Transações Autenticadas para Aplicativos Não Confiáveis em Blockchains Permissionadas

André Defrémont[1,2], Jeffson Celeiro Sousa[1,3], Billy Anderson Pinheiro[2], Roberto Samarone Araújo[1], Antônio Jorge G. Abelém[1]

[1]Programa de Pós-Graduação em Ciência da Computação.
Instituto de Ciências Exatas e Naturais – Universidade Federal do Pará (UFPA)
Belém – PA – Brasil

[2]Amazônia Blockchain Solutions (Amachains)
Parque de Ciência e Tecnologia do Guamá (PCT Guamá), Espaço Empreendedor
Belém – PA – Brasil

[3]Centro de Pesquisa e Desenvolvimento em Telecomunicações (CPQD)
Campinas – SP – Brasil

{andre, billy}@amachains.com, jcsousa@cpqd.com.br, {rsa, abelem}@ufpa.br

Abstract. *The use of external applications by organizations, that is, those that are not trusted, has become a common practice with the advancement of the internet. However, allowing data registration through untrusted third parties and ensuring non-repudiation can be a difficult task. To overcome these challenges, we propose a solution through the development of an architecture that guarantees authentication and non-repudiation of records on blockchain, even when inserted by other identities. The architecture consists of a service responsible for communication between third parties and the blockchain (Auth Service), methods for generating a unique transaction tokens, and protocols for authentication in smart contracts. The proposed architecture provides a secure and efficient process for sending transactions through untrusted applications, reducing the dependence on the Certificate Authority (CA) of the network and ensuring the non-repudiation of data. With our solution, users can access various network services with confidence.*

Resumo. *A utilização de aplicativos externos as organizações, ou seja, não confiáveis, tornou-se uma prática comum com o avanço da internet. No entanto, permitir o registro de dados através de terceiros não confiáveis e garantir o não repúdio, pode ser uma tarefa difícil. Para superar esses desafios, propomos uma solução por meio do desenvolvimento de uma arquitetura que garante a autenticação e o não repúdio dos registros em blockchain, mesmo quando inseridos por outras identidades. A arquitetura é composta por um serviço responsável pela comunicação entre terceiros e a blockchain (Auth Service), métodos para geração de tokens únicos de transações e protocolos para autenticação em contrato inteligente. A arquitetura proposta fornece um processo seguro e eficiente para o envio de transações por meio de aplicativos não confiáveis, reduzindo a dependência da Autoridade Certificadora (CA) da rede e garantindo o não repúdio dos dados. Com a nossa solução, os usuários podem acessar diversos serviços da rede com confiança.*

1. Introdução

Blockchain é uma tecnologia que já transforma diversas indústrias ao permitir transações confiáveis e transparentes sem a necessidade de intermediários. O termo *"blockchain"* refere-se a um livro razão descentralizado e distribuído que registra transações em blocos interligados e criptografados [Nakamoto 2009]. Essa estrutura proporciona aumento de segurança, responsabilidade e transparência, pois todos os participantes da rede têm acesso às mesmas informações e os registros não podem ser alterados retroativamente sem o consenso da rede.

Blockchain e *Distributed Ledger Technology (DLT)* compartilham muitas semelhanças que as tornam atraentes para vários casos de uso. Ambas as tecnologias são descentralizadas e dependem de uma rede de nós para manter e validar transações [Swan 2015]. Elas usam técnicas criptográficas para garantir a segurança dos dados e garantir a imutabilidade, o que significa que uma vez que uma transação é registrada, ela não pode ser alterada ou excluída. Além disso, ambas as tecnologias oferecem transparência, já que todos os participantes podem ver os mesmos dados, o que pode promover a confiança e reduzir o potencial de fraude. Embora existam diferenças entre *blockchain* e DLT, como o mecanismo de consenso usado para validar transações, os benefícios e características gerais dessas tecnologias as tornam adequadas para uma ampla gama de aplicações, desde transações financeiras até gestão da cadeia de suprimentos e além. Portanto, este trabalho usará o termo "*blockchain*"para se referir a essa tecnologia pois transmite a natureza e as capacidades dessa tecnologia inovadora.

A forma permissionada desta tecnologia, também conhecido como *blockchain* de consórcio, é um tipo de *blockchain* especificamente projetado para uso em ambientes privados de nível empresarial [Cachin et al. 2016]. Diferentemente de redes públicas de *blockchain*, como o *Bitcoin* ou *Ethereum*, o acesso a uma rede de *blockchain* permissionada é controlada por uma autoridade centralizada, como um consórcio de organizações. Isso permite um maior controle sobre quem tem acesso à rede e quais ações são capazes de realizar [Alghamdi and Javaid 2022]. Nessas redes, os participantes geralmente se conhecem e têm um objetivo em comum, entretanto, não há como garantir a integridade dos dados em transações enviadas por outras identidades, como em serviços ou aplicações da rede.

Neste contexto, garantir a integridade dos dados é crucial para os usuários da rede, sendo necessário que cada usuário conceda acesso a outras identidades. No entanto, para permitir que os usuários registrem dados através de múltiplos serviços que utilizam identidade própria para interagir com a rede, é necessário implementações mais robustas, em contratos inteligentes.

Os contratos inteligentes são contratos autoexecutáveis, com os termos do acordo entre as partes sendo diretamente escritos em código [Buterin et al.]. Eles são armazenados e replicados em *blockchain*, proporcionando aumento de segurança e transparência. Ao utilizar contratos inteligentes, a arquitetura proposta neste trabalho proporciona um processo de envio de transações autenticadas seguro e eficiente, com o uso de métodos de senha de uso único (OTP).

O conceito de *One-Time Password* (OTP) foi proposto pela primeira vez por [Lamport 1981] e tem sido usado para gerar senhas de uso único que mudam continu-

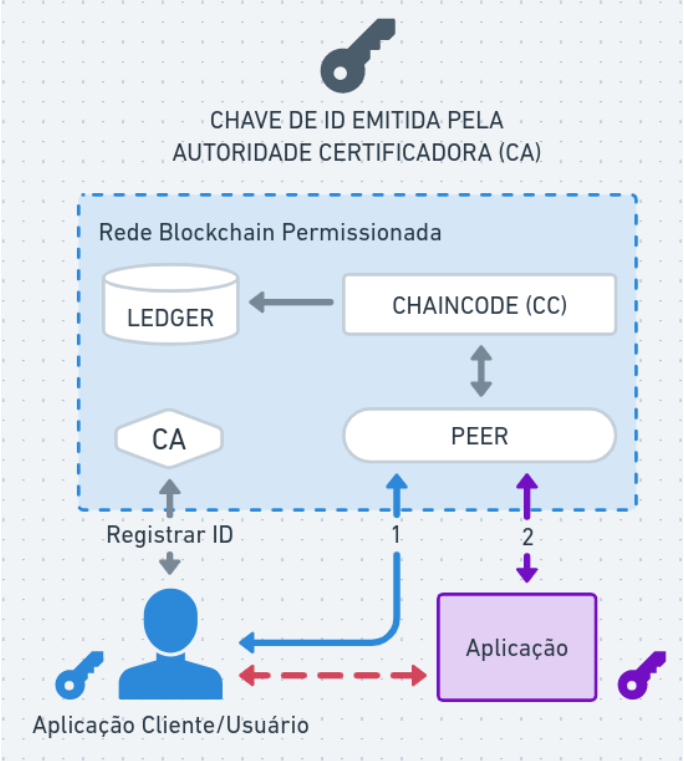

Figura 1. Arquitetura padrão em redes permissionadas (Autor)

amente ao longo do tempo ou uso. Os provedores de serviços não precisam participar da fase de registro e precisam apenas encaminhar o *Token* OTP (a senha de uso único enviada pelo provedor, ou seja, o cliente) para o serviço de autenticação. Sem dúvida, isso reduz a carga dos provedores de serviços para implantar soluções de autenticação e permite que os usuários acessem vários serviços com a mesma identidade. Neste sentido, este trabalho utiliza *Token* OTP para criar propostas de transações autenticadas, que podem ser enviadas através de identidades não confiáveis.

A Figura 1 mostra como a arquitetura padrão de uma rede *blockchain* permissionada funciona. Ao interagir com a *blockchain*, os usuários devem utilizar sua própria identidade, que é emitida pela autoridade de certificação (CA) da rede (1). Nesse caso, quaisquer aplicativos externos também devem ser confiáveis para garantir a integridade e o não repúdio das transações, já que as mesmas precisam manipular a identidade e os dados das transações do usuário. Isso não só gera preocupações de segurança, como restringe o desenvolvimento de aplicativos para apenas organizações confiáveis dentro da rede. Existe, no entanto, outra maneira: os aplicativos podem usar sua própria identidade (2), emitida pela CA da rede, para conduzir transações em nome do usuário. Embora essa abordagem possa resolver alguns dos desafios, ainda são necessários métodos para garantir o não repúdio dos dados, já que a informação foi registrada por outra identidade. Neste contexto, este trabalho explora os desafios do envio de transações através da identidade de terceiros e propõe uma nova abordagem para supera-los.

A implementação de protocolos de autorização, métodos OTP e gestão de chaves públicas em contratos inteligentes elimina a necessidade de confiança em provedores de identidade e habilita o registro de dados por terceiros não confiáveis de forma segura.

Este estudo destaca o potencial da tecnologia blockchain permissionada para enfrentar os desafios relacionados à autenticação de terceiros para transações, fornecendo dados relacionados a comparação de segurança e eficiência de outros trabalhos e experimentos, para pesquisas futuras nessa área.

2. Trabalhos Relacionados

Um dos primeiros trabalhos nesta área é o de [Park et al. 2018]. Ele propõe um esquema de autenticação de dois fatores baseado em senha única (TOTP) para Hyperledger Fabric. A proposta concentra-se exclusivamente no problema de adquirir dados sensíveis através do Jason Web Token (JWT) encontrado a primeira versão do Fabric e não aborda outras preocupações de segurança como o ataques ao servidor TOTP. Além disso, o esquema proposto suporta apenas o Fabric 1.0 e depende da confiança na Autoridade de Certificação (CA). No geral, embora este trabalho apresente uma abordagem inovadora para autenticação na blockchain, ele as limitações significativas como o servidor TOTP que preserva o problema de centralização e pode não ser adequado para casos de uso mais complexos como o envio de transações através de identidades de terceiros.

Em [Lu et al. 2018], os autores propõem um sistema de autenticação/autorização/contabilidade interorganizacional que preserva a privacidade usando a tecnologia blockchain. Eles apresentam um sistema projetado para uma blockchain pública e armazena senhas codificadas na blockchain. No entanto, o esquema de autorização proposto é apenas para regras de acesso do usuário sobre as funcionalidades da rede e não sobre identidades de terceiros em relação os dados do usuário, limitando sua aplicabilidade a outros cenários.

O artigo de [Zhang et al. 2019] descreve um método de autenticação baseado em blockchain com uma senha única. Embora o método proposto seja promissor, o artigo apenas descreve o método de autenticação com OTP e blockchain permissionada, sem abordar o envio de transações seguras a partir de outras identidades.

Finalmente, o artigo de [Catalfamo et al. 2021] propõe um protocolo baseado em microserviços e blockchain para autenticação aprimorada de senha única (MBB-OTP). Além de se restringir apenas a autorização dos usuários da rede, o uso de segredos compartilhados e múltiplos microserviços possibilitam um maior custo computacional e vulnerabilidades em comparação a proposta deste trabalho. Além disso, ele requer mais tempo para concluir apenas o processo de autenticação.

A Tabela 1 mostra que todos estes trabalhos apresentam várias abordagens para autenticação e autorização na *blockchain*. Embora cada um deles tenham suas vantagens e limitações, nenhum deles fornece uma solução para o não repúdio de dados inseridos por outras identidades.

A arquitetura proposta neste trabalho visa preencher essa lacuna, oferecendo uma solução mais abrangente para transações autenticadas em redes do tipo *blockchain* permissionada. Além de ser mais eficiente e menos dependente da CA da rede, ela elimina a necessidade de confiança ao enviar transações através de outras identidades, garantindo maior privacidade e segurança aos usuários. Acreditamos que essa abordagem possa contribuir significativamente para aprimorar a segurança e a eficiência de métodos de autenticação em *blockchain* em diversos cenários, trazendo benefícios para usuários e empresas que lidam com essa tecnologia.

	Proposta	Park et al.	Lu et al.	Zhang et al.	Catalfamo et al.
Autenticação com Tokens	x	x		x	x
Nenhum terceiro totalmente confiável é necessário	x		x	x	
Não repúdio de dados inseridos através de outras identidades	x				
Envio de transações sem identidade emitida pela CA	x				x
Histórico (logs)	x		x	x	x
Impossibilidade de falsificação de Tokens	x	x		x	x
Resistência a ataques de reutilização	x			x	
Blockchain abordada	Fabric v2.x	Fabric v1.x	Ethereum	Fabric v2.x	Fabric v2.x / Ethereum

Tabela 1. Comparação entre a proposta e os trabalhos relacionados (Autor)

3. Transações Autenticadas para Aplicativos Não Confiáveis em Blockchains Permissionadas

Esta seção descreve a arquitetura proposta para autenticação de identidades não confiáveis para transações em uma rede de *blockchain* permissionada, utilizando o framework Hyperledger Fabric. A proposta inclui um modelo adversário e critérios específicos de segurança para resistir a ataques na rede e foca nos desafios de autenticação para envio de transações em aplicativos externos. O artigo descreve os quatro módulos da arquitetura: a rede de *blockchain*, o *Auth Service*, o cliente ou usuário e o aplicativo servidor. Além disso, apresenta o fluxo, incluindo a geração e verificação de transações autenticadas. No geral, essa proposta tem como objetivo fornecer uma solução para o envio de transações através de identidades não confiáveis, de forma segura e descentralizada que pode ser facilmente integrada em aplicativos.

3.1. Modelo Adversário e Limitações

A arquitetura aqui proposta considera que um adversário tem acesso à rede *blockchain* e pode obter informações armazenadas, incluindo valores de *tokens* usados ou não. No entanto, o adversário não pode comprometer ou obter a chave privada armazenada com segurança no cliente. Para garantir a integridade da mensagem e evitar ataques *man-in-the-middle*, as comunicações são protegidas por SSL/TLS [Oppliger 2016].

O adversário não pode modificar o valor de *tokens* armazenados na *blockchain*, mas seu objetivo é gerar ou reutilizar *tokens* que são aceitos pelos nós da *blockchain* ou modificar o valor de um *token* para forjar o próximo.

Os critérios de segurança propostos incluem resistência a ataques de reutilização de *tokens*, resistência a ataques de força bruta, inviabilidade de *tokens*, menor confiança no provedor de identidade, registro à prova de adulteração e garantia de não repúdio. Esses critérios serão usados para avaliar a solução proposta e medir sua eficácia contra possíveis ataques do adversário.

Este trabalho não aborda implementações em *blockchain* pública, não faz uso de chaves privadas geradas pela CA da rede para verificar ou gerar *tokens* e não explora a escalabilidade da solução em termos de comunicação com diversos contratos inteligentes sem a necessidade de registro a cada novo contrato. A falta de consideração desses pontos pode limitar a descentralização, segurança e escalabilidade da solução proposta.

3.2. Arquitetura

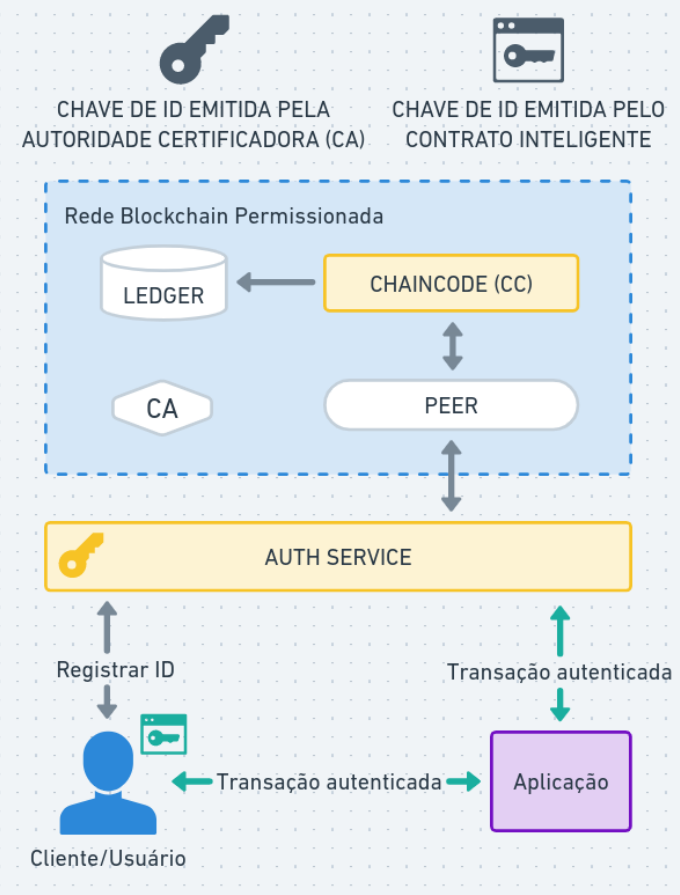

Figura 2. Arquitetura da proposta (Autor)

A arquitetura é dividida em quatro módulos, apresentados na Figura 2: a Rede Blockchain Permissionada, que consiste em *Peers, CAs, Ledger e Chaincode (CC); o Auth Service*, que é responsável por repassar os *tokens* para a *blockchain*; o Cliente ou Usuário, que gera os *tokens* para efetuar seu registro e transações autenticadas e por fim, o Aplicativo que fornece serviços da rede para o Cliente/Usuário, responsável por receber os *tokens* das transações autenticadas. Todo contrato inteligente da rede precisa de configurações adicionais para corresponder aos protocolos descritos e o serviço

de autenticação precisa ser desenvolvido e integrado a rede. Os *tokens* são *hashs* criptográficos que contém dados do usuário como id, senha, chave privada e um índice incremental a cada requisição bem sucedida.

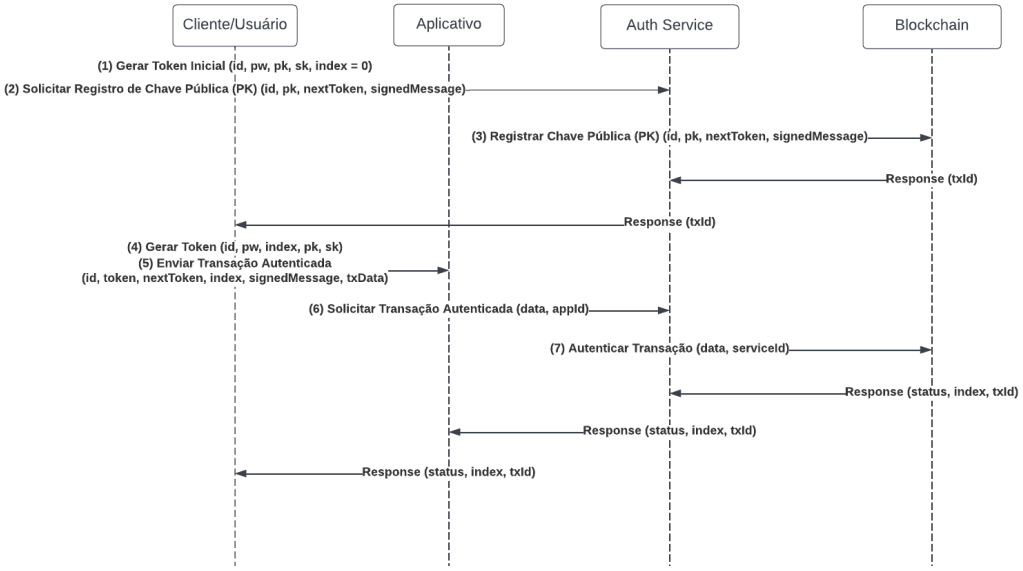

Figura 3. O diagrama de sequência para autenticação da proposta (Autor)

O diagrama de sequência apresentado na Figura 3 inicia-se com o Cliente/Usuário gerando um *token* inicial para efetuar o registro de sua chave pública em *blockchain* e para efetuar as transações autenticadas. Para isso, o cliente utiliza uma função para geração de *token* inicial passando como parâmetros seu par de chaves criptográficas (pk, sk), um valor para seu id e senha (pw); além do valor igual a zero para o índice (index) inicial (1). Com o *token* inicial em mãos, o Cliente/Usuário deve gerar o próximo *token*, apenas incrementando o índice em uma unidade e convocando a mesma função para geração de *token*. Por fim, o Cliente/Usuário gera uma mensagem assinada com a sua chave privada, contendo seu id, chave pública e o próximo *token* gerado. Com esses dados, o Cliente/Usuário pode solicitar ao *Auth Service* o registro de sua chave pública em *blockchain* enviando apenas seu id, chave pública, próximo *token* e a mensagem assinada (2). Na transação de registro de chave pública do contrato inteligente, é verificado se o id solicitado já possui chave pública vinculada e se a mensagem assinada é válida, para registrar o id do Cliente/Usuário, a chave pública do Cliente/Usuário, o próximo *token* e o índice com valor um; em *blockchain* (3).

Seguindo o diagrama de sequência da Figura 3, após o processo de registro de chave pública (3) o Cliente/Usuário pode começar a enviar transações autenticadas. Para isso, é necessário que ele faça um processo para gerar um novo *token* e o próximo *token* de forma similar a anterior, incrementando o índice em uma unidade (4). Com os *tokens* prontos, o Cliente/Usuário informa os dados da transação (txData), o id do *Auth Service*(serviceId) e o id do Aplicativo (appId) para gerar uma nova mensagem assinada. O Cliente/Usuário envia para o Aplicativo seu id, o novo *token*, o próximo *token*, o índice atual, a mensagem assinada, os dados da transação (5) para que o Aplicativo solicite a transação, ao *Auth Service* (6). Embora o Aplicativo possa manipular os da-

dos da transação informados de forma explícita, esses valores também são incluídos na mensagem assinada, por tanto, são verificadas no contrato inteligente para garantir sua integridade. O valor de id do *Auth Service* serve para evitar ataques de repetição (*Replay Attacks*), onde o Cliente/Usuário assina uma mensagem com o id da identidade do *Auth Service* e o id do Aplicativo corretos e o *Auth Service*, se encarrega de identificar os Aplicativos a cada requisição de transação, enquanto o contrato inteligente identifica e valida a identidade que chamou a transação, com o id do serviço na mensagem assinada (7).

Para autenticar uma transação, o *Auth Service* invoca a transação Authenticate do contrato inteligente para verificar se existe um registro de id do Usuário/Cliente em *blockchain*, se o índice enviado está correto, se o próximo *token* (nextToken) enviado é igual ao hash do *token* enviado, resgatar o id da identidade do *Auth Service* para verificar a mensagem assinada. Se passar pelas verificações, o contrato atualiza os registros do usuário de índice e próximo *token*; e finalmente, registra a transação solicitada.

4. Implementação

A rede *blockchain* da proposta foi implementada utilizando o *framework Hyperledger Fabric* [Cachin et al. 2016]. A arquitetura inclui um micro-serviço, chamado *Auth Service*, que pode ser executado de forma autônoma e possui uma identidade emitida pela CA da rede para se comunicar com a *blockchain* e repassar as solicitações. Também é necessário a implementação de funções ao lado do usuário para geração de *tokens*, par de chaves criptográficas e mensagens assinadas. Para implementar a arquitetura proposta, os contratos inteligentes da rede são modificados para incluir os protocolos de autenticação e os registros das chaves públicas.

4.1. Cliente/Usuário

A arquitetura proposta utiliza um esquema de autenticação de senha única baseado no trabalho de [Park 2018], utilizando a função de hash SHA256 e o algoritmo de assinatura RSA-SHA256 para gerar e verificar os *tokens*. A implementação da arquitetura foi feita em *JavaScript*, utilizando o ambiente de execução *Node.js* e os pacotes *crypto* e *keypair*.

```
function generateToken(
  id,
  password,
  privateKey,
  publicKey,
  index,
  txData,
  appId,
  serviceId
) {
  if (index === 0) {
    let token = hash({ id, index, password, privateKey });
    let nextToken = hash(token);
    let message = { id, publicKey, nextToken };
    let signedMessage = sign(privateKey, message);
    let result = { id, token, nextToken, index, signedMessage };
    return result;
  } else {
    let newToken = hash({ id, index, password, privateKey });
    let nextToken = hash(newToken);
```

```
21      let message = { id, publicKey, nextToken, txData, appId, serviceId
        };
22      let signedMessage = sign(privateKey, message);
23      let previousToken = hash({ id, index: index - 1, password,
        privateKey });
24      let result = { id, previousToken, nextToken, index, signedMessage
        };
25      return result;
26    }
27  }
```

Listagem de Código 1. Algoritmo da Função GenerateToken do Cliente/Usuário (Autor)

A implementação na parte do cliente da arquitetura, consiste em quatro funções principais: *hash*, *sign* e *generateToken*, sendo esta última responsável por gerar os *tokens* para o usuário. A função *hash* cria um *hash SHA256* da *string* de entrada, a função *sign* gera uma assinatura da mensagem utilizando a chave privada fornecida. A Listagem de Código 1 mostra o algoritmo da função generateToken, que se receber um índice (index) igual a zero, gera o *token* inicial e o próximo utilizando o *hash SHA256* [Dang 2015] dos dados sensíveis do usuário (*id, password, index, privateKey*) e o algoritmo de assinatura RSA-SHA256 [Unal et al. 2021] para gerar a mensagem assinada com a chave privada e os dados públicos do usuário (id, publicKey, nextToken). Os próximos *tokens* são equivalentes ao *hash SHA256* do *token* anterior, incrementado um ao índice (index + 1) e a mensagem assinada deve conter, além dos dados públicos do usuário, o id do Aplicativo (appId), o id do *Auth Service* (serviceId) e os dados da transação solicitada.

O processo de autenticação das transações envolve o uso da função *generateToken* para gerar os próximos *tokens* através da inserção dos valores de índices maiores que zero para que o usuário possa reunir os dados necessários com os *tokens* gerados e repassar ao Aplicativo e posteriormente para o *Auth Service*, para que este último faça a verificação em blockchain dos *tokens* e assinaturas, a qual será descrita na próxima seção.

No geral, a implementação do esquema de *tokens* fornece um método simples e seguro de autenticação que pode ser facilmente integrado em aplicativos web ou móveis. Com a ajuda deste esquema, os usuários podem gerar seus próprios *tokens* sem depender de um terceiro confiável para gerar ou armazenar senhas. Além disso, o uso de *hash* e assinaturas criptográficas fornece um alto nível de segurança para o processo de envio de transações.

4.2. Chaincodes

Os contratos inteligentes da rede devem incluir funções que chamam as demais transações do contrato para garantir transações autenticadas, isso envolve definir métodos que verifiquem os *tokens* enviados, antes de enviar cada transação do contrato.

```
1  function register(id, publicKey, nextToken, signedMessage) {
2    const message = {
3      id: id,
4      publicKey,
5      nextToken,
6    };
7    const userData = getState(id);
```

```
  if (userData) {
    throw new Error("Already registered");
  }
  if (!verifySignedMessage(publicKey, signedMessage, message)) {
    throw new Error("Signature verification invalid");
  }
  putState(id, {
    publicKey,
    nextToken,
    index: 1,
  });
}
```

Listagem de Código 2. Algoritmo da Função Register do Contrato Inteligente (Autor)

O contrato inteligente deve ser personalizado de forma que qualquer identidade pode chamar as duas funções principais: *authenticate* e *register*. A função *register*, mostrada na Listagem de Código 2, é utilizada para registrar novos usuários na rede *blockchain*. Ela recebe como parâmetros o id do usuário, chave pública, o próximo *token* e a mensagem assinada. A função verifica se é um novo usuário e se a mensagem assinada pelo usuário é válida. Se a verificação for bem-sucedida, o usuário é registrado na rede *blockchain*.

```
function authenticate(
  index,
  id,
  publicKey,
  token,
  nextToken,
  txData,
  appId,
  signedMessage
) {
  const serviceId = ClientIdentity().getID()
  const userData = getState(id);
  if (!userData) {
    throw new Error(`ID not found: ${id}`);
  }
  if (userData.index !== index) {
    throw new Error(`Index not match: ${userData.index}`);
  }
  const nextToken = hash(token);
  if (userData.nextToken !== nextToken) {
    throw new Error("Next Token not match");
  }
  const message = {
    id,
    publicKey,
    nextToken: userData.nextToken,
    txData,
    appId,
    serviceId,
  };
  if (!verifySignedMessage(userData.publicKey, signedMessage, message))
    {
```

```
32      throw new Error("Signature verification invalid");
33    }
34    userData.index++;
35    userData.nextToken = nextToken;
36    putState(userData.id, userData);
37    putState(txData.id, txData);
38  }
```

Listagem de Código 3. Algoritmo da Função Authenticate do Contrato Inteligente (Autor)

A função *authenticate*, mostrada na Listagem de Código 3, recebe como parâmetros os dados da transação solicitada (txData), id do usuário (id), chave pública (publicKey), os *tokens* (token, nextToken), o id do Aplicativo (appId) e uma mensagem assinada (signedMessage). A função verifica se o *token* é válido, se o id de usuário existe, se o id do serviço (serviceId) é o que está sendo chamado e se a chave pública do usuário corresponde à mensagem assinada. Se a verificação for bem-sucedida, os dados do usuário em *blockchain* é atualizado incrementando o índice e atribuindo o próximo *token* informado pelo usuário. Caso a verificação falhar, o usuário recebe o índice atual da sua autenticação para sincronizar seus valores de *token*, caso necessário.

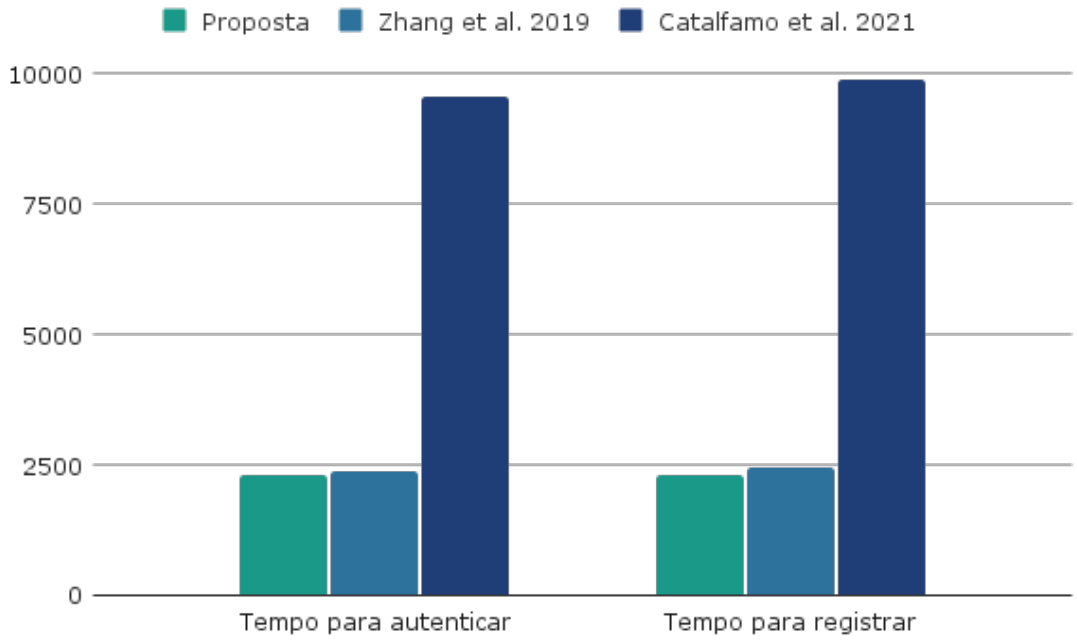

Figura 4. Tempo de execução por função dos trabalhos relacionados (Autor)

5. Avaliação

A partir da implementação da arquitetura proposta, foi possível testar as funções de geração de chaves e *tokens* com segurança, sem repassar informações a qualquer servidor e utilizá-las para enviar transações em contrato inteligente da rede através de serviços não confiáveis, com segurança.

A solução foi posta em comparação com outros trabalhos e experimentos usando um servidor local com os seguintes recursos: *Intel® Core™ i5-7300HQ CPU @ 2,50GHz × 4, 16 GB* de *RAM* rodando *Ubuntu Server* 22.04. Os testes foram realizados através da CLI e também foi construído um aplicativo cliente confiável em *Vue 3* que faz o papel de cliente e aplicativo nas funções de registro e autenticação, respectivamente.

O fluxo completo, desde a geração dos valores iniciais para registro de chave pública até o envio de uma transação no contrato inteligente principal, levou em média 4,8 segundos para ser executado. Além do tempo de criação de bloco de 2 segundos para esta rede, os consumos específicos para geração de chaves públicas e privadas, geração de token inicial e geração de token para transação, consumiram 2.474, 105, 104, milissegundos respectivamente.

Em relação a comparação dos tempos de processamento total das transações com transações sem o esquema de autenticação, sendo enviadas diretamente pelo usuário, foi possível constatar que há uma pequena perda de desempenho, cerca de 100 ms. A Figura 4 mostra a comparação dos tempos de processamento entre esquemas para geração de *tokens* considerando dos trabalhos [Zhang et al. 2019] e [Catalfamo et al. 2021] que forneceram dados de tempo de execução total.

Em relação a [Zhang et al. 2019], os resultados podem ser considerados similares para o tempo total de execução, mesmo a proposta deste trabalho apresentando um leve ganho, entretanto em [Zhang et al. 2019] considera esse tempo total apenas para executar uma autenticação, sem apresentar a integração com o envio de transações como na proposta deste trabalho. diferente de [Catalfamo et al. 2021] que leva até 10 segundos para o tempo total apenas para gerar e receber o *token*, entende-se a quantidade de micro-serviços necessários para [Catalfamo et al. 2021] como uma perda tanto de desempenho, como segurança. Apesar de não ter encontrado trabalhos com a mesma proposta de transações autenticadas utilizando *tokens*, Os resultados apresentados na Figura 4 indicam que o tempo de execução desta proposta é altamente competitivo em relação a outros trabalhos e experimentos realizados. Em comparação com esses cenários, que podem levar até 10 segundos apenas para concluir o processo de autenticação, sem incluir a transação principal, a proposta deste trabalho demonstrou um desempenho significativamente melhor.

A Tabela 1 também destaca as propriedades de segurança apresentadas no trabalho, em comparação com outras propostas e métodos tradicionais. Por exemplo, o trabalho apresenta uma redução na necessidade de confiança na CA da rede e a integração de chaves únicas para o envio de transações por terceiros não confiáveis, como outras aplicações servidoras. Isso torna o processo para registrar transações através de terceiros mais seguro e simplificado, garantindo a não repudiação dos dados registrados na blockchain.

No entanto, para uma avaliação mais abrangente, é importante considerar algumas limitações deste trabalho. Por exemplo, a arquitetura desenvolvida foi testada em um ambiente controlado e pode haver desafios ao implantá-la em um ambiente real. Além disso, o desempenho da solução pode variar dependendo do *hardware* e do tempo de resposta da rede. Essas limitações podem ser abordadas em futuros trabalhos.

6. Conclusão

Este trabalho apresentou uma arquitetura segura e eficiente para envio de transações por meio de identidades e aplicações externas a uma rede *blockchain* permissionada, enfren-

tando os desafios de garantir segurança e controle sobre o acesso à rede. A arquitetura proposta inclui um serviço de autenticação, protocolos de autenticação em contratos inteligentes e um métodos de geração de *tokens* únicos e seguros. Com esses componentes, arquitetura proposta fornece um processo de autenticação de transações eficiente sem depender de terceiros confiáveis, reduzindo a necessidade de confiança na CA da rede e habilitando o envio de transações através de identidades de terceiros.

As principais contribuições deste artigo incluem a arquitetura proposta, a comparação com outras abordagens e experimentos, bem como uma lista de critérios de segurança para avaliar o protocolo de autenticação proposto. A implementação da arquitetura fornece um método simples e seguro para enviar transações autenticadas, que pode ser facilmente integrado em aplicativos web ou móveis.

Embora o trabalho tenha algumas limitações e apresente desafios ao ser implantado em um ambiente real, a arquitetura proposta apresenta uma solução abrangente para o problema proposto. Pesquisas futuras devem abordar as limitações dessas abordagens em relação à comunicação entre os diferentes contratos inteligentes, desenvolver esquemas de autenticação mais robustos e seguros para sistemas baseados em *blockchain*, como incluir chaves geradas pela CA da rede no fluxo de endosso dos dados e garantir a segurança dos dados da transação, já que estas são visíveis durante todo o processo; além de testar em ambiente mais performáticos afim reduzir o tempo de execução da proposta.

7. Agradecimentos

Este trabalho foi realizado com o apoio do Grupo de Pesquisa em Redes de Computadores (GERCOM-UFPA), Amazonia Blockchain Solutions (Amachains), Conselho Nacional de Desenvolvimento Cientifico e Tecnologico (CNPq) e a Fundacao de Apoio a Pesquisa do Estado de Sao Paulo (FAPESP) - Processo No. 2018/23097-3.

Referências

Alghamdi, T. and Javaid, N. (2022). A comprehensive survey on security, privacy and authentication in blockchain.

Buterin, V. et al. A next-generation smart contract and decentralized application platform.

Cachin, C. et al. (2016). Architecture of the hyperledger blockchain fabric. In *Workshop on distributed cryptocurrencies and consensus ledgers*, volume 310, pages 1–4. Chicago, IL.

Catalfamo, A., Ruggeri, A., Celesti, A., Fazio, M., and Villari, M. (2021). A microservices and blockchain based one time password (mbb-otp) protocol for security-enhanced authentication. In *2021 IEEE Symposium on Computers and Communications (ISCC)*.

Dang, Q. H. (2015). Secure hash standard.

Lamport, L. (1981). Password authentication with insecure communication.

Lu, P. J., Yeh, L.-Y., and Huang, J.-L. (2018). An privacy-preserving cross-organizational authentication/authorization/accounting system using blockchain technology. In *2018 IEEE International Conference on Communications (ICC)*.

Nakamoto, S. (2009). Bitcoin: A peer-to-peer electronic cash system.

Oppliger, R. (2016). *SSL and TLS: Theory and Practice*. Artech House.

Park, C.-S. (2018). One-time password based on hash chain without shared secret and re-registration. *Computers and Security*.

Park, W.-S., Hwang, D.-Y., and Kim, K.-H. (2018). A totp-based two factor authentication scheme for hyperledger fabric blockchain. In *2018 Tenth International Conference on Ubiquitous and Future Networks (ICUFN)*.

Swan, M. (2015). *Blockchain: Blueprint for a new economy*. "O'Reilly Media, Inc.".

Unal, D., Al-Ali, A., Catak, F. O., and Hammoudeh, M. (2021). A secure and efficient internet of things cloud encryption scheme with forensics investigation compatibility based on identity-based encryption. *Future Generation Computer Systems*.

Zhang, M., Wang, L., and Yang, J. (2019). A blockchain-based authentication method with one-time password. In *2019 IEEE 38th International Performance Computing and Communications Conference (IPCCC)*.