Autenticação entre Aplicações Distribuídas com Blockchain[1]

Luiz Fernando Perez Redondaro

Faculdade de Ciências e Tecnologia
Universidade Estadual Paulista "Júlio de Mesquita Filho" (FCT/UNESP)
Departamento de Matemática e Computação – Presidente Prudente, SP, Brasil

luiz.redondaro@unesp.br

Abstract. *This work demonstrated the use of blockchain technology in application authentication on a distributed system. For this purpose, a blockchain structure and distributed application system with diverse functionalities were created, with the applications being integrated with the blockchain, so that they could authenticate their messages and store data on their functionalities. Several communications were carried out in order to verify the expected properties.*

Resumo. *Este trabalho demonstrou o uso da tecnologia blockchain na autenticação de aplicações em um sistema distribuído. Para tal finalidade, foi criada uma estrutura blockchain e sistema de aplicações distribuídas com funcionalidades diversas, sendo as aplicações integradas com a blockchain, de modo que pudessem autenticar suas mensagens e armazenar dados de suas funcionalidades. Foram realizadas várias comunicações, de modo a verificar as propriedades esperadas.*

1. Introdução

Aplicação distribuída é um sistema formado por um conjunto de aplicações executadas em diferentes computadores, que trabalham de forma coordenada para prover um determinado serviço. Sistemas deste tipo compõem variados tipos de serviços presentes na internet, que vão desde servidores de jogos, e-mail e multimídia a serviços de compra e venda, transações bancárias, sistemas restritos de corporações ou do governo.

As aplicações distribuídas se popularizaram e são incorporadas em operações com informações valiosas, tornando-se evidente que os riscos por ações mal-intencionadas aumentam, visando finalidades diversas como interceptar, alterar ou destruir informações passadas entre dispositivos, ou até mesmo instalar código malicioso e obter acesso indevido, não autorizado, ao sistema distribuído ou ao sistema do usuário final.

A tecnologia blockchain é basicamente constituída de uma corrente de blocos interligados pelos respectivos valores em código hash, sendo amplamente utilizada na tecnologia das moedas virtuais. Esta tecnologia tem sido adotada em variados serviços e sistemas, podendo ser utilizada em aplicações distribuídas, de modo a acrescentar autenticação segura e criptografia [Patgiri, Acharjamayum e Devi, 2019].

---

1 Trabalho de Conclusão de Curso. Orientação: Prof. Dr. Milton Hirokazu Shimabukuro

1.1. Objetivos

Este trabalho de conclusão de curso (TCC), como um todo, teve o objetivo principal de investigar o uso da tecnologia blockchain com aplicações distribuídas, de modo a implementar autenticação entre as aplicações, aplicando seus conceitos e realizando simulações, adicionando melhorias e soluções com blockchain em aplicações distribuídas.

1.2. Estrutura do documento

Neste texto, são apresentadas fundamentação, revisão da literatura, implementação do experimento, execução e análise de resultados. A seção 2 discorre sobre aplicações distribuídas e blockchain, apresentando conceitos que serviram de base teórica para o trabalho, etapas que foram seguidas e algumas técnicas mais conhecidas. Na seção 3, são apresentados alguns trabalhos relacionados envolvendo aplicações distribuídas com blockchain. Na seção 4, são tratadas a implementação, a execução e os testes de uma simulação de um sistema distribuído com blockchain, composto por um conjunto de aplicações distribuídas com funcionalidades diversas, às quais puderam autenticar comunicações e coordenar ações através da blockchain. Na seção 5, são apresentadas as conclusões finais após toda a análise da implementação e possibilidades para trabalhos futuros.

2. Fundamentação Teórica

Nesta seção, são apresentados os conceitos de aplicações distribuídas, segurança, redes peer-to-peer e blockchain, para então fundamentar a autenticação entre aplicações distribuídas com a tecnologia blockchain.

2.1. Aplicações Distribuídas

Courouris, Dollimore, Kindberg e Blair (2013) definem sistemas distribuídos como “componentes de hardware ou software, localizados em computadores interligados em rede, comunicam-se e coordenam suas ações apenas enviando mensagens entre si”. O objetivo de sistemas distribuídos é compartilhar recursos computacionais entre máquinas conectadas em uma rede, abrangendo desde componentes de hardware como discos e impressoras, até elementos de software, como arquivos, bancos de dados e objetos de dados de todos os tipos. Para o presente trabalho, são abordados os softwares, que são os programas em execução (processos) nas máquinas de sistemas distribuídos, e é utilizada a expressão aplicações distribuídas. Na Figura 1 é exemplificado um sistema distribuído de um portal de turismo, composto de um servidor principal que atende os usuários e acessa outros serviços distribuídos de hotéis, passagens e locadoras de veículos.

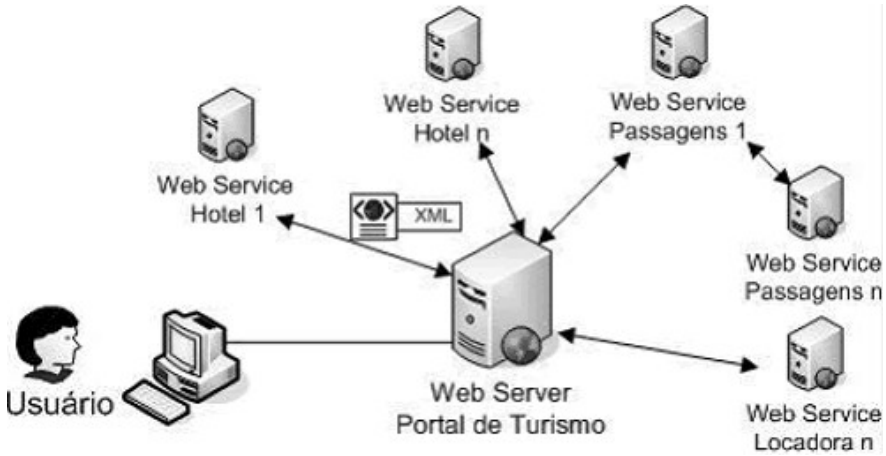

Figura 1. Exemplo de sistema distribuído de um portal de turismo. Fonte: culturamix.com, 2013.
http://tecnologia.culturamix.com/dicas/o-que-e-um-sistema-distribuido

Um sistema de aplicações distribuídas possui recursos modularizados entre computadores, fornecidos pelo respectivo processo em execução de cada máquina, com restrição de acesso, de forma que outras aplicações possam acessá-los por mecanismos de comunicação. Cada processo atua como cliente e/ou servidor, sendo o servidor responsável por aceitar pedidos de serviço de processos clientes e responder apropriadamente, e o cliente que envia pedidos para determinado processo servidor e recebe a devida resposta. O processo que atua apenas como cliente funciona apenas enquanto a aplicação que esse faz parte está ativa, enquanto o servidor atua continuamente [Courouris, Dollimore, Kindberg e Blair, 2013].

As aplicações distribuídas estão presentes em vários serviços cotidianos, como páginas na internet, jogos *online*, *e-mails*, redes sociais, comércio eletrônico, entre outros, sendo tendência o aumento da quantidade de aplicações distribuídas, em especial para telefonia móvel, serviços multimídia e produtos de software [Courouris, Dollimore, Kindberg e Blair, 2013].

Tanenbaum e van Steen (2007) acrescentam as vantagens das aplicações distribuídas:

- Facilidade de acessar e compartilhar recursos remotos;
- Transparência ao apresentar aplicações distribuídas como um único sistema, ocultando o fato de que processos e recursos estão fisicamente distribuídos por vários computadores;
- Escalabilidade ao possibilitar que o sistema aumente de tamanho em nível físico, geográfico e administrativo.

2.2. Segurança

Courouris, Dollimore, Kindberg e Blair (2013) afirmaram que, como as aplicações distribuídas possuem recursos compartilhados por meio de uma rede com vários usuários, deve-se adotar mecanismos de segurança, de modo a permitir acesso aos recursos apenas aos usuários autorizados.

Goodrich e Tamassia (2012) destacam algumas propriedades de um sistema computacional em relação a sua segurança, as quais são alvos comuns em tentativas de

exploração por atacantes que pretendem obter algum tipo de acesso a conteúdo restrito ou comprometer o funcionamento de um sistema:

- Integridade é definida como uma propriedade de determinada informação, de que não seja alterada de maneira não autorizada;
- Confidenciabilidade é definida como a proteção de informações confidenciais, de forma a evitar a revelação não autorizada, ou seja, a proteção de dados, de modo que sejam acessados somente por aqueles que estiverem devidamente autorizados, impedindo que outros acessem o conteúdo;
- Autenticidade é definida como a legitimação de afirmações, políticas e permissões entre pessoas e sistemas;
- Anonimato é definido como a não atribuição de quaisquer registros ou transações ocorridas a qualquer indivíduo.

Courouris, Dollimore, Kindberg e Blair (2013) ressaltam a necessidade de adotar mecanismos de segurança para a proteção de sistemas distribuídos, dentre os quais, estabelecendo controles de acesso no projeto do sistema, registrando logs de acesso e utilizando criptografia digital.

Um arquivo de log de segurança deve conter uma sequência de registros, cada um destes contendo informações do tipo: identificação do usuário, operação executada, identificação do recurso solicitado e um carimbo de tempo. Quando houver necessidade analisar violações e ações suspeitas, poderá ser realizada uma auditoria do log [Courouris, Dollimore, Kindberg e Blair, 2013].

Goodrich e Tamassia (2012) definem a criptografia como uma tecnologia que permite comunicação confidencial entre dois participantes em um meio inseguro e sujeito à intromissão. Uma mensagem de texto a ser enviada é cifrada em um conjunto de caracteres que não revelará a mensagem original, de acordo com uma sequência de caracteres secreta -a chave- e um algoritmo criptográfico, para então ser enviada para o outro usuário, que usando os respectivos algoritmo e chave, poderá recuperar a mensagem original. Um exemplo de criptografia é mostrado na Figura 2.

Figura 2.Exemplo de criptografia. Fonte: [Drescher 2018]

Existem dois tipos de criptografia: simétrica e assimétrica. Na criptografia simétrica, uma única chave é utilizada tanto para criptografar, quanto descriptografar as mensagens. Desta forma, a chave deve ser mantida em segredo entre os participantes da comunicação. A criptografia assimétrica é mais sofisticada, na qual cada usuário possui duas chaves: uma pública e outra privada. A pública pode ser disponibilizada livremente, devendo ser mantida a chave privada em segredo. Assim, para receber mensagens criptografadas de outros usuários, basta que estes criptografem a mensagem com a chave pública, que então poderá ser descriptografada com a chave privada [Goodrich e Tamassia, 2012]. A Figura 3 ilustra uma criptografia assimétrica.

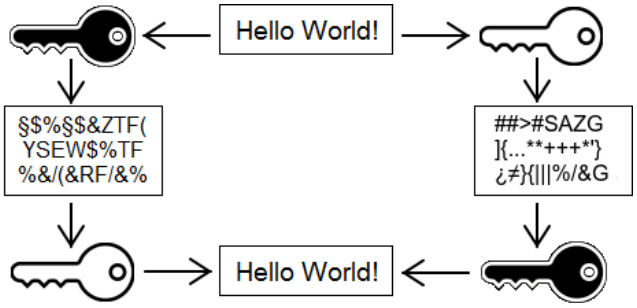

Figura 3. Criptografia assimétrica, com par de chaves pública/privada. Fonte: [Drescher 2018]

Stallings (2008) ressalta duas características importantes dos criptossistemas de chave pública/privada:

- É computacionalmente inviável determinar a chave privada com base na chave pública e do algoritmo de criptografia;
- Qualquer uma das duas chaves pode ser usada para criptografia, restando a outra como a respectiva chave para descriptografar.

Um importante recurso que é possibilitado pela criptografia de chave pública e privada é a assinatura digital, possibilitando autenticar a autoria de mensagens. Considerando que o destinatário tenha a chave pública, criptografa-se com a chave privada a mensagem inteira ou um código hash da mesma.

Drescher (2018) define que funções hash transformam algum tipo de dado em um valor numérico, independentemente do tamanho dos dados de entrada. Devem gerar valores idênticos para dados de entrada idênticos, deve ser difícil de gerar a mesma saída para entradas diferentes, e não permitindo que os valores de entrada sejam rastreados com base na saída, de forma que os dados de saída mudem de forma imprevisível, conforme seja alterada a entrada. A Figura 4 exemplifica o uso dos algoritmos mais comuns.

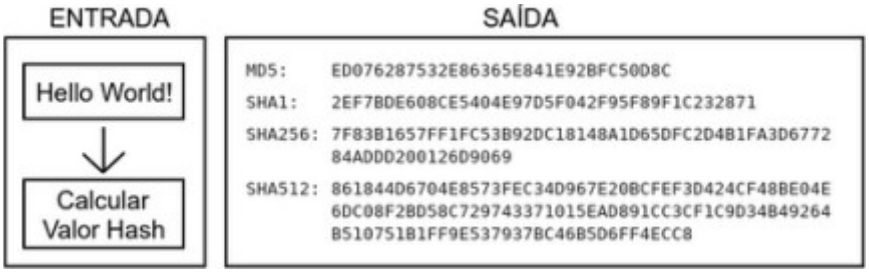

Figura 4. Cálculo de valores de hash para um texto pequeno com diferentes algoritmos. Fonte: [Drescher 2018]

Conforme Goodrich e Tamassia (2012), a solução de assinaturas digitais são derivadas do fato que, em sistemas de chave pública/privada, pode-se inverter a ordem de uso dos algoritmos de criptografar e descriptografar. Um usuário pode usar o algoritmo de descriptografar em uma mensagem, usando sua chave privada. A mensagem poderá ser recuperada utilizando o algoritmo de criptografia, por outro usuário que tenha a respectiva chave pública, confirmando-se desta forma a autoria da mensagem, pelo fato de que somente o portador da chave privada poderia ter feito a criptografia. Uma forma de assinatura digital comumente utilizada é aplicando a

criptografia no hash de mensagem, ao invés de aplicar na mensagem inteira.

Drescher (2018) explica o sistema de assinaturas digitais. O emissor gera o hash da mensagem, que então será criptografado com a chave privada, anexado com a mensagem, que então será enviada para o destinatário. Ao receber a mensagem, o receptor descriptografa o texto cifrado em anexo com a chave pública do emissor, obtendo o hash gerado pelo emissor. Gera-se o hash da mensagem recebida e o compara com o hash descriptografado, que caso sejam iguais, confirmará que a mensagem pertence ao emissor e que não foi alterada, pois qualquer mudança geraria um valor de hash diferente. Esse modelo é exemplificado na Figura 5.

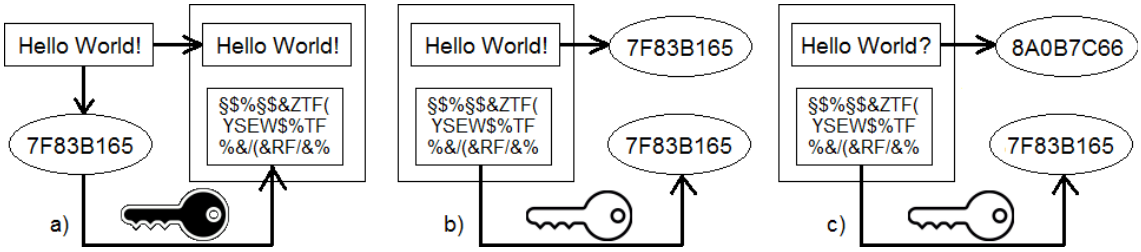

Figura 5. Assinaturas digitais: a) criando assinatura digital com chave privada; b) verificando assinatura digital com chave pública; c) mensagem alterada. Fonte: [Drescher 2018]

2.3. Peer-to-peer

Segundo Courouris, Dollimore, Kindberg e Blair (2013), o objetivo dos sistemas peer-to-peer é permitir que todos os nós da rede atuem tanto como cliente, quanto como servidor, possibilitando o compartilhamento de dados e recursos em uma escala muito grande, de forma descentralizada e organizada, equilibrando as cargas de armazenamento e processamento entre todos os computadores participantes. Desta forma, elimina-se a limitação de desempenho dado pela capacidade computacional dos servidores e sua infraestrutura associada, permitindo alta escalabilidade e tolerância a falhas ao sistema. Possui as seguintes características:

- oferece anonimato aos seus usuários;
- possui um algoritmo de distribuição e acesso aos dados, responsável por equilibrar a carga computacional e garantir disponibilidade;
- utiliza uma técnica de tolerância a falhas para inibir a possibilidade de falsificação de dados feita por computadores (nós) mal-intencionados.

A falta de garantia de disponibilidade está associada aos usuários, que podem não estar com seus computadores ligados.

O peer-to-peer surgiu em 2001, tendo evoluído ao longo de três gerações. Na primeira geração, os índices de arquivos eram centralizados, que direcionam para o endereço de rede dos computadores disponibilizadores do arquivo. Na segunda geração, os índices eram particionados e distribuídos juntos com os dados, oferecendo maior escalabilidade, anonimato e tolerância a falhas. A terceira geração é caracterizada pelo surgimento de camadas middleware, às quais fornecem abstração do gerenciamento de recursos distribuídos peer-to-peer para a aplicação, de modo a permitir a distribuição e recuperação de recursos (objetos de dados e arquivos) automaticamente, bem como permitir adicionar máquinas com recursos na rede ou removê-las.

Os recursos são indexados por funções de hash, gerando os seus identificadores. Os recursos são distribuídos aleatoriamente por computadores na internet, retirando deles as decisões sobre posições de recursos. As réplicas destes são colocadas de maneira estruturada com os computadores disponíveis, considerando a volatilidade, confiabilidade e requisitos de equilíbrio de carga, localização e uso de informações. Por meio do identificador gerado por hash, é possível a certificação automática do recurso, inibindo falsificações por nós não confiáveis. Por essa peculiaridade, redes peer-to-peer são apropriadas somente para dados imutáveis, caso contrário, cada mudança geraria um valor hash diferente.

Os nós e recursos de uma rede peer-to-peer são localizados pelo algoritmo de sobreposição de roteamento, presente em uma camada de middleware que direciona requisições de recursos para qualquer nó que contenha o recurso endereçado. É chamado de sobreposição porque implementa um mecanismo de gerenciamento e localização de objetos replicados, garantindo que qualquer nó acesse qualquer recurso, por meio do conhecimento dos demais nós para localizar o recurso destinado. Dessa forma, o conhecimento das localizações dos objetos é particionado e distribuído pela rede, possibilitando escalabilidade e disponibilidade, conforme visto na Figura 6.

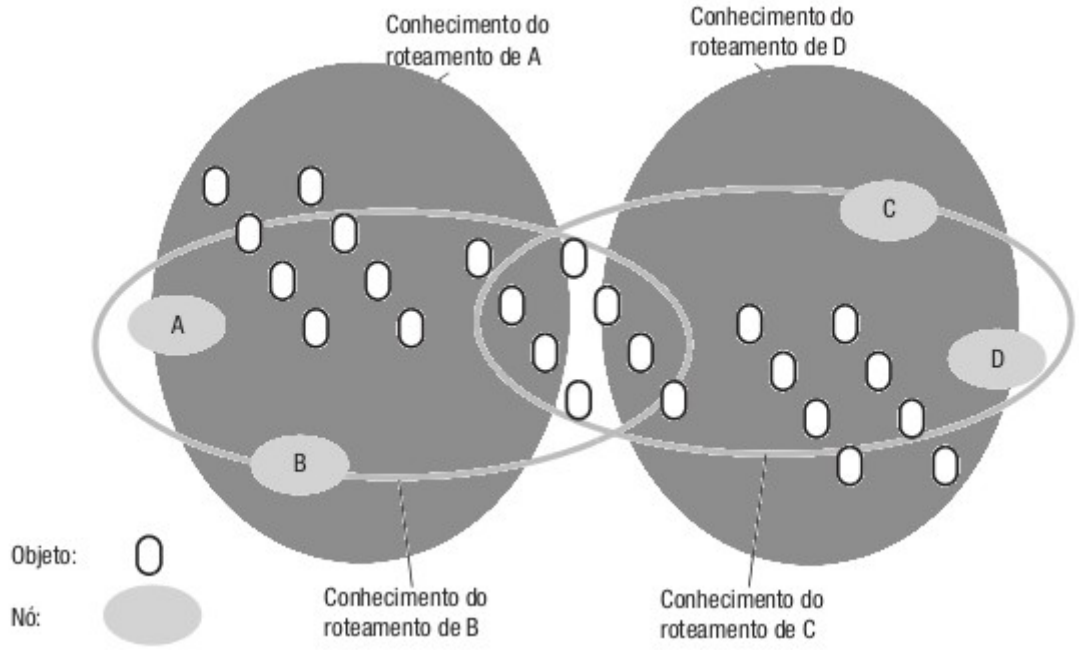

Figura 6. Distribuição de informações com sobreposição de roteamento. Fonte: [Courouris, Dollimore, Kindberg e Blair, 2013]

Existem duas estratégias de implementação do peer-to-peer: estruturadas e não estruturadas. Nas estruturadas, existe algum tipo de estrutura de dados distribuída, servindo como base para a localização de recursos, e um algoritmo operando sobre essa estrutura de dados. Dessa forma, os algoritmos oferecem bons limites de tempo para a localização de objeto, com alto custo computacional para manter as estruturas, em contrapartida. Em implementações não estruturadas, não há controle sobre a topologia ou posicionamento de objetos, sendo que cada nó que ingressa estabelece contato com seus vizinhos, criando a sobreposição de maneira ad hoc, de acordo com alguma regra local simples. Embora essa forma elimine o alto custo de manter as estruturas de dados da implementação estruturada em troca do desempenho das buscas e escalabilidade, muitas estratégias não estruturadas têm apresentado melhorias.

2.4. Blockchain

Nakamoto (2008) propôs um mecanismo para um sistema de moeda digital – o Bitcoin –, que funciona em uma rede peer-to-peer, herdando suas características, de modo que a moeda não passe por nenhuma instituição financeira. Dentre as principais características, estão:

- resistência ao gasto duplo: não permite que um usuário registre o gasto da mesma moeda digital mais de uma vez;
- transações irreversíveis: característica que impede o gasto duplo; uma vez que um usuário transferiu o determinado valor para outro usuário e a transação foi registrada, não é possível a reversão;
- pseudoanonimato: cada usuário gera endereços hash para usar nas transações, não necessitando a criação de contas;
- resistência a inserções e alterações maliciosas, desde que a maioria dos usuários seja honesta.

Segundo Patgiri, Acharjamayum e Devi (2019), o blockchain foi implementado pela primeira vez em 2009, no código fonte original do Bitcoin, no formato código aberto. Embora tenha servido de base tecnológica para diversas moedas virtuais, é uma tecnologia inovadora e promissora, que tem despertado interesse de bancos, empresas, organizações governamentais, entre outros setores.

Sua estrutura de dados é distribuída pela rede global de computadores, de forma descentralizada, ao invés de ficar armazenado em um único servidor. Registra transações de forma segura e confiável, ao mesmo tempo que é transparente e verificável, e também preserva o anonimato e a segurança de seus usuários. A estrutura de dados é um livro-razão, que utiliza um sofisticado sistema de criptografia para registrar as transações.

Drescher (2018) explica que é usada a estrutura de árvore Merkle para referenciar todas as transações de um bloco com um único valor de hash. Para gerar tal estrutura, deve-se obter o valor hash de todas as transações do bloco, e mantendo-os na ordem que as transações estão organizadas, agrupar em pares e tirar o hash de cada par. Com os valores de hash obtidos, é realizado o mesmo procedimento, que será repetido até que reste apenas um valor de hash, que será a árvore Merkle do bloco.

Cada bloco a ser adicionado na blockchain deverá atender a um determinado desafio, que consiste na restrição em que cada bloco deverá ter seu valor de hash iniciado com determinada quantidade de zeros. Para a resolução do desafio, inicia-se o

valor da variável nonce em 0 repete iterativamente a verificação de hash e adição de 1 no valor de nonce, até que o valor de hash atenda o desafio. Tal processo tem um custo computacional e o resultado obtido é chamado de prova de trabalho.

O blockchain é composto por uma lista de blocos ordenados, vinculados em ordem cronológica, cada qual possui um cabeçalho e suas transações. Para cada bloco, é gerado um valor de hash utilizando o algoritmo SHA256. O cabeçalho possui o valor hash do bloco anterior, *timestamp* do tempo em que foi criado, nonce e o valor de hash para a primeira transação do bloco, podendo também possuir o valor de hash para o próximo bloco. O primeiro bloco da estrutura é definido pai, e o atributo nonce é usado na prova de trabalho. As transações também possuem *timestamp*, estando ordenadas cronologicamente, em estrutura de árvore Merkle. A estrutura dos blocos é retratada na Figura 7. Para um que cliente que tenha o último bloco possa validá-lo, basta solicitar os blocos anteriores e recriar toda a cadeia de hash até o último, que coincidindo com o último, se certificará que nenhum bloco foi alterado até sua inclusão na cadeia, pois uma mera mudança ocasionaria um efeito dominó em tais valores. Os blocos estão distribuídos através de nós nas redes, os quais formam um consenso sobre o estado de uma transação a qualquer momento, por possuírem cópias das transações autenticadas entre eles [Patgiri, Acharjamayum e Devi, 2019].

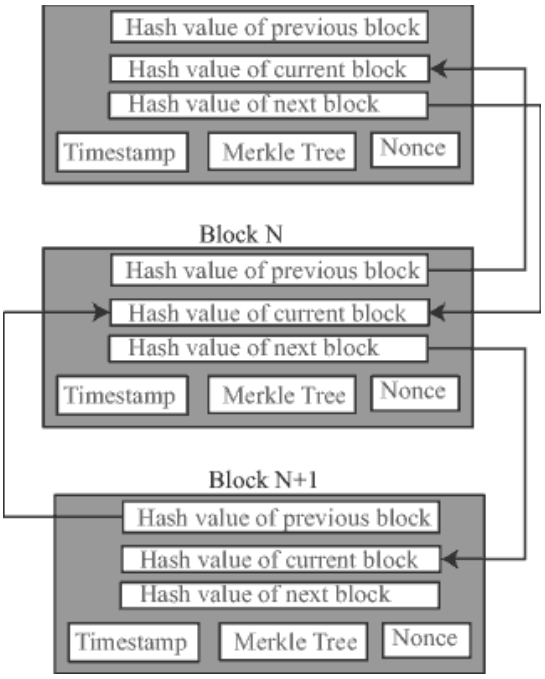

Figura 7. Estrutura dos blocos do blockchain. Fonte: [Patgiri, Acharjamayum e Devi, 2019]

Küfner (2018) menciona o problema de redes peer-to-peer como o blockchain em coordenar a concordância dos registros, bem como adicionar novos, já que não possui autoridade central, existindo a possibilidade de falta ou falsificação de comunicação dos nós, de modo acidental ou deliberado. Para esta situação é usada a analogia do Problema dos Generais Bizantinos, os quais estão isolados em torno de uma cidade inimiga e se comunicam por mensagens. Para vencer a batalha, precisam coordenar suas ações em conjunto, atacando ou fugindo todos ao mesmo tempo, sem discordâncias. Caso o exército não entre nesse consenso, estará dividido em alguns que vão atacar, outros fugirão e o restante permanecerá sem ação, por não considerar este o

momento do ataque, resultando em derrota. O ponto fundamental do problema é o fato das mensagens não serem confiáveis, sendo suscetíveis a erros e falsificações, devendo haver um mecanismo que resolva o problema da não confiabilidade das mensagens.

Patgiri, Acharjamayum e Devi (2019) especificam que o blockchain possui processos para adicionar as transações confirmadas, de modo que todos os nós concordem com a ordem das entradas que são anexadas, resolvendo o problema dos Generais Bizantinos. Cada participante da blockchain possui uma chave pública e uma chave privada, e as transações são registradas com a assinatura digital de seu usuário, de forma a garantir autenticidade e integridade. O endereço de cada usuário é uma chave pública, utilizando sua chave privada para realizar transações.

3. Trabalhos Relacionados

Nesta seção, com o objetivo de focar no tema deste trabalho, serão apresentados alguns trabalhos relacionados com autenticação em sistemas distribuídos e a tecnologia Blockchain.

3.1. Kerberos

De acordo com Courouris, Dollimore, Kindberg e Blair (2013), Kerberos foi desenvolvido nos anos 80, para uso no Projeto Athena no Massachusetts Institute of Technology (MIT), visando oferecer diversos recursos de autenticação e segurança para um campus com milhares de estações de trabalho. Passou por várias versões e aprimoramentos, sendo muito utilizado por empresas e universidades. Baseado no uso de um sistema criptográfico simétrico, possui um servidor de autenticação conhecido como KDC (*Key Distribution Center* - Centro de Distribuição de Chaves), com o banco de dados de autenticação das aplicações. O KDC possui dois serviços: um para autenticação e um para conceder tíquetes. Cada cliente deve primeiramente se autenticar com o serviço de autenticação, para então solicitar um tíquete para comunicação com o servidor de serviço pretendido. Cada tíquete será apresentado ao respectivo servidor de destino, verificando que o remetente foi autenticado pelo servidor Kerberos, possuindo o nome do cliente, *timestamp* e uma chave de sessão para criptografar a comunicação. O *timestamp* dos tíquetes serve como tempo de expiração de sessão, também inibindo eventuais duplicações. O funcionamento do sistema Kerberos é exemplificado na Figura 8.

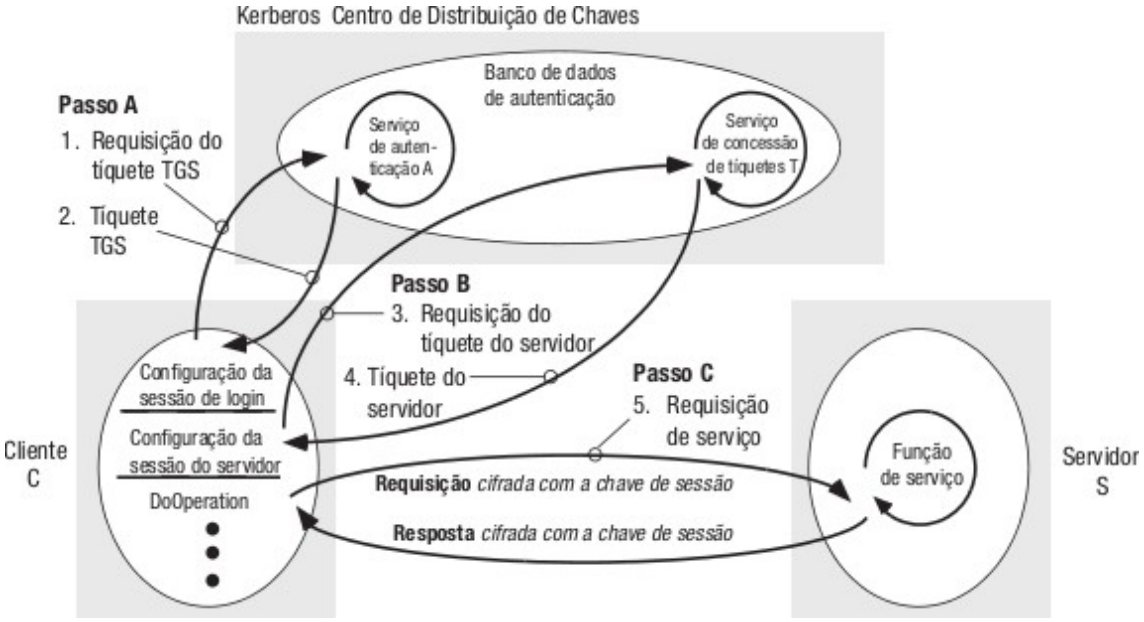

Figura 8. Distribuição de informações com sobreposição de roteamento. Fonte: [Courouris, Dollimore, Kindberg e Blair, 2013]

A primeira mensagem é enviada para o serviço de autenticação do KDC, possuindo nome do cliente, *timestamp* e o nonce que será utilizado para verificar a validade da resposta. A segunda mensagem possui o tíquete solicitado pelo cliente, o qual somente poderá ser obtido pela própria senha do cliente. Com o tíquete válido obtido pelo servidor KDC, o cliente poderá usá-lo quantas vezes quiser, até que expire, podendo solicitar novamente outro tíquete para o mesmo serviço, ou mais tíquetes para se comunicar com outros servidores.

3.2. SPX

Woo e Lam (1999) mencionam o SPX como outro serviço de autenticação destinado para redes abertas, com funcionalidades semelhantes ao Kerberos, tendo como principal diferença ser baseado em sistema criptográfico assimétrico de chave pública/privada. Possui dois serviços: LEAF (Login Enrollment Agent Facility - Agente Facilitador de Login) e CDC (Certificate Distribution Center - Centro de Distribuição de Certificados), equivalentes aos dois serviços do Kerberos. O LEAF atua na inicialização da credencial e o CDC como repositório das chaves públicas para entidades e autoridades certificadoras (AC), e privadas das entidades. As ACs podem ser organizadas hierarquicamente.

Tudo é armazenado criptografado, de forma que o CDC não precisa usar um protocolo de comunicação confiável. Sua função é emitir certificados de chaves públicas (nomes de ligação e chaves públicas de entidades). Confiança global não é necessária no SPX, pois cada AC normalmente tem jurisdição sobre apenas um subconjunto de todos os principais P, enquanto cada P confia apenas em um subconjunto de todas as ACs, referidas como autoridades confiáveis de P. A escalabilidade do sistema é bastante aprimorada pela ausência de confiança global e componentes confiáveis.

3.3. Autenticação Blockchain de aplicação de rede

Mohsin et al. (2018) fizeram uma revisão sistemática sobre sistemas de autenticação utilizando a tecnologia blockchain. O trabalho consistiu na realização de revisões de literatura, abordando e comparando variados tipos de sistemas que utilizam a tecnologia blockchain em suas funcionalidades, inclusive autenticações. Cada sistema possui características e necessidades diferentes: *smartphones*, medidores de eletricidade, redes veiculares, cidades inteligentes, dispositivos médicos, redes de satélites, redes sociais, setor financeiro, entre outros. Além de ser adequada pela descentralização de dispositivos e usuários, a tecnologia blockchain se mostrou facilmente implementável e adaptável para os diversos sistemas, contribuindo com suas vantagens características em todos os sistemas, dentre as principais: armazenando o banco de dado distribuído com alto grau de confiabilidade e transparência, ao mesmo tempo que é seguro contra adulterações pela natureza imutável de seus dados, oferecendo um certo grau de privacidade pelo uso de identidades digitais, e reduzindo custos e complexidade de infraestrutura de TI (Tecnologia da Informação) centralizada.

3.4. Protocolos de autenticação usando blockchain para a Internet das Coisas

Zhong et al. (2020) apontam que na crescente abrangência da Internet das Coisas (*Internet of Things*, IoT), em que as aplicações se comunicam e autoconfiguram em rede sem auxílio humano, surgiu a *Smart Grid*, com a finalidade de ser uma rede da IoT eficiente no uso de energia. Porém, na maioria das *smart grids* o protocolo utilizado é centralizado, sendo assim um considerável risco de segurança. Foi proposto um protocolo de autenticação descentralizado para redes *smart grid* baseado em blockchain, de forma a usar contratos inteligentes, que são algoritmos executáveis armazenados como transação na blockchain, que realizam funcionalidades quando determinadas condições são atendidas, podendo ser copiados e executados nos dispositivos que desejam utilizar tais contratos. Os contratos inteligentes do experimento foram definidos para estabelecer critérios de comunicação. Foram discutidas questões de segurança de alguns protocolos centralizados e como o modelo em blockchain proposto resolveria tais problemas. O protocolo foi implementado em uma plataforma de blockchain de código aberto FISCO BCOS (https://fisco-bcosdocumentation.readthedocs.io/en/latest/), aplicando algoritmos de contratos inteligentes. Foram realizados testes com as vulnerabilidades dos modelos centralizados que foram discutidas e o protocolo demonstrou ser seguro e eficiente.

Yavari et al. (2021) também abordaram a autenticação de aplicações na Internet das Coisas usando blockchain com a criação de um protocolo usando a linguagem de programação JavaScript e uma rede local de blockchain Ethereum criada com a ferramenta TestRPC (https://www.npmjs.com/package/ethereumjs-testrpc/). Foi criado um contrato inteligente, inicializada a blockchain e realizados alguns testes de segurança com a ferramenta Scyther (https://people.cispa.io/cas.cremers/scyther). O protocolo implementado foi considerado uma boa solução para a Internet das Coisas e seguro no escopo dos testes realizados.

4. Ambiente distribuído simulado

Esta seção trata a implementação de um ambiente de simulação para um sistema distribuído integrado com uma blockchain, podendo ser adicionadas aplicações e definidas regras de comunicação e funcionalidades para estas. A implementação permite

a configuração de várias aplicações para um sistema distribuído, de forma que possam autenticar explorar a blockchain para autenticar suas comunicações e armazenar dados de suas funcionalidades.

4.1. Base para o sistema distribuído com blockchain

O projeto deste trabalho foi desenvolvido no ambiente de programação Java na versão 8, acrescido da biblioteca Bouncy Castle (https://www.bouncycastle.org), que disponibiliza a função criptográfica ECDSA para uso de chaves pública/privada com assinaturas e verificações. Para as demais funcionalidades, foram usadas bibliotecas nativas da linguagem.

O sistema blockchain implementado é composto pela entidade *Usuário* e as estruturas de dados *Registro* e *Bloco*:

- O *Usuário* é a entidade atuante do sistema, que pode adicionar *Blocos* e *Registros* na blockchain, possuindo seu par de chaves pública/privada, podendo gerar assinaturas com sua chave privada e ser reconhecido pelas outras entidades por sua chave pública;
- O *Registro* é uma unidade de informação adicionada por uma aplicação da rede, equivalente a uma transação em um modelo de blockchain genérico, contendo: autor, dados, *timestamp* e assinatura, sendo:
  - autor a chave pública da aplicação que criou o *Registro*;
  - dados as informações armazenadas;
  - *timestamp* a referência temporal;
  - assinatura a hash do *Registro* criptografada com a chave privada do *Usuário* que o criou.
- O *Bloco* é a estrutura que armazena uma determinada quantidade de *Registros* no sistema blockchain, sendo assim persistidos e inalteráveis. Um *Bloco* contém em seu cabeçalho: autor, hash do *Bloco* anterior, timestamp, nonce, e assinatura.

O *Registro* deve ser assinado por seu autor e seu valor de hash será gerado aplicando o algoritmo SHA-256 em seus valores armazenados. Para facilitar o experimento desta implementação, serão aceitos como persistidos no sistema todos os *Registros* validados e adicionados, independente de já estarem em um *Bloco* adicionado na blockchain. Tais *Registros* estarão em um *Bloco* ainda não adicionado que será chamado de *Cache*. O passo a passo para adicionar um *Registro* no *Cache* está ilustrado na Figura 9.

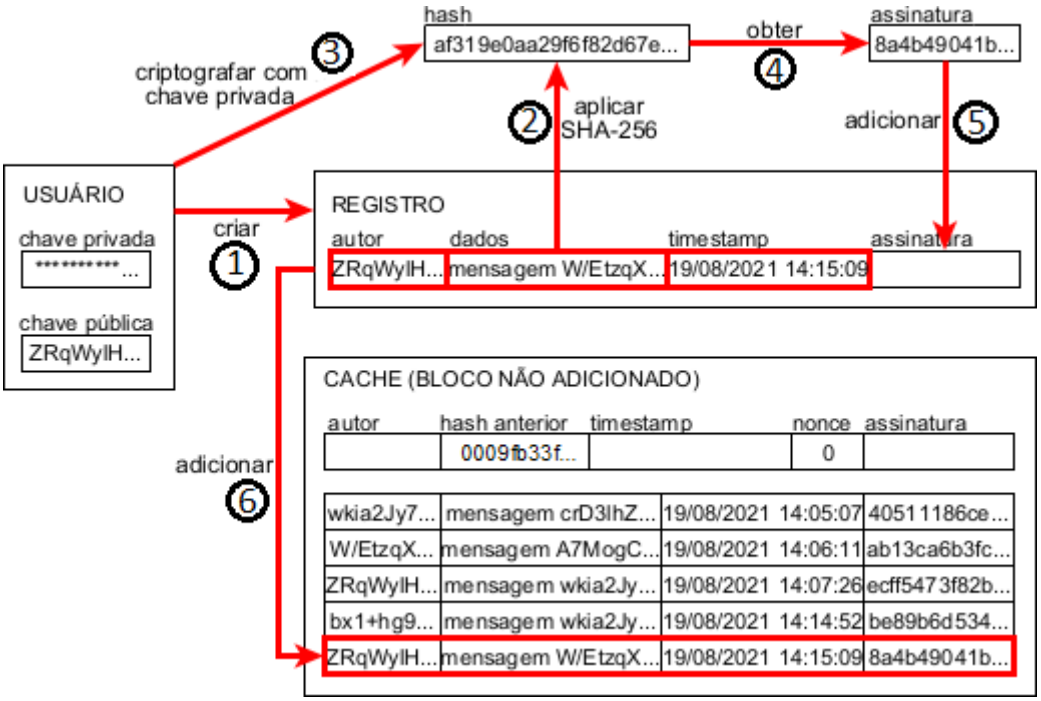

Figura 9. Passo a passo para adicionar um Registro no Cache: 1) O Usuário cria o Registro e adiciona sua autoria, as informações e timestamp; 2) É aplicado o algoritmo SHA-256 em todas as informações do Registro para obter o valor de hash; 3) O Usuário aplica o algoritmo de criptografia com sua chave privada no hash obtido; 4) Esse procedimento resulta na assinatura; 5) A assinatura é adicionada no Registro; 6) O Registro é adicionado no Cache. Fonte: o autor

Poderá ser obtida a Merkle de um *Bloco* aplicando o algoritmo gerador desta função em seus *Registros*, e o valor de hash aplicando o algoritmo SHA-256 em seu cabeçalho acrescido da respectiva Merkle. Para cada *Bloco* a ser adicionado na blockchain deverá ser realizada uma prova de trabalho, que será comprovada com a hash do *Bloco* sendo iniciada por uma determinada quantidade de zeros, definido como desafio. Para concluir tal desafio, a aplicação que for adicionar o *Bloco* inicia o nonce com zero e verifica a hash sucessivamente, incrementando o valor de nonce a cada tentativa sem êxito. Após cumprir o desafio, o *Bloco* será assinado pelo autor e persistido na blockchain. Para que sejam adicionados novos *Blocos* após poucas iterações, de forma que o sistema não acumule muitos *Registros* no *Cache*, o sistema está definido para que o *Cache* seja adicionado quando lhe for acrescentado apenas 5 *Registros*, tornando-se um novo *Bloco* a ser acrescentado na blockchain. O *Usuário* que acrescentar o quinto *Registro* fará o processo para cumprir a prova de trabalho e adicionar o novo *Bloco*. Quando o *Bloco* adicionado for reconhecido pela blockchain, será criado um novo *Cache* vazio. O passo a passo para adicionar um *Bloco* na blockchain está ilustrado na Figura 10.

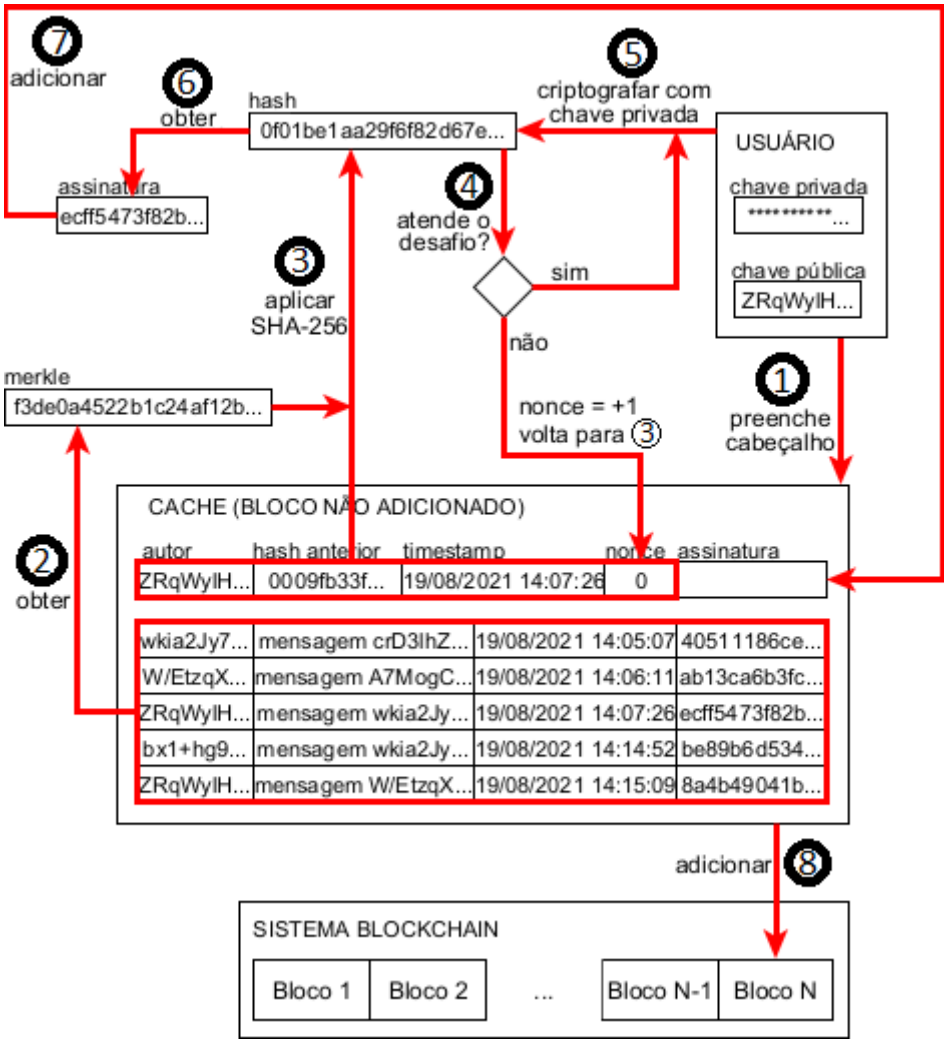

Figura 10. Passo a passo para adicionar um Bloco na blockchain: 1) O Usuário adiciona sua autoria e o timestamp no cabeçalho; 2) É aplicado o algoritmo gerador de árvore Merkle nos Registros do Bloco; 3) Aplica-se o algoritmo SHA-256 no cabeçalho acrescido da árvore Merkle, obtendo o valor de hash do Bloco; 4) É verificado se o hash atende ao desafio, e caso seja negativo, o valor de nonce é acrescido de 1 e a execução volta para o passo 3; 5) O Usuário aplica o algoritmo de criptografia com sua chave privada no hash obtido; 6) Esse procedimento resulta na assinatura; 7) A assinatura é adicionada no Bloco; 8) O Bloco é adicionado na blockchain. Fonte: o autor

As aplicações do sistema distribuído foram implementadas localmente na mesma máquina, com o endereço IP localhost 127.0.0.1 em comum para todas, sendo as comunicações referenciadas apenas pelos seus números de porta. Dessa forma, para fins de explicação do experimento, o endereço IP não será mencionado no presente trabalho.

Foram criados *Grupos* para a classificação de aplicações, sendo que cada uma destas pertencerá a um único *Grupo*; e *Regras* para definir cada relação origem e destinatário de comunicação entre os *Grupos*. Todo acesso entre estas não será permitido caso não tenha uma *Regra* definida do *Grupo* de origem para o *Grupo* de destinatário. A relação entre *Grupos* e *Regras* de acesso é exemplificada na Figura 11.

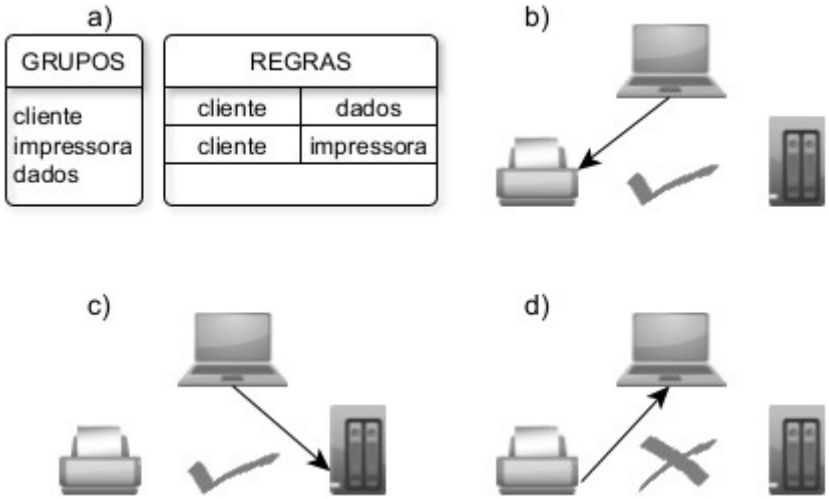

Figura 11. Grupos e Regras: a) definições; b) acesso de cliente para impressora permitido; c) acesso de cliente para dados permitido; d) acesso de impressora para cliente não permitido. Fonte: o autor

A entidade *Aplicação* estende a *Usuário*, herdando todas as suas características, acrescida dos atributos: número de porta, *Grupo* e dados. Cada *Aplicação* possui uma thread com um socket UDP aberto em seu número de porta para receber a comunicação de outras *Aplicações*, e também poderá enviar mensagens para estas. O protocolo UDP foi escolhido pelas *Aplicações* serem executadas localmente, sem a necessidade de implementar segurança e confiabilidade para as mensagens do experimento. Uma mensagem enviada deve conter chave pública e assinatura da aplicação emissora, permitindo assim que a *Aplicação* destinatária possa autenticar que a mensagem pertence a uma *Aplicação* legítima do sistema. Cada *Aplicação* que receber uma mensagem adiciona um *Registro* com o status da comunicação e/ou dados específicos para a operação solicitada.

Para simplificar os testes e permitir que todo o sistema funcione em um único executável, foi criada a entidade *Central*, que possui todos os *Grupos*, *Regras* e *Aplicações* no sistema, e o blockchain. A *Central* também possui a predefinição de todo o arranjo de *Grupos*, *Regras* e *Aplicações* a serem instanciados no início do funcionamento do sistema. Como a finalidade do presente é analisar apenas as características do sistema distribuído em relação ao blockchain, e para fins de abstração de complexidade, o sistema blockchain não será implementado de forma distribuída, mas em uma única estrutura de dados presente na *Central*, que faz a intermediação do sistema como se fosse o consenso, permitindo que *Aplicações* visualizem e adicionem *Registros* e *Blocos* no sistema blockchain. O modelo conceitual de todas as entidades que compõem o sistema estão dispostos na Figura 12.

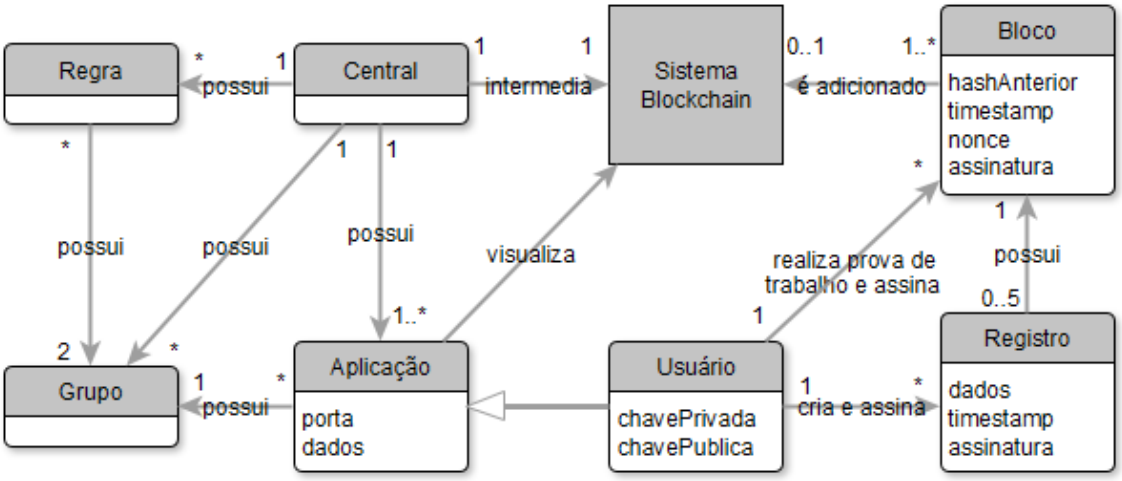

Figura 12. Modelo conceitual das entidades do sistema. Fonte: o autor

Grupos, *Regras* e *Aplicações* em execução com suas características serão exibidos graficamente em um painel JPanel. Os *Blocos* adicionados e o *Cache* serão exibidos em um painel de texto JTextPane.

4.2. Aplicações no sistema distribuído

Grupos, *Regras* e *Aplicações* foram definidos nas predefinições da *Central*, criando grupos de *Aplicações* com funcionalidades diversificadas, de modo a possibilitar testes sobre algumas características gerais do sistema, com algumas *Aplicações* aproveitando recursos da blockchain em funcionalidades específicas. Tais funcionalidades utilizam informações do atributo dados das *Aplicações*. Os números de portas para as *Aplicações* são gerados em ordem crescente, conforme são adicionadas no sistema, iniciando o número em 1230 para a primeira *Aplicação*.

Foi definido um grupo chamado admin e uma única *Aplicação* fará parte deste. Tal *Aplicação* não fará nenhuma comunicação com as demais, sendo a sua única finalidade gerir os *Grupos*, *Regras*, podendo adicionar, alterar ou excluir. O admin também pode alterar as informações contidas no atributo dados de *Aplicações*. Toda operação do admin será adicionada em um *Registro*, que será assinado e incluído na blockchain. Com o *Registro* tendo a autoria da aplicação admin autenticada, sua operação será aplicada no sistema. Seguem os arranjos de aplicações no sistema:

- O grupo mais genérico de comunicações é o chat, que possui uma única *Regra* que permite apenas comunicações entre o mesmo grupo. Para este, foram adicionadas três *Aplicações* no sistema;
- Os *Grupos* cliente, impressora e dados foram criados, com *Regras* permitindo acesso do *Grupo* cliente para impressora e dados. Além das *Regras* de acesso do próprio sistema, este arranjo de aplicações possui uma funcionalidade específica, definindo que cada *Aplicação* do *Grupo* cliente deverá ter permissão de acesso para os *Grupos* impressora e/ou dados, sendo tal permissão armazenada em seu atributo dados. Foram adicionadas duas *Aplicações* cliente, uma com permissão para acessar impressora e outra para acessar dados, e também uma *Aplicação* impressora e outra dados;
- Um último arranjo de aplicações é composto pelos *Grupos* intermediário,

comprador e vendedor, com apenas duas *Regras* para permitir que o *Grupo* comprador acesse os grupos intermediário ou vendedor. Esta combinação de aplicações compõem um sistema de compra e venda, em que o comprador precisa de autorização de compra previamente fornecida por um intermediário. O comprador possui em seus dados: saldo em dinheiro, quantidade de produto comprado e valor autorizado para compra; o vendedor possui saldo em dinheiro e quantidade de produto disponível; e o intermediário possui apenas o valor máximo que pode autorizar. Para obter uma autorização de compra, o comprador envia uma mensagem de solicitação com o valor desejado a um intermediário. O intermediário verifica se o valor é menor ou igual ao que pode autorizar e se o comprador o possui em seu saldo, para então conceder a autorização, que será adicionado nos dados do comprador. Após obter autorização, o comprador deve solicitar a compra com um vendedor. O vendedor verifica se o comprador possui autorização, se o valor solicitado é igual ao solicitado e se possui a quantidade de produtos disponíveis para a venda, para então concluir a transação, atualizando os dados de comprador e vendedor. Para fins de simplificação que não excedam o escopo do experimento, não serão utilizados rótulos para o tipo de moeda ou produto vendido, sendo também considerado que cada unidade de produto custe uma unidade de moeda.

A interface final com todas as configurações, aplicações, blocos e cache do sistema exibidos graficamente está ilustrada na Figura 13.

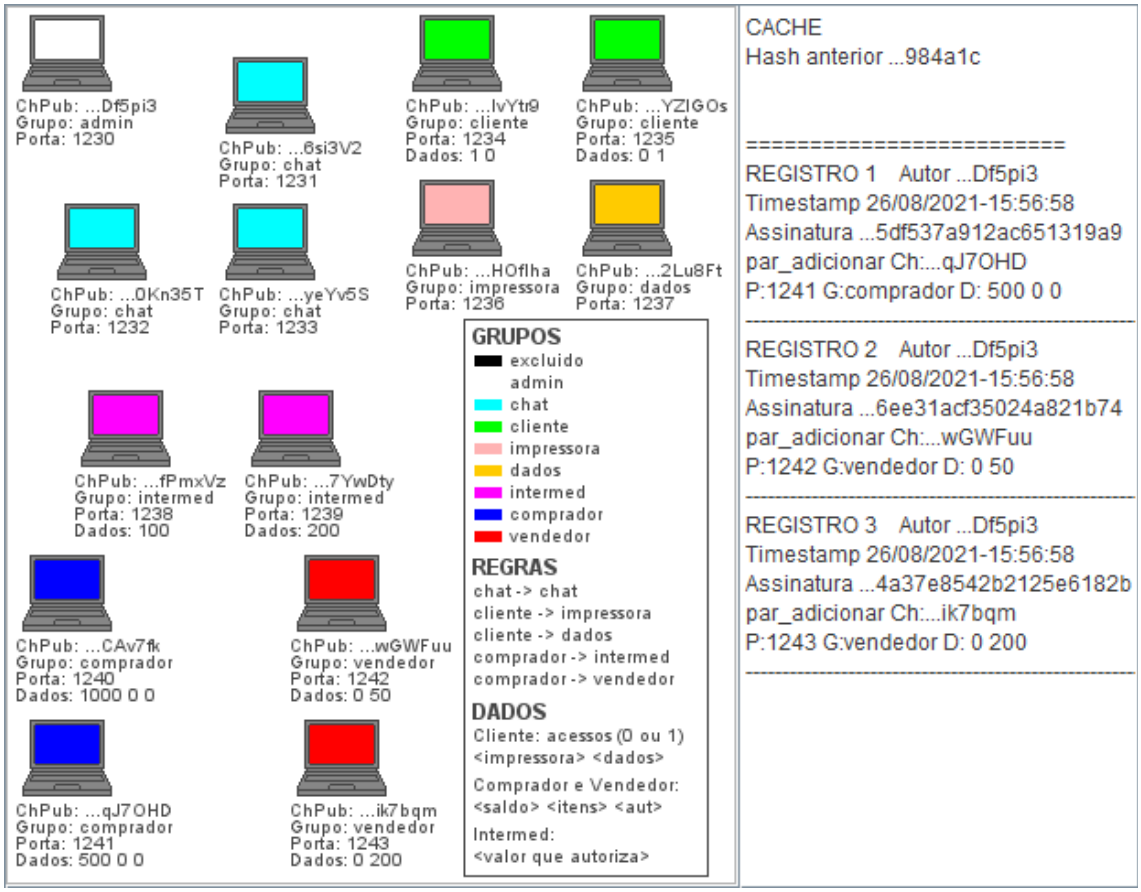

Figura 13. Interface com a disposição do sistema. Fonte: o autor

4.3. Experimentos e resultados

Foram realizadas várias comunicações envolvendo os três arranjos de aplicações, de forma a explorar a autenticação de mensagens no sistema distribuído, o qual deverá seguir os critérios de comunicação estabelecidos na blockchain. As comunicações foram realizadas de forma variada, seguindo os critérios de comunicação ou violando algum destes, com a finalidade de verificar se o sistema distribuído trata as comunicações válidas ou inválidas adequadamente, conforme esperado.

Para o arranjo de 3 aplicações do grupo chat, as comunicações devidamente assinadas entre estas eram aceitas normalmente, de acordo com a regra de permissão de acesso. Para as aplicações de outros grupos, a comunicação assinada era negada, cumprindo a regra de acesso. Para as aplicações que foram excluídas do sistema, independente do grupo que pertenciam, bem como para as aplicações legítimas que não assinaram a mensagem e/ou que a assinatura não pudesse ser verificada, tiveram a comunicação negada, não sendo possível reconhecê-las no sistema. Na Figura 14 são exibidas as características deste arranjo.

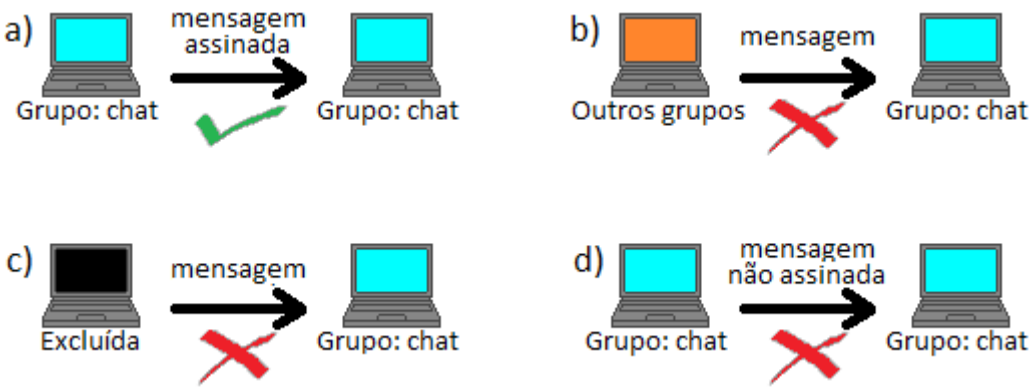

Figura 14. Comunicações para o grupo chat: a) Uma aplicação do grupo chat envia mensagem assinada para outra aplicação do grupo chat e a mensagem é aceita; b) Aplicação de outros grupos enviam mensagens para uma aplicação do grupo chat, que é negada, independente da assinatura ser válida; c) Uma aplicação que foi excluída do sistema envia mensagem para uma aplicação do grupo chat, que é negada, independente da assinatura ser válida; d) Uma aplicação do grupo chat envia uma mensagem não assinada para outra aplicação do grupo chat, que é negada. Fonte: o autor

Para o segundo arranjo composto por 2 aplicações cliente, 1 impressora e 1 dados, os testes aplicados no arranjo anterior obtiveram os mesmos resultados para comunicações devidamente assinadas mas não respaldadas pelas regras de acesso, bem como para aplicações excluídas do sistema e/ou com problemas na assinatura da mensagem. Embora a regra de acesso do sistema permita para a aplicação cliente acessar impressora ou dados, a funcionalidade específica para esse arranjo define nos dados de cada cliente se o mesmo pode acessar impressora e/ou dados. Dessa forma, as aplicações cliente somente podiam acessar impressora e/ou dados se isso estivesse permitido em seus dados. O arranjo de comunicações do grupo cliente para os grupos impressora e dados está ilustrado na Figura 15.

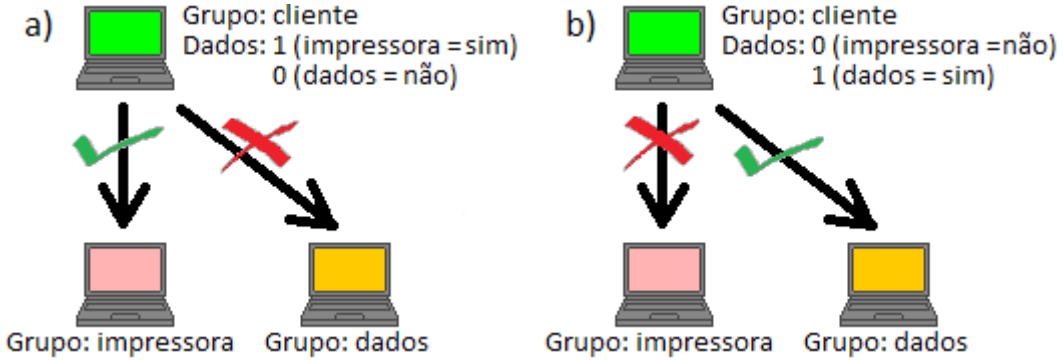

Figura 15. Comunicações do grupo cliente para os grupos impressora e dados: a) A aplicação do grupo cliente está definida para se comunicar apenas com aplicações do grupo impressora; b) A aplicação do grupo cliente está definida para se comunicar apenas com aplicações do grupo dados. Fonte: o autor

Para o terceiro arranjo composto por 2 aplicações comprador, 2 intermediário e 2 vendedor, os testes aplicados no arranjo anterior obtiveram os mesmos resultados para comunicações devidamente assinadas mas não respaldadas pelas regras de acesso, bem como para aplicações excluídas do sistema e/ou com problemas na assinatura da mensagem. As regras de acesso do sistema permitem para a aplicação comprador

acessar intermediário ou vendedor, aceitando tais comunicações devidamente assinadas. Porém, as funcionalidades específicas do arranjo somente permitiram que o comprador realizasse compras com o vendedor com a devida autorização obtida do intermediário, mediante ter atendido às regras de negócio específicas.

Para realizar uma compra, a aplicação compradora deverá enviar uma mensagem de solicitação com o valor pretendido para uma aplicação intermediária. A aplicação intermediária recebe a solicitação de compra, verifica se a mensagem pertence a uma aplicação compradora, se o valor solicitado está dentro do valor que ela pode autorizar e se a compradora o possui em seu saldo. Com os critérios da solicitação atendidos, a aplicação intermediária adiciona a autorização de compra para a compradora em um registro na blockchain, sendo assim o valor de compra autorizado adicionado nos dados da aplicação compradora. Após obter a autorização, a aplicação compradora pode enviar a solicitação de compra com o valor pretendido para uma aplicação vendedora. A aplicação vendedora recebe a solicitação de compra, verifica se a mensagem pertence a uma aplicação compradora, se a compradora possui autorização de compra no valor solicitado e se possui a quantidade de produto relativa ao valor de compra. Caso os critérios da solicitação de compra sejam atendidos, a aplicação compradora adiciona a compra na blockchain, sendo os valores da transação atualizados nos dados das aplicações compradora e vendedora. O passo a passo para realizar uma transação completa neste arranjo está ilustrado na Figura 16.

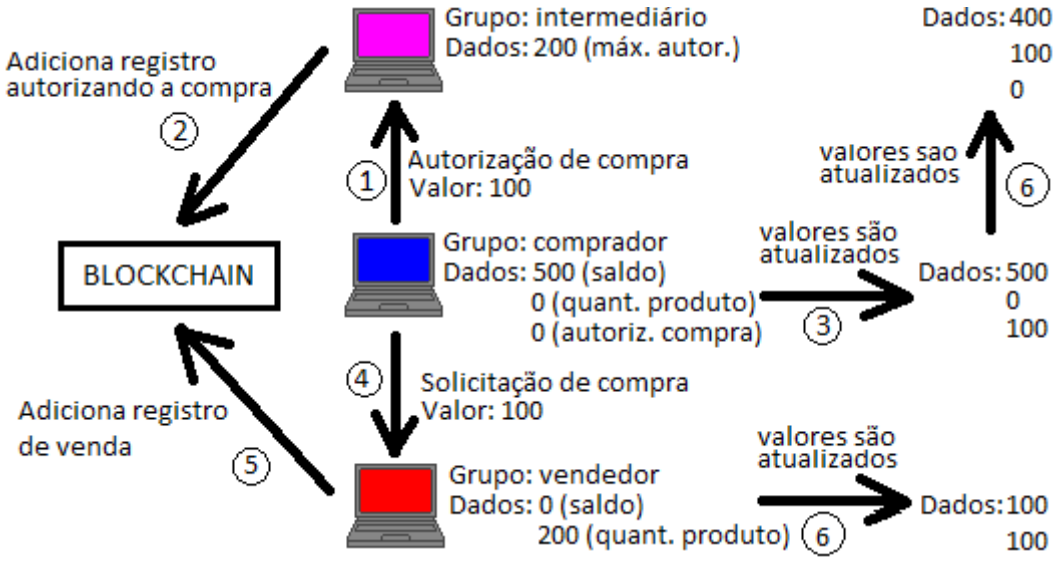

Figura 16. Passo a passo para a transação completa do arranjo: 1) A aplicação compradora solicita autorização de compra no valor de 100 para a aplicação intermediária; 2) A aplicação intermediária adiciona um registro com a autorização; 3) O valor autorizado é atualizado para a aplicação compradora; 4) A aplicação compradora solicita a compra para a aplicação vendedora; 5) A aplicação vendedora adiciona um registro de venda no valor solicitado; 6) A autorização é descontada da compradora e os valores de saldo e quantidade de produto são atualizados para a compradora e a vendedora. Fonte: o autor

Todas as aplicações usaram a blockchain para visualizar e armazenar dados de suas comunicações, servindo como log de segurança e armazenamento de dados para suas funcionalidades. As aplicações aproveitaram o sistema de criptografia de chaves pública/privada e assinaturas característicos da blockchain para autenticar suas ações no sistema. Embora tenha sido simulada em uma única instância no experimento, a blockchain em um sistema real é descentralizada, e como tal, mostrou-se uma alternativa adequada ao modelo de autenticação que é realizado por terceiros.

5. Conclusão

As aplicações distribuídas têm sido amplamente utilizadas para disponibilizar variados serviços por meio da internet. Proporcionalmente à sua popularização e a sensibilidade das informações tratadas, surgem preocupações de segurança, tornando-se necessários estudos e pesquisas contínuos para acrescentar melhorias e proteger o sistema de falhas e ações maliciosas. A proposta deste trabalho foi empregar a tecnologia blockchain para realizar a autenticação de aplicações em um sistema distribuído.

Foi criada uma simulação de um sistema distribuído com várias aplicações que possuem funcionalidades diversas e se comunicam, integrado com uma estrutura blockchain. Nela, estavam adicionadas todas as aplicações com suas chaves públicas e dados, sendo categorizadas por grupos e estabelecidas algumas regras para permissão de comunicação, e também adicionadas algumas funcionalidades específicas para alguns arranjos de aplicações. Todas as aplicações puderam usar os recursos da blockchain para autenticar as comunicações e armazenar dados para suas funcionalidades.

Trabalhos futuros a este podem explorar sistemas distribuídos complexos e remotos, utilizando uma implementação completa da blockchain, de modo que cada aplicação tenha uma instância desta de forma sincronizada e participante. Sugere-se também que a blockchain possa ser implementada como um sistema de gerenciamento de banco de dados para as aplicações do sistema distribuído.

Referências

COULOURIS, George; DOLLIMORE, Jean; KINDBERG, Tim; BLAIR, Gordon. Sistemas Distribuídos: Conceitos e Projeto. 5ª ed. Bookman, Porto Alegre, 2013.

DEITEL, H. M.; CORNELL, G. Java: Como Programar, 10a. Edição. 2016.

DRESCHER, Daniel. Blockchain Básico. Novatec, São Paulo, 2018.

GOODRICH, Michael T.; TAMASSIA, Roberto. Introdução à Segurança de Computadores. Bookman, Porto Alegre, 2012.

KÜFNER, Robert A. The Byzantine Generals Problem. Medium.com/nakamo-to, 2018. https://medium.com/nakamo-to/the-byzantine-generals-problem-1ae994eaba7e, setembro de 2021.

MOHSIN, Ali Hadi; ZAIDAN, A. A.; ZAIDAN, Bilal Bahaa; ALBAHRI, O. S.; ALBAHRI, A. A.; ALSALEM, M. A.; MOHAMMED, K. I. Blockchain Authentication of Network Applications: Taxonomy, Classification, Capabilities, Open Challenges, Motivations, Recommendations and Future Directions. ResearchGate, 2018. https://www.researchgate.net/publication/330032241_Blockchain_Authentication_of_Network_Applications_Taxonomy_Classification_Capabilities_Open_Challenges_Motivations_Recommendations_and_Future_Directions, setembro de 2021.

NAKAMOTO, Satoshi. Bitcoin: A Peer-to-Peer Electronic Cash System. 2008. https://bitcoin.org/bitcoin.pdf, julho de 2021.

PATGIRI, Ripon; ACHARJAMAYUM, Irani; DEVI, Dhruwajita. Blockchain: A Tale of Peer to Peer Security. ResearchGate, 2019. https://www.researchgate.net/publication/330883530_Blockchain_A_Tale_of_Peer_to_Peer_Security, setembro de 2021.

STALLINGS, William. Criptografia e segurança de redes: Princípios e práticas. 4ª ed. Pearson Prentice Hall, São Paulo, 2008.

TANENBAUM, Andrew S; VAN STEEN, Maarten. Sistemas Distribuídos: Princípios e Paradigmas. 2ª ed. Pearson Prentice Hall, São Paulo, 2007.

WOO, Thomas Y. C., LAM, Simon S. Authentication for Distributed Systems. ResearchGate, 1999. https://www.cs.utexas.edu/users/lam/Vita/Bpapers/WooLam98b.pdf, agosto de 2021.

YAVARI, Mostafa; SAFKHANI, Masoumeh; KUMARI, Saru; KUMAR, Sachin; CHEN, Chien-Ming. An Improved Blockchain-Based Authentication Protocol for IoT Network Management. Hindawi, 2021. https://www.hindawi.com/journals/scn/2020/8836214/, setembro de 2021.

ZHONG, Yuxin; ZHOU, Mi; LI, Jiangnan; LIU, Yan; ZHAO, Yun; CHEN, Jiahui; HU, Muchuang. Distributed Blockchain-Based Authentication and Authorization Protocol for Smart Grid. Hindawi, 2020. https://www.hindawi.com/journals/wcmc/2021/5560621/, setembro de 2021.