIERS, C.; KOSLOVSKI, G.; PILLON, M.; JR, M. A. S.; UZH, B. B. R.; BATTISTI, J. H. Análise de mecanismos para consenso distribuído aplicados a blockchain. SBC, 2019.

NAKAMOTO, S. **Bitcoin: A peer-to-peer electronic cash system**. 2009. Disponível em: <http://www.bitcoin.org/bitcoin.pdf>.

NAKAMOTO, S.; BITCOIN, A. A peer-to-peer electronic cash system. **Bitcoin.–URL: https://bitcoin. org/bitcoin. pdf**, 2008.

PALLADINO, S. **Ethereum for Web Developers: Learn to Build Web Applications on Top of the Ethereum Blockchain**. [S.l.]: Apress, 2019.

QIN, B.; HUANG, J.; WANG, Q.; LUO, X.; LIANG, B.; SHI, W. Cecoin: A decentralized pki mitigating mitm attacks. **Future Generation Computer Systems**, Elsevier, v. 107, p. 805–815, 2020.

REBELLO, G. A. F.; CAMILO, G. F.; SILVA, L. G.; SOUZA, L. A. C. de; GUIMARÃES, L. C.; ALCHIERI, E. A.; GREVE, F.; DUARTE, O. C. M. Correntes de blocos: Algoritmos de consenso e implementação na plataforma hyperledger fabric. In: **38o Jornada de Atualização em Informática (JAI) do XXXIX Congresso da Sociedade Brasileira de Computação (CSBC 2019)**. [S.l.: s.n.], 2019.

SALMAN, T.; ZOLANVARI, M.; ERBAD, A.; JAIN, R.; SAMAKA, M. Security services using blockchains: A state of the art survey. **IEEE Communications Surveys & Tutorials**, IEEE, v. 21, n. 1, p. 858–880, 2018.

SCHMEH, K. **Cryptography and public key infrastructure on the Internet**. [S.l.]: John Wiley & Sons, 2006.

SHIREY, R. **RFC2828: Internet security glossary**. [S.l.]: RFC Editor, 2000.

SINGLA, A.; BERTINO, E. Blockchain-based pki solutions for iot. In: IEEE. **2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)**. [S.l.], 2018. p. 9–15.

SIVAKUMAR; SINGH; KUNWAR. Privacy based decentralized public key infrastructure (pki) implementation using smart contract in blockchain. **technical report**, 2017.

SOLOMON. **Ethereum for dummies**. [S.l.]: John Wiley & Sons, 2019.

SWAN, M. **Blockchain: Blueprint for a new economy**. [S.l.]: "O'Reilly Media, Inc.", 2015.

SZABO, N. Formalizing and securing relationships on public networks. **First Monday**, 1997.

SZYDLO, M. Merkle tree traversal in log space and time. In: SPRINGER. **International Conference on the Theory and Applications of Cryptographic Techniques**. [S.l.], 2004. p. 541–554.

TAPSCOTT, D.; TAPSCOTT, A. **Blockchain revolution: how the technology behind bitcoin is changing money, business, and the world**. [S.l.]: Penguin, 2016.

TELES, D. P. G. C. **Data Protection with Ethereum Blockchain**. Tese (Doutorado) — Instituto Superior de Engenharia do Porto, 2018.

WANG, J.; XIE, H. **Block generation method, device and blockchain network**. [S.l.]: Google Patents, jun. 6 2019. US Patent App. 16/314,635.

WANG, R.; HE, J.; LIU, C.; LI, Q.; TSAI, W.-T.; DENG, E. A privacy-aware pki system based on permissioned blockchains. In: IEEE. **2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS)**. [S.l.], 2018. p. 928–931.

WIDICK, L.; RANASINGHE, I.; DANTU, R.; JONNADA, S. Blockchain based authentication and authorization framework for remote collaboration systems. In: IEEE. **2019 IEEE 20th International Symposium on"A World of Wireless, Mobile and Multimedia Networks"(WoWMoM)**. [S.l.], 2019. p. 1–7.

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

WU, X.; ZOU, Z.; SONG, D. **Learn Ethereum: Build your own decentralized applications with Ethereum and smart contracts**. Packt Publishing, 2019. ISBN 9781789953572. Disponível em: <https://books.google.com.br/books?id=RFqxDwAAQBAJ>.

YAKUBOV, A.; SHBAIR, W.; WALLBOM, A.; SANDA, D. *et al.* A blockchain-based pki management framework. In: **The First IEEE/IFIP International Workshop on Managing and Managed by Blockchain (Man2Block) colocated with IEEE/IFIP NOMS 2018, Tapei, Tawain 23-27 April 2018**. [S.l.: s.n.], 2018.

ZHANG, L.; LEE, B.; YE, Y.; QIAO, Y. Ethereum transaction performance evaluation using test-nets. In: SPRINGER. **European Conference on Parallel Processing**. [S.l.], 2019. p. 179–190.

ZHANG, P.; BOULOS, M. N. K. Blockchain solutions for healthcare. In: **Precision Medicine for Investigators, Practitioners and Providers**. [S.l.]: Elsevier, 2020. p. 519–524.

Apêndice A - Glossário de Termos Técnicos

Análise	Exame detalhado de cada parte que compõe um todo, buscando compreender tudo aquilo que o caracteriza.
API	Em português, Interface de Programação de Aplicativos, baseada na web, que representa um conjunto de rotinas e padrões de programação para acesso a aplicativos de software ou plataformas.
Avaliação	Apreciação quantitativa e/ou qualitativa sobre dados relevantes do processo.
Back-end	Nicho da programação que trabalha com a lógica da aplicação, armazenamento e segurança de todos os dados gerados.
Bytecode	Código intermediário, gerado a partir de um código fonte, que ao ser interpretado por um programa específico (o interpretador) executa alguma tarefa (programa).
Captcha	Tipo de teste de resposta para garantir que a resposta não seja gerada por um computador, mas por um ser humano.
Chave Privada	Chave secreta do par de chaves criptográficas (a outra é uma chave pública) em um sistema de criptografia assimétrica. É mantida secreta pelo seu dono (detentor de um certificado digital) e usada para criar assinaturas digitais e para decifrar mensagens ou arquivos cifrados com a chave pública correspondente.

Chave Pública	Chave mantida pública (a outra é uma chave privada) em um sistema de criptografia assimétrica. É divulgada pelo seu dono e usada para verificar a assinatura digital criada com a chave privada correspondente. Dependendo do algoritmo, a chave pública também é usada para cifrar mensagens ou arquivos que possam, então, ser decifrados com a chave privada correspondente.
Cidades Inteligentes	Cidades que se utilizam de tecnologias e sensores para coletar dados sobre as interações entre pessoas-cidade-tecnologia para promover uma melhor utilização dos recursos e oferecer aos cidadãos mais mobilidade, custos de vida menores e mais qualidade de vida.
Criptografia Assimétrica	Tipo de criptografia que usa um par de chaves criptográficas distintas (privada e pública) e matematicamente relacionadas. A chave pública está disponível para todos que queiram cifrar informações para o dono da chave privada ou para verificação de uma assinatura digital criada com a chave privada correspondente; a chave privada é mantida em segredo pelo seu dono e pode decifrar informações ou gerar assinaturas digitais.
Criptomoeda	Meio de troca digital que usa criptografia para proteger transações financeiras, controlar a criação de unidades adicionais e verificar a transferência de ativos.
Cluster	Técnica utilizada para colocar diversos processos em paralelo para aumentar a capacidade de processamento .
DApp	Aplicação descentralizada. No mínimo, é um contrato inteligente e uma interface de usuário da web. De forma mais ampla, um Dapp é um aplicativo da web criado com base em serviços de infraestrutura ponto a ponto abertos, descentralizados. Além disso, muitos Dapps incluem armazenamento descentralizado e/ou um protocolo e plataforma de mensagem.
Dashboard	É a apresentação visual das informações mais importantes e necessárias para alcançar um ou mais objetivos de negócio, consolidadas e ajustadas em uma tela para fácil acompanhamento.

Datacenter	trata-se do local onde empresas abrigam os servidores responsáveis por processar dados ativos em uma rede.
***Distributed Ledger Technology* (DLT)**	Trata-se dos processos e das tecnologias relacionadas de uma moeda virtual distribuída (criptomoeda) que, de forma segura, proponha, valide e registre mudanças de estados em um registro distribuído sincronizado entre os nós da rede.
Eficiência	Poder de realizar algo, convenientemente, despendendo de um mínimo de esforço, tempo e outros recursos ou competências.
Experimento	Pesquisa planejada para obter novos fatos, para obter ou confirmar resultados, tendo por objetivo tomar decisões.
Front-end	Nicho da programação que trabalha com a experiência do usuário (UX) e com a interface (UI ou GUI), trabalhando principalmente com HTML, CSS e JavaScript.
Hash	Resultado da ação de algoritmos que fazem o mapeamento de uma sequência de bits de tamanho arbitrário parauma sequência de bits de tamanho fixo menor -conhecido como resultado hash - de forma que seja muito difícil encontrar duas mensagens produzindo o mesmo resultado hash (resistência à colisão) e que o processo reverso também não seja realizável (dado um hash, não é possível recuperar a mensagem que o gerou).
***Hardware Security Module* (HSM)**	Criptoprocessador dedicado, especificamente projetado para a proteção do ciclo de vida de uma chave criptográfica. Um HSM age como âncora segura que protege a infraestrutura criptográfica gerenciando, processando e armazenando chaves criptográficas em um ambiente seguro e resistente a adulterações.
***Internet of things* (IoT)**	Termo utilizado para a conexão de basicamente qualquer coisa à internet, seja ela um eletrodoméstico, dispositivo, tênis, enfim, qualquer coisa. a IoT tem tido grande impacto nas empresas, sendo aplicada em projetos e até mesmo na gestão da empresa.
Método	Conjunto de passos para se chegar a um objetivo, de forma repetível, obedecendo a um conjunto de regras.

Metodologia	Estudo do método.
Processo	Conjunto sequencial e particular de ações com objetivo.
Tecnologia Blockchain	É a estrutura blockchain aplicada à realidade.
Validação	É o ato de tornar ou declarar algo como válido.
Verificação	É o ato de examinar se algo é o que deve ser ou o que se declarou ser.

FOLHA DE REGISTRO DO DOCUMENTO

1. CLASSIFICAÇÃO/TIPO	2. DATA	3. REGISTRO N°	4. N° DE PÁGINAS
DM	05 de abril de 2021	DCTA/ITA/DM-017/2021	118

5. TÍTULO E SUBTÍTULO:

Uma abordagem comparativa entre ICPs baseadas em Blockchain.

6. AUTOR(ES):

Rogério Caldas dos Santos

7. INSTITUIÇÃO(ÕES)/ÓRGÃO(S) INTERNO(S)/DIVISÃO(ÕES):

Instituto Tecnológico de Aeronáutica - ITA

8. PALAVRAS-CHAVE SUGERIDAS PELO AUTOR:

Blockchain; ICP; Escalabilidade; Desempenho; Testbed

9.PALAVRAS-CHAVE RESULTANTES DE INDEXAÇÃO:

Certificação; Protocolo de confiança; Segurança de dados; Regressão linear; Computação.

10. APRESENTAÇÃO: **(X) Nacional ( ) Internacional**

ITA, São José dos Campos. Curso de Mestrado. Programa de Pós-Graduação em Engenharia Eletrônica e Computação. Área de Informática. Orientador: Prof. Cesar Augusto Cavalheiro Marcondes. Defesa em 02/03/2021. Publicada em 2021.

11. RESUMO:

Os sistemas de certificação digital têm um fundamental papel na segurança das informações. Neste contexto, aplicações de Infraestruturas de Chaves Públicas (ICPs) baseadas na tecnologia de *Blockchain* vêm sendo estudadas para melhor atender às novas demandas tecnológicas da sociedade e do estado moderno. Neste cenário, esta pesquisa envolveu uma campanha de análise de desempenho, com o intuito de compreender as ICPs baseadas na tecnologia de Cadeia de Blocos (*Blockchain*) de forma realista, verificando aspectos como escalabilidade e tempo de respostas de transações. Com este objetivo, foram realizados experimentos no ambiente de teste da moeda virtual *Ethereum*, utilizando a rede *Rinkeby*, por meio de comparação de contratos inteligentes publicamente reconhecidos, variando-se a carga para avaliar a escalabilidade e o tempo de resposta. Além disso, foi desenvolvido um ambiente de teste (*testbed*) para execução desses experimentos e realizada uma caracterização do ambiente experimental. Para obter melhor acurácia, em relação à comparação dos algoritmos, foram utilizados modelos de regressão linear, realizadas alterações nos contratos inteligentes, exploradas funções criptográficas mais intensas e elaborada uma proposta de melhoria do tempo de execução por distribuição das requisições em diferentes blocos de *Blockchain*. Os resultados mostraram diferenças importantes de desempenho entre os algoritmos testados e as aplicações de ICPs baseadas em *Blockchain*, em termos de escalabilidade, limitadas pelas quantidades máximas de transações por bloco da tecnologia de *Blockchain*.

12. GRAU DE SIGILO:

(X) OSTENSIVO ( ) RESERVADO ( ) SECRETO