Revista Brasileira de Computação Aplicada, Novembro, 2024

DOI: 10.5335/rbca.v16i3.15634
Vol. 16, Nº 3, pp. 116–123
Homepage: seer.upf.br/index.php/rbca/index

RBCA
ISSN 2176-6649

ARTIGO ORIGINAL

R2SC: sistema RBAC baseado em contrato inteligente

R2SC: RBAC system smart contract-based

Lucas Vargas Dias [1], Tiago Antonio Rizzetti[1], Wagner Brignol[2], Luciane Neves Canha[1]

[1]Universidade Federal de Santa Maria, [2]Instituto Federal de Educação, Ciência e Tecnologia Sul-rio-grandense

*lucas_dias@redes.ufsm.br; rizzetti@redes.ufsm.br;brignol@ifsul.pelotas.edu.br;lucianecanha@ufsm.br

Recebido: 04/03/2024. **Revisado:** 12/11/2024. **Aceito:** 30/11/2024.

Resumo

Devido a popularidade da tecnologia Blockchain, existem uma diversidades de aplicações em diferentes contextos. Essa é uma tecnologia que permite uma comunicações confiável entre entidades não-confiáveis sem uma terceira parte centralizada. Além disso, a Blockchain é uma sequência cronológica de blocos ligados através do hash de cada um deles. Essa característica garante a integridade e imutabilidade dos dados. Adicionalmente, a Blockchain é uma grande tecnologia para armazenamento de dados como regras de controle de acesso. Este trabalho utiliza a Blockchain para fornecer um framework aplicando o conceito de Contrato Inteligente para um sistema de controle de acesso baseado em papéis (RBAC). Esse é o Sistema RBAC baseado em Contratos Inteligentes (R2SC). Também, o trabalho usa o framework Truffle para testar as funcionalidades do Contrato Inteligente. Em contrapartida aos trabalhos relacionados, o R2SC apresenta escalabilidade.

Palavras-Chave: Blockchain; Contratos Inteligentes; Controle de Acesso.

Abstract

Due to the popularity of Blockchain technology, there have been many applications of it in many contexts. It is a technology that allows trusted communication between untrusted entities without a centralized third party. Also, Blockchain is a chronological sequence of blocks linked through the hash of each one. This characteristic ensures data integrity and immutability. Therefore, Blockchain is a powerful technology for recording data like access control rules. This work utilizes Blockchain to provide a framework applying the Smart Contract concept for a Role-Based Access Control (RBAC) system. It is the RBAC System Smart Contract-based (R2SC). Also, it uses the framework Truffle to test the Smart Contract functionalities. In counterpart to the related works, R2SC has scalability.

Keywords: Access Control; Blockchain; Smart Contracts.

1 Introdução

Em uma diversidade de aplicações como medição inteligente (Dias and Rizzetti, 2021), atualização de eventos e alarmes de subestação (Quincozes et al., 2021), transação energética de veículo para veículo (V2V), veículo para rede (V2G), entre outros (Bertineti et al., 2020), os dados utilizados são considerados sensíveis e a comunicação entre as entidades participantes deve ser confiável e segura (Kim et al., 2019). Para esse fim, a Internet utiliza o *Transport Layer Security* (TLS). Ele é baseado na Infraestrutura de Chave Pública (ICP), em que as entidades comunicantes têm um certificado digital válido emitido por uma autoridade certificadora (CA). Porém, o TLS apresenta ponto-único de falha na CA como ocorre na proposta de Dias et al. (2021) em que a ICP gerencia os nós de uma rede *Distributed Hash Table* (DHT).

Uma forma de contornar esse problema é a utilização

de tecnologias distribuídas e uma delas que ganhou destaque nos últimos anos é a Blockchain. Ela é uma rede *peer-to-peer* (P2P) que permite a comunicação confiável entre entidades que não se confiam. Para isso, técnicas como assinatura digital, hash criptográfico e algoritmos de consenso são utilizadas (Aggarwal et al., 2020).

Adicionalmente, a Blockchain possui alta disponibilidade dos dados e uma vez que uma informação é registrada na Blockchain, ela é imutável. Isso permite a execução de contratos inteligentes de maneira confiável (Zheng et al., 2020).

Nesse sentido, este trabalho apresenta uma proposta de uso de contratos inteligentes para prover controle de acesso na interação entre dispositivos. A proposta procura agregar características da rede como rastreabilidade, eliminação de ponto-único de falha e imutabilidade. Vale ressaltar que o controle de acesso é baseado no modelo *Role-Based Access Control* (RBAC) (Ghafoorian et al., 2018) e o sistema proposto é denominado ***R**BAC **S**ystem **S**mart **C**ontract-based* (R2SC). Ele é estruturado da seguinte forma: a Seção 2 apresenta o referencial teórico sobre os conceitos abordas no trabalho. Em sequência, os trabalhos relacionados são abordados na Seção 3. Após, a arquitetura proposta é descrita na Seção 4. Os resultados e discussões são tratados na Seção 5. Por fim, a Seção 6 faz a conclusão do trabalho e aponta possíveis trabalhos futuros.

2 Referencial Teórico

2.1 Blockchain

A Blockchain é um tipo de *Distributed Ledger Technology* (DLT) (Miglani et al., 2020). Ela tem esse nome porque cada bloco é interligado através do seu hash, formando uma cadeia de blocos em sequência. Por outro lado, existem DLTs em formato de grafo acíclico direcionado (DAG). Vale ressaltar que cada entidade, na Blockchain, possui um par de chaves assimétrica. A chave pública é aplicada em hash para endereçamento do nó e a chave privada permite que ele assine digitalmente as transações (Shi et al., 2020).

A rede utiliza uma arquitetura descentralizada com topologia de comunicação *peer-to-peer* (P2P). Para alocação de uma informação na rede, os nós precisam entrar em acordo sobre o novo bloco (Ante, 2021). Para isso, algum mecanismo de consenso deve ser utilizado. Alguns exemplos de algoritmos de consenso utilizados em Blockchain são o *Practical Byzantine Fault Tolerance* (PBFT), *Proof-of-Stake* (PoS), *Proof-of-Work* (PoW) e outros (Xiao et al., 2020).

As transações são armazenados em blocos. Cada bloco possui um carimbo de tempo, *hash* do bloco anterior, assinatura digital de quem o emitiu, uma árvore de *Merkle* das transações ou dados dentro do bloco, um *nonce* e o *hash* do bloco (Viriyasitavat and Hoonsopon, 2019).

A interligação entre os blocos com o uso do *hash* do bloco anterior cria uma cadeia. Essa cadeia se torna difícil a modificação de um bloco já alocado. Para isso, todos os blocos posteriores teriam de ter seu *hash* recalculado. O que torna impraticável a modificação de blocos já alocados (Bach et al., 2018). Isso permite auditoria das comunicações e imutabilidade destas.

2.2 Contratos Inteligentes

Os contratos inteligentes formalizam e garantem as interações de computadores usando protocolos e interfaces bem definidas (Szabo, 1997). Por exemplo, uma máquina de venda automática de alimentos recebe como entrada o valor de R$5,00 e o cliente escolhe um produto de R$2,00, então, a máquina devolve o troco e o produto para o cliente.

No contexto de Blockchain, os contratos inteligentes também são um programa de computador com lógica bem definida. Ele fornece execução automática e facilita transações na Blockchain. Adicionalmente, eles permitem a transferência de bens na Blockchain e são programas descentralizados operando sobre a rede (Hewa et al., 2021). Outra característica dos contratos inteligentes são a imutabilidade e sua verificação de maneira criptográfica para garantir sua confiança (Wang and Su, 2020).

Sendo assim, este trabalho utiliza dos contratos inteligentes para propor um sistema de controle de acesso RBAC. Em sequência, na Seção 3, os trabalhos relacionados são apresentados.

3 Trabalhos Relacionados

A proposta de Lin et al. (2018) tem por objetivo prover privacidade dos dados, autenticidade entre as partes comunicantes, auditoria e confidencialidade voltada a indústria 4.0. O sistema é composto por validadores, rede industrial, terminais, ambiente *cloud* e recursos físicos (Lin et al., 2018).

A primeira entidade é responsável por armazenar e realizar o consenso PBTF sobre os novos blocos da Blockchain. A rede industrial e o ambiente *cloud* são responsáveis por agir de acordo com as informações registradas na Blockchain pelos terminais. As operações podem ser de controle ou leitura. Sendo que quando é uma informação de controle, a rede industrial emite comandos aos recursos físicos e quando é de leitura, o ambiente *cloud* retorna os dados aos terminais. Vale ressaltar que para a autenticidade dos dados é utilizado o esquema *Attribute-based signature* (ABS) e para a confidencialidade é utilizada a criptografia *Multi-receivers encryption* (MRE) (Lin et al., 2018).

Segundo Lin et al. (2018), a autoridade responsável pela fábrica interage com os terminais, *gateway* da rede industrial e o *gateway* da *cloud* para gerar o par de chaves utilizados no MRE. Além disso, a autoridade gera os parâmetros públicos usados no ABS. Quando cada entidade se registra a ela, ela computa a chave de assinatura baseada no atributo do terminal. Dessa forma, é possível realizar autenticação das mensagens e a privacidade dos dados contra um agente externo ao sistema. Todavia, a proposta de Lin et al. (2018) apresente ponto único de falha nessa autoridade, uma vez que ela inicia e atualiza os participantes do sistema, além de gerar os parâmetros utilizados no ABS e MRE.

Em contrapartida, a trabalho de Wang et al. (2019) utiliza contratos inteligentes em conjunto com Criptografia Baseada em Atributos de política de texto cifrado ou *(Ciphertext-Policy Attribute-Based Encryption* (CP-ABE) para controle de acesso da leitura de informações armazenadas em ambientes em nuvem. O primeiro é acordado entre os donos e usuários das informações e armazena o

identificador da informação e um mapa de usuário e chave para fazer a decifragem dos dados. Por outro lado, o CP-ABE é aplicado para realizar um controle de acesso dos usuários que podem ou não ler os dados. Além dessas entidades, um servidor *Cloud* e a rede Blockchain Ethereum fazem parte do sistema proposto. Os donos dos dados utilizam o contrato inteligente para enviar o dados cifrados ao servidor *Cloud*. O usuário de dados faz requisição de acesso aos dados através do contrato inteligente. O dono dos dados definem um atributo que é o período de tempo que o acesso é válido e o usuário pode decifrar e recuperar os dados armazenados no servidor *Cloud*. Contudo, o usuário recupera a chave usada para decifragem através do contrato inteligente e um atacante realizando um *sniff* no canal de comunicação pode recuperar a chave e acessar o dado, causando uma fragilidade na proposta.

Em contraparte, a referência Liu et al. (2020) aplica o controle de acesso *attributed based access control* (ABAC) para dispositivos relacionados ao conceito de *Internet of Things* (IoT). Para isso, três contratos inteligentes são usados: `POLICY`, `DEVICE` e `ACCESS`. O primeiro é responsável por gerenciar as políticas ABAC, o segundo armazena a URL dos recursos produzidos pelos dispositivos e o método para consultá-los. Por fim, o terceiro é usado pelos usuários para acessar os recursos. Vale ressaltar que a proposta utiliza uma Blockchain privada e é necessária a configuração de uma CA, criando ponto-único de falha na proposta Liu et al. (2020). O esquema R2SC não apresenta ponto-único de falha e utiliza o modelo RBAC para controle de acesso. A seguir, na Seção 4, a arquitetura proposta é descrita.

4 Arquitetura Proposta

A arquitetura R2SC para controle de acesso RBAC usando contratos inteligentes é composta por três contratos inteligentes: `Manager Role`, `Role` e `Service`. O primeiro é responsável por gerenciar os participantes do contrato `Role`. O contrato inteligente `Role` contém as contas que solicitarão acesso a `Service`. Por fim, `Service` armazena os `Role`s e seus respectivos acessos que podem ser de leitura, escrita ou execução.

Vale ressaltar que um contrato `Manager Role` pode gerenciar diversos `Roles`. Por sua vez, esse tipo de contrato pode estar vinculado a diferentes `Services` com diferentes permissões de acesso. E `Service`, por outro lado, possui diversos `Roles` com diferentes tipos de acesso. Adicionalmente, um `Service` pode ter diversos operadores para que não se crie ponto-únicos de falha. Essa característica é importante uma vez que a Blockchain se caracteriza por ser distribuída. Portanto, as aplicações mantidas nela devem seguir a mesma característica.

Inicialmente, uma conta é responsável por criar um contrato de qualquer um dos tipos. Não existe um requisito relacionado a ordem da criação dos contratos. Contudo, para seu funcionamento é necessário um `Manager Role` ser vinculado a um `Role` para gerir os usuários desse contrato e um `Role` e suas permissões devem ser adicionados a um `Service`. Uma vez que essas etapas estão feitas, o usuário pode requisitar acesso ao serviço. A Fig. 1 demonstra esse processo.

Para facilitar o entendimento da proposta do trabalho, um exemplo é apresentado na Fig. 2. A aplicação é da infraestrutura de recarga de veículos elétricos composta por três componentes: centro de operações, veículos elétricos e Blockchain.

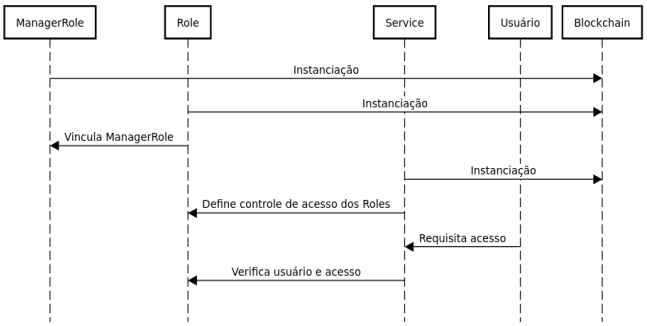

Figura 1: Diagrama sequencial para funcionamento da arquitetura proposta.

O centro de operações vai colocar em funcionamento os contratos inteligentes `ManagerRole`, `Role BIDIRECIONAL`, `Role UNIDIRECIONAL` e `Service ENERGIA` na Blockchain. O primeiro já vai possuir o centro de operações como um dos participantes. Após, ele vincula contrato inteligente `ManagerRole` a `UNIDIRECIONAL` e `BIDIRECIONAL`. Por fim, ele define que o primeiro terá acesso de leitura apenas em `ENERGIA` e o segundo terá permissão de leitura e escrita. No exemplo, os veículos contatam o centro de operações indicando seu modo de recarga e partir disso, o centro de operações adiciona o usuário a um dos dois contratos inteligentes.

Isso indica que o primeiro poderá apenas fazer consumo de energia elétrica, enquanto que os usuários que façam parte de `BIDIRECIONAL` podem fazer o consumo e inserção de energia elétrica como mostra a Fig. 3. Os veículos contatam uma estação de recarga, ela por sua vez, verifica as permissões de acesso do usuário do veículo e realiza (ou não) a recarga/inserção de energia elétrica do veículo.

4.1 Contrato Inteligente `Manager Role`

O contrato inteligente denominado `Manager Role` armazena o endereço das contas responsáveis por gerir os participantes considerados *Managers*. Basicamente, ele é composto por funcionalidade de adição e remoção de contas.

Para a adição de uma nova conta ao contrato, é necessário que quem faz a requisição esteja adicionado previamente e que o novo endereço não faça parte do contrato inteligente. Vale ressaltar que quando o contrato inteligente é instanciado, o endereço de quem coloca o contrato inteligente em funcionamento na Blockchain é adicionado na lista de *Managers*.

Por outro lado, para a operação de remoção de uma conta é necessário que ambos façam parte do contrato inteligente e que sempre exista pelo menos um endereço na lista de *Managers*.

4.2 Contrato Inteligente `Role`

O contrato inteligente denominado `Role` é responsável por armazenar e gerenciar as contas vinculadas a um determi-

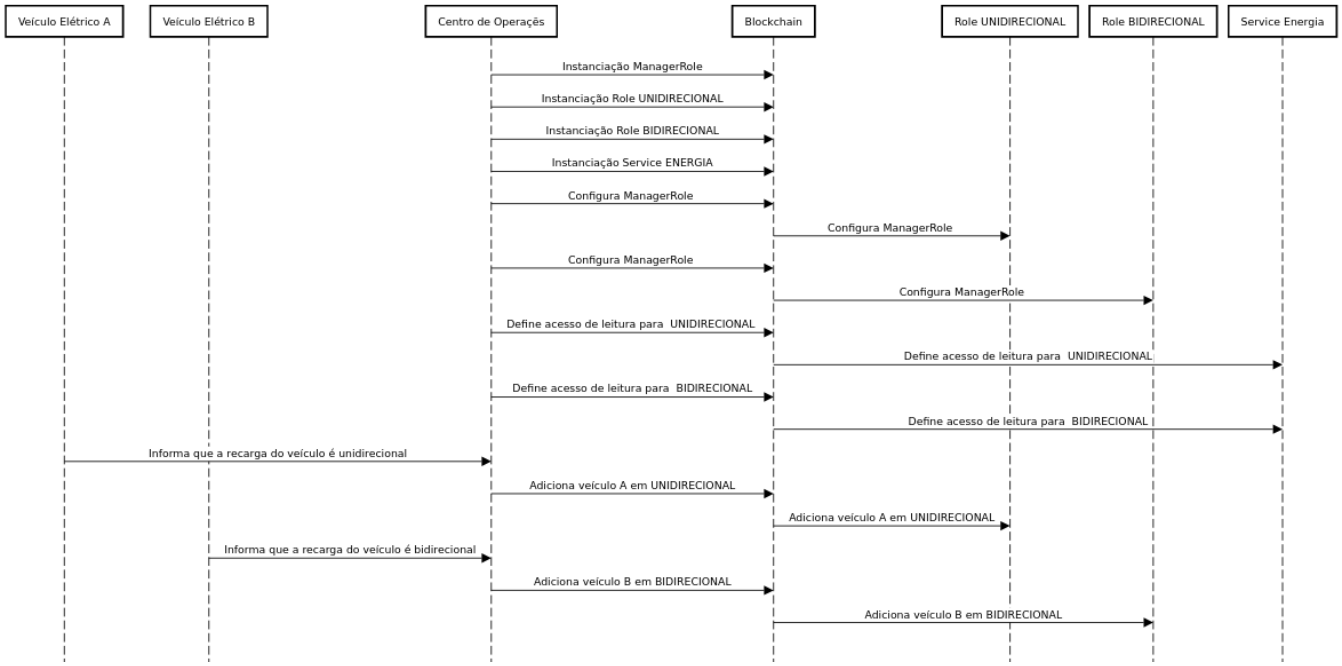

Figura 2: Cenário de exemplo de uso do controle de acesso.

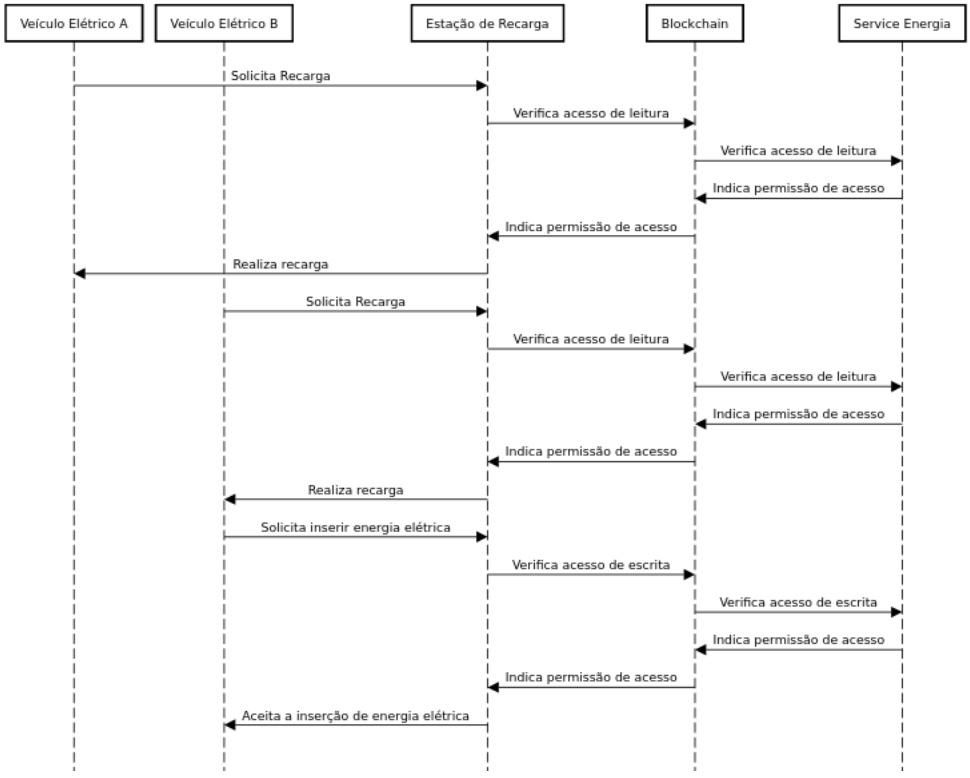

Figura 3: Exemplo de solicitação de recarga e inserção de energia usando o controle de acesso.

nado papel. Essas contas irão solicitar acesso ao contrato de inteligente `Service`. O contrato inteligente `Role` utiliza de `Manager Role` para adicionar ou remover as contas. Tal contrato armazenará quem pode alterar a lista de usuários armazenados em `Role`. Dessa forma, essa é a primeira operação a ser feita no contrato inteligente após ele ter sido colocado em funcionamento na Blockchain.

A partir disso, é possível adicionar ou remover mem-

bros de `Role`. O método de adição requer que quem faz a solicitação seja um endereço que faça parte do contrato inteligente `Manager Role` e que o novo endereço não é um membro que faça parte do contrato inteligente `Role`.

Em contrapartida, o método de remoção requer que quem faça a solicitação seja um endereço que faça parte do contrato inteligente `Manager Role` e que o novo endereço faça parte do contrato inteligente `Role`.

4.3 Contrato Inteligente `Service`

O contrato inteligente denominado `Service` armazena as contas responsáveis por gerir o controle de acesso dos `Roles`. Ele é composto por procedimentos de verificação de permissão de leitura, escrita e execução.

Adicionalmente, ele tem procedimentos para adicionar um novo `Role` e suas permissões. O contrato inteligente também contém funções para adicionar e remover operadores do contrato, que são as entidades responsáveis por gerenciar o controle de acesso dos `Roles`.

5 Análises e Discussões

Essa Seção apresenta os testes unitários de cada um dos contratos inteligentes. Para isso, o *framework* truffle é utilizado. Ele permite a utilização de quantidade arbitrária de endereços na Blockchain (Hartel and van Staalduinen, 2019). Os testes foram feitos usando a linguagem de programação JavaScript e executaram em uma máquina com Ubuntu 20.04 e processador AMD Ryzen 5 3500U e memória RAM de 8GB.

5.1 Testes do Contrato Inteligente `ManagerRole`

Essa Subseção aborda os testes do contrato inteligente `ManagerRole`. Os testes feitos servem para validar a lógica dos contratos inteligentes. O primeiro cenário de teste é de adição de uma conta no contrato inteligente. Para isso, a conta é adicionada e após, uma consulta é feita para validar se isso ocorreu com sucesso.

Em sequência, o segundo teste tenta adicionar novamente uma conta. Conforme dito alhures, para adição de uma conta é necessário que quem faça a requisição seja um membro do contrato inteligente e que a nova conta não seja. O teste espera que uma mensagem de erro seja emitida pelo contrato inteligente. Por outro lado, o terceiro teste verifica a remoção de um membro do contrato inteligente `ManagerRole`.

Após isso, o quarto teste tenta remover novamente uma conta não pertencente ao contrato inteligente. Conforme dito acima, é necessário que quem faça a requisição e conta a ser removida sejam membros do contrato inteligente. O teste espera que uma mensagem de erro seja emitida pelo contrato inteligente. Por fim, o último teste unitário desenvolvido verifica que uma conta não pertencente ao contrato inteligente não pode remover um membro existente. O resultado da execução dos testes é apresentado na Fig. 4. Ela indica que todos os testes passaram e a lógica do contrato inteligente `ManagerRole` está em acordo com a proposta do trabalho.

Após validar o funcionamento do contrato inteligente,

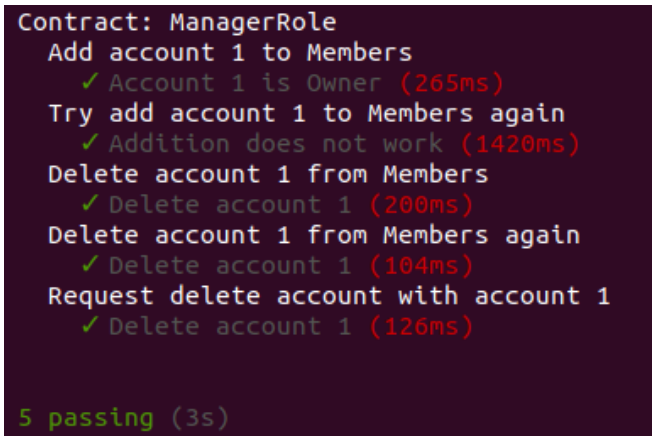

Figura 4: Resultado dos testes unitários do contrato inteligente `ManagerRole`.

é necessário verificar a escalabilidade da aplicação. Dessa forma, foi implementado um teste que percorre a lista de membros do contrato inteligente `ManagerRole`. Dez (10) rodadas foram executadas, calculando o tempo para percorrer uma lista com um (1), dez (100), cem (100) e mil (1000) participantes. Após isso, foi feita uma média dos valores obtidos, com o resultado apresentado na Fig. 5. O crescimento no tempo é linear devido a busca sequencial utilizada para verificar um membro no contrato inteligente.

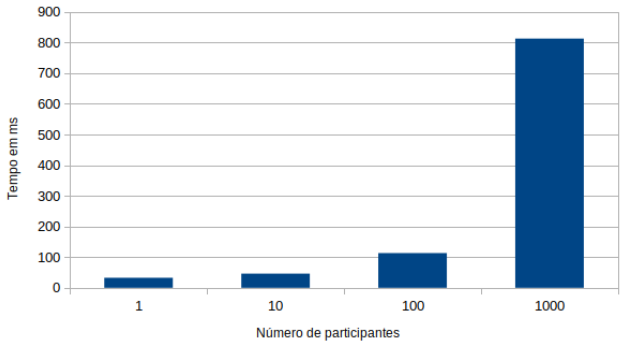

Figura 5: Tempo de execução para percorrer a lista de participantes do contrato inteligente `ManagerRole`.

5.2 Testes do Contrato Inteligente `Role`

Essa Subseção aborda os testes do contrato inteligente `Role`. Os testes realizados servem para verificar: (i) se a lógica dos contrato inteligente está em conformidade com a proposta e (ii) a escalabilidade do contrato inteligente `Role`.

O contrato inteligente `Role` precisa de um contrato inteligente `ManagerRole` configurado para poder adicionar e remover membros. O primeiro teste verifica que o contrato inteligente emite uma mensagem de erro ao receber uma requisição de adição de membro sem um `ManagerRole` con-

figurado. Já o segundo teste verifica a tentativa de remoção de um membro. É esperado um erro como no processo de adição.

Por outro lado, o terceiro teste faz a adição de um membro no contrato inteligente `Role` com `ManagerRole` configurado previamente e dessa forma, a execução ocorre corretamente. Por fim, o teste verifica se a conta adicionada é um membro do contrato inteligente `Role`. Em contrapartida, o quarto teste tenta realizar a adição de um membro já existe ao contrato inteligente `Role`. O contrato inteligente emitirá erro na execução e isso é esperado pela lógica da implementação realizada. Além disso, um teste de adição solicitado por um membro inválido também é feito. Testes de remoção sem membros no contrato inteligente `Role` e a partir de contas inválidas também foram realizados.

O teste em sequência é da adição de uma nova conta ao contrato inteligente `Role`. O contrato inteligente executa o método normalmente uma vez que a requisição é feita por um membro do contrato inteligente `ManagerRole`.

Em contrapartida, um teste de remoção da conta adicionada é o nono teste. Após, é verificado se a conta ainda faz parte do contrato inteligente. O método de execução ocorre com sucesso pois a conta ser removida é de um membro válido e quem faz a requisição faz parte do contrato inteligente `ManagerRole`.

Após a definição dos testes, a Fig. 6 apresenta seus resultados. Ela indica que todos os testes passaram. Isso implica na execução correta dos testes em acordo com a proposta.

```
Contract: Role
  Try to add a member without manager
    ✓ Add a member without manager configured (2364ms)
  Try to del a member without manager
    ✓ Del a member without manager configured (226ms)
  Add one member
    ✓ verify account 0 is member of Role (678ms)
  Error adding an existent member
    ✓ Add account at 0 into member again (301ms)
    ✓ Add account at 0 into member from an invalid owner (272ms)
  Del an invalid member
    ✓ Try to del a member without member (268ms)
    ✓ Try to del a member from invalid account (361ms)
  Add account 1 to Role
    ✓ Account 1 is member of Role (397ms)
  Del account 1 to Role
    ✓ Account 1 is not member of Role (786ms)


9 passing (7s)
```

Figura 6: Resultado dos testes unitários do contrato inteligente `Role`.

Um vez que o funcionamento do contrato inteligente foi validado, a escalabilidade do contrato inteligente também foi verificada. Os valores foram obtidos da mesma maneira do teste de escalabilidade do contrato inteligente `ManagerRole` com o resultado apresentado na Fig. 7 que indica o mesmo comportamento do contrato inteligente `ManagerRole`.

5.3 Testes do Contrato Inteligente `Service`

Essa Subseção aborda os testes do contrato inteligente `Service`. Os testes feitos servem para validar a lógica do

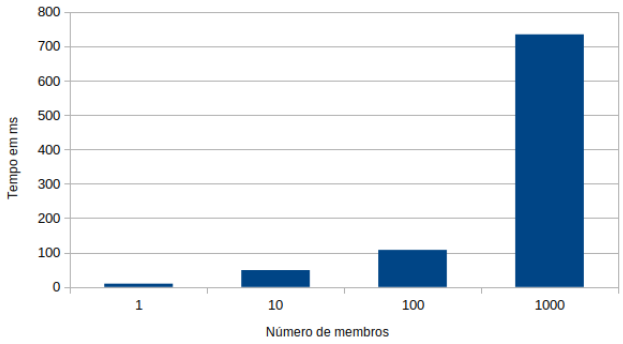

Figura 7: Tempo de execução para percorrer a lista de participantes do contrato inteligente `Role`

contrato inteligente. Todos os testes descritos a seguir verificam a permissão de leitura, escrita e execução.O primeiro teste verifica o acesso a um serviço sem nenhum `ManagerRole` configurado e nenhum `Role` com permissões definidas. Dessa forma, qualquer tentativa de leitura, escrita ou execução será negada.

Adicionalmente, segundo verifica as permissões com um contrato inteligente `ManagerRole` configurado e nenhum `Role` é adicionado. Por outro lado, o terceiro teste é com um `Role` adicionado. Todavia, a permissão de leitura, escrita e execução estão negados. Dessa forma, qualquer tentativa de acesso por um usuário que esteja adicionado somente neste `Role` será negado.

Em sequência, é feito um teste com um `Role` tendo acesso de leitura, escrita e execução. Todavia, a verificação de permissão é feita usando uma conta não pertencente ao contrato inteligente. Dessa forma, todas as permissões retornarão falso.

Em contraparte, o quinto teste verifica as permissões de um usuário pertencente a um `Role` com permissão de leitura, escrita e execução. Portanto, todas as verificações de uma conta pertencente ao `Role` devem retornar verdadeiras.

Por outro lado, o sexto teste verifica um `Role` com acesso somente de leitura apenas. Para validar isso, é verificado o acesso de leitura, escrita e execução, esperando que apenas o primeiro retorne verdadeiro. Após, o `Role` é alterado com permissão de escrita apenas. O resultado esperado é o mesmo descrito alhures, diferenciando nas chamas de verificação de acesso de leitura, que é esperado um resultado de permissão negada e na verificação de escrita, em que é esperado uma permissão concedida.

Adicionalmente, o último teste de verificação de um `Role` com acesso de execução permitido. Os acessos de leitura, escrita e execução são verificados. O resultado esperado é que os dois primeiros sejam falsos e o último verdadeiro. Por fim, o resultado da execução dos testes é apresentado na Fig. 8. Ela indica que todos os testes passaram e a lógica do contrato inteligente `Service` está em acordo com a proposta do trabalho. Portanto, é possível utilizar o contrato para definição de regras de controle de acesso.

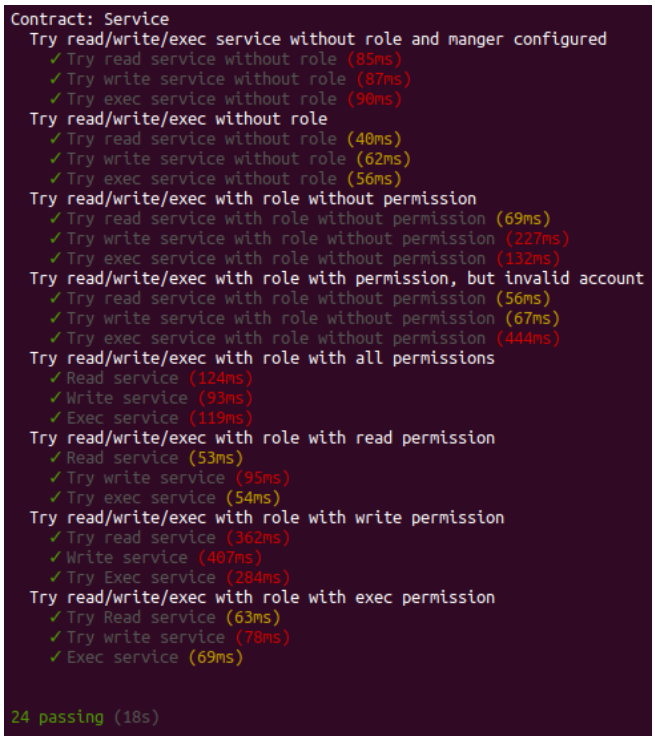

Figura 8: Resultado dos testes do contrato inteligente `Service`.

5.4 Comparação com Trabalhos Relacionados

A Tabela 1 apresenta as métricas de comparação como a escalabilidade, se a proposta apresenta ponto-único de falha e a utilização do modelo RBAC. O trabalho proposto e (Wang et al., 2019) não apresentam ponto-único de falha, uma vez que o primeiro não é restrito a Blockchain privadas ou em consórcio que precisam de uma ICP configurada para definir os participantes da rede. Esse problema ocorre na proposta de (Liu et al., 2020) que utiliza CA para indicar os nós que participarão do consenso e da rede. Na referência (Lin et al., 2018) o ponto único de falha se encontra na autoridade central responsável por iniciar os parâmetros públicos e registrar membros no sistema. Isso implica que a proposta não apresenta escalabilidade.

Além disso, o modelo RBAC é utilizada apenas pelo trabalho proposto. Ele apresenta maior flexibilidade relacionado ao modelo ABAC, uma vez que a manutenção de uma regra mal configurada em um modelo ABAC requer que os atributos sejam reconfigurados em todas entidades que o possuem enquanto que no modelo RBAC basta remover a permissão. As propostas de (Wang et al., 2019) e (Liu et al., 2020) são similares no sentido de utilizarem um controle de acesso baseado em atributos. O primeiro utiliza CP-ABE e segundo usa o modelo ABAC. Em contraparte, a referência (Lin et al., 2018) faz uso de um esquema MRE para definir quem pode ou não decifrar os dados. Sendo assim, o esquema R2SC é o único dos trabalhos que atinge as métricas definidas para comparação.

Tabela 1: Comparação de Características dos Trabalhos

Trabalho	Escalabilidade	Facilidade de Modificação das Permissões	Sem ponto-único de falha
Lin et al. (2018)	X	X	X
Wang et al. (2019)	V	X	V
Liu et al. (2020)	X	X	X
R2SC	V	V	V

6 Considerações Finais

A tecnologia Blockchain aparece como um conceito amplamente difundido e utilizado na literatura devido a disponibilidade, imutabilidade e integridade dos dados que ela possui. Além disso, a Blockchain fornece o conceito de contrato inteligente, permitindo uma diversidade de aplicações distribuídas.

Este trabalho faz uso disso para propor um sistema de controle de acesso RBAC. A proposta é verificada utilizando o *framework* Truffle e os testes demonstram êxito na lógica dos contratos inteligentes. Adicionalmente, a proposta não apresenta ponto-único de falha como ocorre em alguns trabalhos relacionados.

Por fim, as principais contribuições deste trabalho consistem em apresentar um sistema RBAC escalável utilizando contratos inteligentes, desenvolver a proposta utilizando a linguagem de programação Solidity e implementar testes para a sua validação. Como trabalho futuro fica o estudo de caso da proposta em ambiente real e sua utilização em conjunto com sistemas de autenticação para abordar a tríade *Authentication*, *Authorization* e *Accounting* (AAA).

Agradecimentos

Os autores agradecem ao INCTGD, órgãos financiadores (CNPq processo n° 465640/2014-1, CAPES processo n° 23038.000776/2017-54 e FAPERGS n° 17/2551-0000517-1).

Referências

Aggarwal, S., Kumar, N. and Gope, P. (2020). An efficient blockchain-based authentication scheme for energy-trading in v2g networks, *IEEE Transactions on Industrial Informatics* **17**(10): 6971–6980. https://doi.org/10.1109/TII.2020.3030949.

Ante, L. (2021). Smart contracts on the blockchain–a bibliometric analysis and review, *Telematics and Informatics* **57**: 101519. https://doi.org/10.1016/j.tele.2020.101519.

Bach, L. M., Mihaljevic, B. and Zagar, M. (2018). Comparative analysis of blockchain consensus algorithms, *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, IEEE, pp. 1545–1550. https://doi.org/10.23919/MIPRO.2018.8400278.

Bertineti, D. P., Canha, L. N., Brignol, W., Medeiros, A. P., de Azevedo, R. M. and Nadal, Z. L. (2020). Flexible

energy management strategy for electric vehicles charging stations, *2020 55th International Universities Power Engineering Conference (UPEC)*, IEEE, pp. 1–6. https://doi.org/10.1109/UPEC49904.2020.9209763.

Dias, L. and Rizzetti, T. A. (2021). A review of privacy-preserving aggregation schemes for smart grid, *IEEE Latin America Transactions* **19**(7): 1109–1120. https://doi.org/10.1109/TLA.2021.9461839.

Dias, L. V., Rizzetti, T. A., Brignol, W. S. and Canha, L. N. (2021). Inserção de infraestrutura de chave pública no projeto opendht, *Anais do XXI Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*, SBC, pp. 379–384. https://doi.org/10.5753/sbseg.2021.17329.

Ghafoorian, M., Abbasinezhad-Mood, D. and Shakeri, H. (2018). A thorough trust and reputation based rbac model for secure data storage in the cloud, *IEEE Transactions on Parallel and Distributed Systems* **30**(4): 778–788. https://doi.org/10.1109/TPDS.2018.2870652.

Hartel, P. and van Staalduinen, M. (2019). Truffle tests for free–replaying ethereum smart contracts for transparency, *arXiv preprint arXiv:1907.09208* . https://doi.org/10.48550/arXiv.1907.09208.

Hewa, T., Ylianttila, M. and Liyanage, M. (2021). Survey on blockchain based smart contracts: Applications, opportunities and challenges, *Journal of Network and Computer Applications* **177**: 102857. https://doi.org/10.1016/j.jnca.2020.102857.

Kim, M., Park, K., Yu, S., Lee, J., Park, Y., Lee, S.-W. and Chung, B. (2019). A secure charging system for electric vehicles based on blockchain, *Sensors* **19**(13): 3028. https://doi.org/10.3390/s19133028.

Lin, C., He, D., Huang, X., Choo, K.-K. R. and Vasilakos, A. V. (2018). Bsein: A blockchain-based secure mutual authentication with fine-grained access control system for industry 4.0, *Journal of network and computer applications* **116**: 42–52. https://doi.org/10.1016/j.jnca.2018.05.005.

Liu, H., Han, D. and Li, D. (2020). Fabric-iot: A blockchain-based access control system in iot, *IEEE Access* **8**: 18207–18218. https://doi.org/10.1109/ACCESS.2020.2968492.

Miglani, A., Kumar, N., Chamola, V. and Zeadally, S. (2020). Blockchain for internet of energy management: Review, solutions, and challenges, *Computer Communications* **151**: 395–418. https://doi.org/10.1016/j.comcom.2020.01.014.

Quincozes, S. E., Albuquerque, C., Passos, D. and Mossé, D. (2021). A survey on intrusion detection and prevention systems in digital substations, *Computer Networks* **184**: 107679. https://doi.org/10.1016/j.comnet.2020.107679.

Shi, S., He, D., Li, L., Kumar, N., Khan, M. K. and Choo, K.-K. R. (2020). Applications of blockchain in ensuring the security and privacy of electronic health record systems: A survey, *Computers & Security* p. 101966. https://doi.org/10.1016/j.cose.2020.101966.

Szabo, N. (1997). Formalizing and securing relationships on public networks, *First monday* . https://doi.org/10.5210/fm.v2i9.548.

Viriyasitavat, W. and Hoonsopon, D. (2019). Blockchain characteristics and consensus in modern business processes, *Journal of Industrial Information Integration* **13**: 32–39. https://doi.org/10.1016/j.jii.2018.07.004.

Wang, Q. and Su, M. (2020). Integrating blockchain technology into the energy sector—from theory of blockchain to research and application of energy blockchain, *Computer Science Review* **37**: 100275. https://doi.org/10.1016/j.cosrev.2020.100275.

Wang, S., Wang, X. and Zhang, Y. (2019). A secure cloud storage framework with access control based on blockchain, *IEEE access* **7**: 112713–112725. https://doi.org/10.1109/ACCESS.2019.2929205.

Xiao, Y., Zhang, N., Lou, W. and Hou, Y. T. (2020). A survey of distributed consensus protocols for blockchain networks, *IEEE Communications Surveys & Tutorials* **22**(2): 1432–1465. https://doi.org/10.1109/COMST.2020.2969706.

Zheng, Z., Xie, S., Dai, H.-N., Chen, W., Chen, X., Weng, J. and Imran, M. (2020). An overview on smart contracts: Challenges, advances and platforms, *Future Generation Computer Systems* **105**: 475–491. https://doi.org/10.1016/j.future.2019.12.019.