Auditoria de aplicações de *Big Data* usando *Hashes* de Similaridade e *Blockchains*

Carlos A. R. Oliveira
Inmetro
Duque de Caxias, Brasil
aroliveira@colaborador.inmetro.gov.br

Paulo Assumpção
PPGI/UFRJ
Rio de Janeiro, Brasil
passumpcao@ufrj.br

Pablo Ortiz, Wilson Melo, Luiz Carmo
Inmetro
Duque de Caxias, Brasil
{phortiz, wsjunior, lfrust}@inmetro.gov.br

***Abstract*—Com a expansão das aplicações de *BigData*, garantir a segurança e confiabilidade dos dados armazenados tornou-se uma tarefa desafiadora. Tal desafio é especialmente preocupante quando se considera o monitoramento de infraestruturas críticas, em especial aquelas que contemplam ativos físicos monitorados por sensores e dispositivos IoT de coleta de dados. Uma alternativa consiste no uso de *blockchains* como mecanismo de auditoria de aplicações *Big Data* a partir da técnica *off-chain*, onde os pacotes de dados brutos são armazenados em um sistema de banco de dados convencional e apenas um resumo criptográfico dos dados é escrito no *blockchain*. Embora bastante aplicada na literatura recente sobre o tema, essa estratégia não permite a auditoria de dados em cenários de perda parcial de informações, onde pacotes de dados correspondentes a subconjuntos do pacote original precisam ser verificados. Este artigo propõe uma estratégia de auditoria de dados em aplicações *Big Data* que emprega *hashes* de similaridade para estender as funcionalidades do modelo *off-chain*. Usado em conjunto com resumos criptográficos e *smart contracts*, tais *hashes* permitem auditar pacotes de dados distintos em situações de possível perda parcial, possibilitando diferenciar uma situação não intencional de uma tentativa deliberada de fraude. Em nossos experimentos, avaliamos os algoritmos Minhash e Simhash, apontando resultados computacionais que indicam que o Minhash é bastante promissor para esse tipo de aplicação, e pode contribuir significativamente para a robustez de processos de auditoria em aplicações *Big Data*.**

***Index Terms*—*Big Data*, auditoria, integridade, *blockchain*, *hash* de similaridade, LSH.**

I. Introdução

O recente número de desastres naturais e eventos catastróficos causados pelas atividades humanas fez surgir uma consciência global a respeito da importância das infraestruturas críticas (ICs), em especial aquelas que envolvem ativos físicos, como por exemplo plantas industriais, usinas de energia, refinarias, rodovias, e centrais de telecomunicações. Em 2015, o rompimento de uma barragem liberou mais de 60 milhões de metros cúbicos de resíduos de minério de ferro sobre a cidade de Mariana, MG. O desastre deixou 19 mortos e mais de 600 desabrigados [14], [22]. Três anos depois, um novo desastre em Brumadinho teve consequências muito mais graves, com mais de 248 óbitos confirmados e impactos sobre uma população de mais de um milhão de pessoas devido aos rejeitos de minérios que fluíram pelo Rio Doce [17]. Os eventos que se seguiram mostraram que os dados dos sistemas de monitoramento da barragem não estavam disponíveis para investigação. Mesmo empresas bem conceituadas, responsáveis pela auditoria desses sistemas, foram envolvidas em suspeitas de displicência e corrupção [12].

Eventos como os descritos acima mostram a necessidade de soluções que possam monitorar ICs de forma segura e, principalmente, garantir a confiabilidade e disponibilidade dos dados coletados. Sistemas de monitoramento de ICs caracterizam-se pela geração de um volume elevado de dados oriundos de diversos sensores e dispositivos de coleta IoT, caracterizando um cenário desafiador de implementação de *Big Data* [5]. Entretanto, os meios tradicionais de gerenciamento de dados geralmente não asseguram a integridade e procedência das informações. Uma alternativa é a busca de soluções descentralizadas, com propriedades intrínsecas de segurança cibernética, para contemplar os requisitos emergentes de ICs [28], [31]. Vance e Vance [23] concluíram em sua pesquisa que *blockchains* constituem uma tecnologia promissora para proteção de ICs. Embora o *blockchain* adicione propriedades essenciais como confiança e rastreabilidade, o problema da escalabilidade ainda dificulta seu uso em aplicações reais. Um dos aspectos que limitam a escalabilidade de uma rede *blockchain* é a impossibilidade dos protocolos de consenso de gerar novos blocos de forma proporcional à demanda e o tamanho do *payload* das transações, o que afeta diretamente a taxa de vazão ou *throughput* (medido em transações por segundo) [6], [11], [30]. Com a intenção de mitigar esse efeito, diversos trabalhos colocam em prática a técnica *off-chain* [11], [21], [26], [30], como proposta por Esposito et al. [6], que consiste em manter os dados armazenados fora do *blockchain*, em um sistema de armazenamento *Big Data* central ou distribuído, inserindo no *blockchain* apenas o resumo (ou *hash*) criptográfico dos dados. Chen et al. [4] propõe dividir os dados em dois grupos de informações, sensíveis e não sensíveis, sendo inseridos no *blockchain* as informações sensíveis e o *hash* das informações não sensíveis. Embora essas soluções amenizem o problema da escalabilidade, aumentando a quantidade de transações que cabem em um bloco através da diminuição do seu tamanho, todas essas propostas compartilham uma mesma limitação em relação à auditoria dos

Este trabalho foi parcialmente financiado com recursos da Norte Energia S.A., subvenção P&D-S004/2021 - Projeto P&D ANEEL PD-07427-0421/2021.

dados. É possível apenas a verificação completa do conjunto de dados associado ao respectivo *hash* criptográfico. Em cenários que podem envolver a perda parcial das informações de um pacote de dados, esse *hash* se torna irrelevante para investigar se os dados remanescentes permanecem íntegros ou não.

Neste artigo, propomos um novo mecanismo complementar de auditoria e verificação de integridade em sistemas de monitoramento que integram *Big Data* e *blockchains*. Nossa proposta utiliza *hashes* de similaridade, usualmente descritos na literatura como *Hash* Sensível à Localidade (do inglês, *Locality-Sensitive Hashing*, ou LSH), em conjunto com *hashes* criptográficos. A vantagem dessa aplicação consiste na possibilidade de se auditar os dados *off-chain* (i.e., armazenados em sistemas *Big Data* e verificados por meio de um *blockchain*) mesmo em caso de uma perda parcial dos dados. Sendo assim, em uma auditoria onde um atacante induza uma supressão de informações legítimas à ocorrência de uma perda parcial, a fim de invalidar a verificação de integridade através do *hash* criptográfico, é possível utilizar um LSH para investigar se de fato o conjunto de dados remanescente pode ser caracterizado de forma confiável como um subconjunto dos dados originais. Isso permite diferenciar uma perda de dados não intencional de uma tentativa deliberada de fraude. Como cenário de estudo, consideramos o sistema de monitoramento da barragens e taludes descrito no trabalho de Assumpção et al. [2]. Trata-se de um sistema baseado em sensores de ultrassom, que coleta dados de forma segura e os armazena em nuvem, integrando *Big Data* e *blockchains* permissionados por meio da técnica *off-chain*. Neste cenário, avaliamos uso de LSH para auditoria dos dados, comparando o desempenho dos algoritmos MinHash [27] e SimHash [29], usualmente descritos na literatura para aplicações envolvendo busca de conteúdo e análise forense de dados. Em nosso estudo, avaliamos também como o tamanho do *payload* das transações impacta uma rede *blockchain* implementada sobre a plataforma Hyperledger Fabric [1], evidenciando que de fato a solução de armazenamento *off-chain* otimiza o *throughput* da solução, mesmo com o acréscimo dos *hashes* LSH. Os resultados obtidos mostram-se promissores, indicando que *hashes* de similaridade podem desempenhar um papel importante na auditoria de dados *off-chain*.

O restante deste trabalho está organizado da seguinte forma: A Seção II descreve os conceitos fundamentais necessários para o entendimento adequado do artigo. A Seção III descreve o cenário onde a aplicação está inserida, bem como o modelo de ataque abordado e a solução proposta. A Seção IV contém uma descrição do estudo de caso e da arquitetura do *blockchain* associada ao experimento, bem como uma avaliação geral dos *hashes* de similaridade utilizando experimentos computacionais. A Seção V consiste em uma discussão dos resultados obtidos. Por fim, a Seção VI apresenta as conclusões alcançadas com o trabalho e as propostas para trabalhos futuros.

II. Conceitos elementares e trabalhos relacionados

Nesta subseção apresentamos alguns conceitos elementares necessários à devida compreensão de nossa proposta, e também os trabalhos relacionados que servem como referência para sua fundamentação. Os conceitos abordados a seguir tratam dos LSH e suas aplicações, da tecnologia de *blockchain*, e sua utilização em conjunto com sistemas de *Big Data*.

A. Hash Sensível à Localidade

O problema de consulta do vizinho mais próximo (do inglês *near-neighbor query*) está presente em diversas aplicações de banco de dados, geralmente associadas à pesquisa de similaridade [7]. Segundo Gionis et al. [7], encontrar o vizinho aproximadamente mais próximo (do inglês, *approximate near-neighbor, ou ANN*) é aceitável em muitas aplicações, tendo resultados parecidos com a busca pelo vizinho mais próximo exato, porém apresentando ganhos no tempo de execução. Este efeito é esperado caso haja uma boa métrica da similaridade.

Através de funções *hash* aleatórias, o Hash Sensível à Localidade (do inglês, *Locality Sensitive Hashing*, ou LSH) é capaz de mapear dados de alta dimensão para uma dimensão inferior, tornando-se uma das soluções mais populares para encontrar o ANN [9]. Inicialmente proposto por Indyk e Motwani [8], seu uso tem aplicações em diversas áreas, como *machine learning*, ciências geológicas, compressão de dados, e investigação forense [8], [9], [18].

MinHash e SimHash são dois algoritmos LSH amplamente adotados em aplicações onde é necessário processar grandes quantidades de dados [19]. O Minhash sugere um algoritmo para comprimir dados mantendo a *Similaridade de Jaccard* entre qualquer par de conjuntos definido a partir de vetores binários [16] [19]. O Simhash, por outro lado, é um LSH para *Similaridade de Cosseno*, que funciona para dados gerais com valores reais [19].

B. Blockchain

O termo *blockchain* tornou-se difundido com o Bitcoin, sendo aplicado ao *design* que sustenta as operações da criptomoeda. Inicialmente proposto por Nakamoto [13], o conceito de *blockchain* como difundido hoje envolve uma lista cronologicamente ordenada de blocos, mantida por uma rede *peer-to-peer* que opera com protocolos de consenso para decidir quando criar novos blocos e transações [31] [10].

As plataformas de implementação de *blockchains* podem ser classificadas como não permissionadas, quando qualquer indivíduo pode participar da rede e do protocolo de consenso, ou permissionadas, quando o consenso é alcançado por um conjunto de *peers* conhecidos e identificáveis [1], [12], [24], [25]. Geralmente, os protocolos de consenso para *blockcchains* permissionados gastam menos recursos computacionais e podem alcançar uma menor latência de transação e maior *throughput* [20]. Entretanto, o estado da arte dos protocolos de consenso ainda impõe restrições de desempenho, pelo fato de impossibilitar que a rede, em termos do tratamento de transações sob demanda, seja escalável [20], [24]. Em outras

palavras, não é possível simplesmente aumentar o número de *peers* envolvidos no protocolo de consenso para atender um número crescente de transações. O *throughput* de uma rede *blockchain* é limitado principalmente pela quantidade de transações que cabem em cada bloco e pela taxa de geração de novos blocos. Aumentar o tamanho do bloco permite que ele suporte mais transações [1], porém incrementa o tempo que o bloco demora para ser processado por toda rede, o que pode ser explorado como uma vulnerabilidade. Sendo assim, o tempo de geração de cada bloco e seu tamanho devem ser pensados a fim de obter uma harmonia entre eles [32].

C. Blockchains e aplicações de Big Data

Diversos trabalhos na literatura propõem a utilização do *blockchain* como um mecanismo de verificação de integridade de *Big Data*. Li et al. [11] propôs um sistema de auditoria pública utilizando a tecnologia *blockchain* no lugar de uma terceira parte autenticadora para reduzir o custo computacional e solucionar os problemas de integridade e confiabilidade associados à computação em nuvem. Yang et al. [30] propôs um sistema de compartilhamento de *Big Data* baseado na tecnologia *blockchain* para garantir que os dados de uma transação não sejam alterados enquanto o usuário estiver em posse deles. Wang e Song [26] propuseram um sistema de armazenamento seguro de dados da área da saúde, utilizando o *blockchain* para garantir a integridade e rastreabilidade dos dados armazenados na nuvem. Sun et al. [21] propõe a utilização do *blockchain* com o *Big Data* de dados médicos, porém utilizando um sistema de armazenamento distribuído, o IPFS (*InterPlanetary File System*), para o armazenamento dos dados. Chen et al. [4] desenvolveu uma aplicação de *blockchain* permissionada baseada na arquitetura do Hyperledger Fabric, utilizando o *blockchain* como mecanismo de controle de integridade. Um aspecto comum entre a maioria dessas aplicações é que elas exploram a técnica *off-chain*, usando o *blockchain* muito mais como um mecanismo de auditoria e verificação de dados do que como um repositório de dados propriamente dito.

III. Método de verificação de *Big Data* usando *blockchains* e LSH

Neste trabalho, nossa proposta consiste em um mecanismo de verificação de dados em um sistema de *Big Data* usando *blockchains* e algoritmos de LSH. Como discutido anteriormente, muitos autores exploraram a técnica *off-chain*, que depende essencialmente do uso de *hashes* criptográficos armazenados em um *blockchain* para verificação e auditoria de pacotes de dados armazenados em um sistema de *Big Data* convencional. Entretanto, o uso do *hash* criptográfico é efetivo apenas em cenários onde o pacote de dados encontra-se disponível na integra para verificação. Em caso de perda parcial das informações (e.g., uma pequena parte do pacote foi perdida, mas o restante encontra-se disponível para auditoria), não há qualquer análise que possa ser feita a partir do *hash* criptográfico. Nossa proposta se baseia na ideia de que o uso de *hashes* LSH podem suprir essa lacuna, possibilitando o desenvolvimento de mecanismos de auditoria para pacotes de dados parciais. Quando agregados à técnica *off-chain*, esses *hashes* de similaridade podem complementar a funcionalidade de um *hash* criptográfico, tornando a solução mais robusta contra incidentes não intencionais envolvendo perda de dados e também ataques cibernéticos. Vale ressaltar que o uso dos LSH diz respeito ao controle de integridade de um pacote de dados armazenado em um sistema de *Big Data*, e não à estrutura de dados adotada para armazenamento de blocos do blockchain.

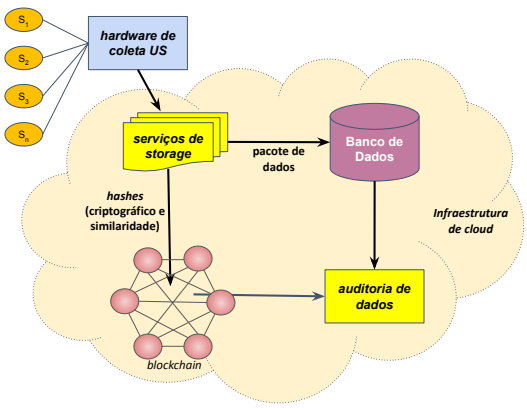

Fig. 1. Estrutura de armazenamento *Big Data* considerada.

A. Cenário de estudo

Nosso cenário de estudo corresponde aos sistemas de monitoramento de ICs baseados em redes de sensores e dispositivos IoT. Em sua maioria, tais sistemas capturam medições associadas a grandezas físicas, e enviam essas informações para um serviço de *Big Data*. Os dados coletados consistem essencialmente de séries temporais. Em sistemas onde o monitoramento ocorre de forma contínua e inclui um elevado número de sensores e dispositivos de coleta, não é possível se utilizar *blockchains* como a tecnologia principal de armazenamento, dadas suas restrições de escalabilidade.

Assim, estabelecemos como ponto de partida uma solução *off-chain* descrita na Figura 1. O fluxo de informação começa no sistema de *hardware* de coleta, que recebe os dados de diferentes sensores e envia para o serviço de *storage*. A partir deste ponto, os pacotes de dados são enviados para um banco de dados que suporta *Big Data*, enquanto elementos de auditoria associados a esses pacotes (inicialmente, *hashes* criptográficos) são escritos no *blockchain*. Por fim, processos de auditoria de dados podem ser utilizados de forma contínua para consultar pacotes no banco de dados, procurar pelo respectivo *hash* no *blockchain*, e atestar se as informações armazenadas permanecem consistentes.

B. Modelo de ataque

Em nosso estudo, consideramos o seguinte modelo de ataque. O atacante é uma entidade com acesso à solução de *Big Data*, com capacidade para inserir, excluir ou modificar pacotes de dados. Assume-se que este atacante pode ter

motivações maliciosas para executar essas ações. Por exemplo, ele pode ter a intenção de sabotar o sistema de monitoramento, removendo informações que seriam importantes para a detecção de algum evento grave em andamento, ou ainda de inserir informações adulteradas que possam gerar falsas detecções e alarmes.

Um ataque envolvendo apenas a manipulação direta dos dados, sem alterar o tamanho do pacote, seria facilmente detectado em uma auditoria devido à inconsistência entre o *hash* criptográfico gerado e aquele armazenado no *blockchain*. Uma possível ação do atacante é destruir completamente os dados de um pacote no banco de dados, impedindo a verificação destes durante a auditoria. Entretanto, a remoção de um pacote completo de dados pode caracterizar uma situação controversa que resulta em um efeito inverso ao desejado, criando suspeição em relação ao ataque e permitindo que o atacante seja exposto.

Sendo assim, assume-se que uma provável ação do atacante seja a exclusão parcial dos dados armazenados, ou seja, apenas a fração da informação que serve ao seu intento. Durante a auditoria, o incidente pode ser apresentado como uma perda parcial informações, sob a alegação de que essa perda foi não intencional e que o conjunto de dados apresentado preserva sua integridade. Como uma pequena mudança em um pacote de dados gera um *hash* criptográfico completamente diferente, não é possível afirmar se os dados parciais apresentados permanecem íntegros, sendo impossível extrair qualquer evidência se de fato o houve um incidente não intencional ou uma tentativa de fraude.

Para fins práticos, define-se que o atacante não tem como comprometer os elementos de auditoria armazenados no *blockchain*. Esta é uma premissa bastante razoável, uma vez que *blockchains* garantem a imutabilidade das informações armazenadas, e consequentemente sua integridade, sendo susceptíveis apenas a ataques de conluio. Desse modo, assume-se que o atacante não dispõe de recursos para realizar ataques de conluio bem sucedidos, nem de subverter os procedimentos automáticos de auditoria de dados, que podem ser baseados em *smart contracts*.

C. Solução baseada em LSH

Em resposta ao modelo de ataque descrito, a solução proposta consiste em utilizar LSH como elemento de auditoria, em conjunto com os *hashes* criptográficos já utilizados com a técnica *off-chain*. Isso torna os procedimentos de auditoria capazes de auditar os pacotes de dados armazenados também em caso de perda parcial de dados, uma vez que é esperado que um LSH do pacote de dados original preserve alta similaridade com o LSH do pacote de dados com perdas, até um determinado limite de perda de informação tolerável.

A aplicação dessa ideia pode ser feita conforme o seguinte procedimento. Primeiramente, compara-se o *hash* criptográfico armazenado no *blockchain* com aquele calculado a partir do respectivo pacote de dados armazenado no banco de dados. Caso os *hashes* criptográficos comparados sejam iguais, tem-se que o pacote de dados armazenado no banco de dados não foi comprometido. Caso o resultado seja diferente, verifica-se se o tamanho esperado para o pacote de dados (i.e., número de amostras nas respectivas séries temporais) está correto. Se o tamanho constatado for menor, existe a possibilidade de se tratar de uma perda parcial de dados. Neste caso, a auditoria prossegue com a comparação do LSH do pacote auditado com aquele armazenado no *blockchain*, estipulando-se um limiar de diferença mínimo que deve ser satisfeito para se evidenciar que aquele pacote auditado é de fato um subconjunto do pacote de dados original. Em qualquer outro caso, o procedimento de auditoria considera o pacote de dados suspeito de ação fraudulenta, registra o resultado no próprio *blockchain*, e permite a notificação de autoridades responsáveis para que iniciem um processo investigativo mais abrangente.

IV. Avaliação usando Experimento Computacional

A. Estudo de caso associado ao experimento

Em nossa proposta, foi utilizado como estudo de caso um sistema de monitoramento de taludes desenvolvido em um projeto de P&D em parceria com uma empresa geradora de energia [2]. O sistema consiste na distribuição de um número elevado de sensores ao longo dos taludes e também nas barragens da usina hidroelétrica. Esses sensores são agrupados por dispositivos de hardware usados para a coleta de dados, que conseguem transmitir as informações geradas para o ambiente em *cloud*, onde as informações serão tratadas e armazenadas. Devido ao elevado número de sensores e dispositivos de coleta, o número de transações, bem como o volume de dados esperado, é significativamente alto. Tal aspecto permite caracterizar esse sistema como um sistema de *Big Data*.

Para o presente experimento, foi utilizado um dispositivo de coleta simplificado, consistindo de quatro transdutores piezoelétricos acoplados em um tanque de água [2]. Um gerador de funções simula a emissão dos sinais de ultrassom captados pelos transdutores. Foi assegurado que os dados de apenas um sensor por sinal seja considerado no *dataset*, para fins de análise de eficiência dos algoritmos LSH. Cada pacote gerado contém dados amostrados a uma taxa de 50KHz durante um segundo, ou seja, 50 mil valores de ponto flutuante. O tamanho médio de um pacote de dados corresponde a 600 *Kbytes*.

B. Arquitetura do blockchain

A implementação da nossa arquitetura utiliza como base a versão 2.3 do Hyperledger Fabric[1]. O Fabric é uma plataforma *open source* que implementa *blockchains* permissionados e é mantido pela Linux Foundation, um consórcio composto por diferentes companhias e que atualmente responde pelas principais iniciativas globais associadas ao software aberto [1], [3]. Dentre as justificativas para seu uso, podemos citar:

- O fato do Fabric ser uma plataforma *open-source* facilita o desenvolvimento de soluções de baixo custo sem a dependência de plataformas proprietárias;

[1]https://hyperledger-fabric.readthedocs.io/en/release-2.3

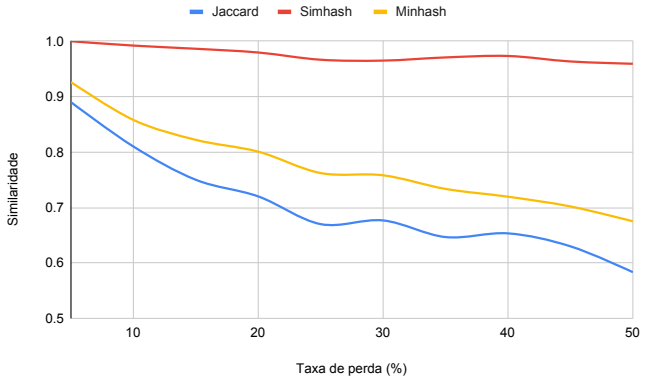

Fig. 2. Gráfico da similaridade em razão da taxa de perda.

- O modelo de rede *blockchain* proposto como arquitetura descentralizada se adequa bem ao Fabric;
- O desempenho do Fabric, que atualmente apresenta uma das melhores taxas de *throughput* entre as demais plataformas [1], constitui uma característica interessante para integração em sistemas *Big Data*;
- O mecanismo de consenso customizável permite alterar o protocolo utilizado, visando tanto segurança quanto desempenho;
- Existe vasta documentação associada ao Fabric, e diferentes ferramentas de desenvolvimento de software (do inglês *Software Development Kits*, ou SDK);
- O suporte aos *smart contracts*, que podem ser invocados pelas transações para realizar determinadas tarefas com os dados.

Em nossa aplicação, o principal elemento de infraestrutura do *blockchain* são as organizações. As organizações representam entidades independentes que, juntas, cooperam entre si para formarem a rede *blockchain*. Essa independência é essencial para evitar ataques de conluio. Cada organização é responsável por alocar um certo número de *peers* necessários para participar do quórum de consenso. Dentre esses *peers*, os *endorsers* são os responsáveis por submeter as transações, enquanto os outros apenas replicam o *ledger* do *blockchain*.

C. Análise dos algoritmos LSH

Para analisar a eficiência do uso de algoritmos LSH na auditoria de pacotes de dados com perdas parciais no escopo da aplicação descrita como nosso estudo de caso, foi elaborado o seguinte experimento.

O primeiro passo envolveu a análise do comportamento dos algoritmos Simhash e Minhash em comparações envolvendo pacotes de dados com perdas parciais. Inicialmente, foi escolhido um pacote de dados aleatório armazenado no sistema de *Big Data*, e a partir desse pacote foram geradas 300 modificações, simulando diferentes condições de perda parcial de dados. Cada pacote de dados modificado foi gerado removendo-se uma quantidade aleatória de informação do início e final do pacote, mas que totalizam um fator fixo de perda de informação. Esse fator de perda começou em 5% e foi aumentado em mais 5% a cada 30 pacotes de dados gerados, chegando a 50% de perda nos últimos 30 pacotes de dados. Cada um dos 300 pacotes de dados modificados foram comparados com o pacote de dados original por meio da Similaridade de Jaccard. Em seguida, foram calculados o Simhash e o Minhash do pacote original e das 300 versões modificadas, de modo a realizar a comparação entre eles e analisar como os dois algoritmos se comportam. A média das Similaridades de Jaccard e dos *hashes* de similaridade aplicados, encontradas para cada taxa de perda de informação, é apresentada na Figura 2.

A partir desse gráfico é possível ver que a curva da similaridade entre os pacotes de dados calculada a partir do Minhash teve um comportamento semelhante à curva calculada através da Similaridade de Jaccard, tal como esperado. Enquanto isso, a curva de similaridade do Simhash teve uma variação muito pequena, permanecendo alta mesmo quando o pacote de dados original foi comparado com pacotes de dados com 50% de perda. O Simhash foi pensado para acompanhar a Similaridade dos Cossenos, porém, como estamos trabalhando com pacotes de dados de tamanhos diferentes na comparação, não é possível implementar essa medida de similaridade para fins de análise. Ainda assim, a comparação entre o Simhash e o Minhash é uma comparação válida, já que os dois algoritmos se propõem a fazer a mesma coisa, mesmo que usando métodos diferentes.

D. Determinação dos parâmetros de verificação do LSH

Nosso experimento teve prosseguimento com a tentativa de se estipular os parâmetros do mecanismo de auditoria baseado em LSH, em especial um limiar aceitável para definir com precisão quão próximos dois *hashes* de similaridade precisam ser para que possamos considerá-los como advindos da mesma origem, bem como para quais taxas de perda o mecanismo atua bem. A ideia consiste essencialmente em tentar encontrar esse limiar para um pacote de dados de teste e, posteriormente, verificar se este limiar pode ser mantido para análise dos demais pacotes de dados dentro do *dataset* analisado em nosso estudo de caso. Assim, foi selecionado de forma aleatória um pacote de dados cujo o sinal é apresentado na Figura 3.

Para verificar o quão bem os algoritmos LSH identificam pacotes de dados similares, foram gerados 120 novos pacotes de dados a partir do pacote de dados de referência. Variamos a taxa de perda entre 5 e 40%, dobrando o valor inicial a cada 30 pacotes de dados gerados (i.e., percentuais de 5, 10, 20, e 40). O mesmo procedimento foi feito utilizando um segundo pacote de dados aleatório do *dataset* para criar um conjunto de dados que permitisse testar quão bem os *hashes* de similaridade diferenciam arquivos que não são oriundos da mesma origem.

A partir da Figura 4, pode-se perceber que o Minhash apresenta uma acurácia bastante elevada para um valor de limiar entre 0,4 e 0,6. Vale ressaltar que, dentro desse intervalo, a taxa de perda não influencia no resultado.

Como visto na Figura 5, o Simhash, por outro lado, apresenta resultados bem menos consistentes. A acurácia do

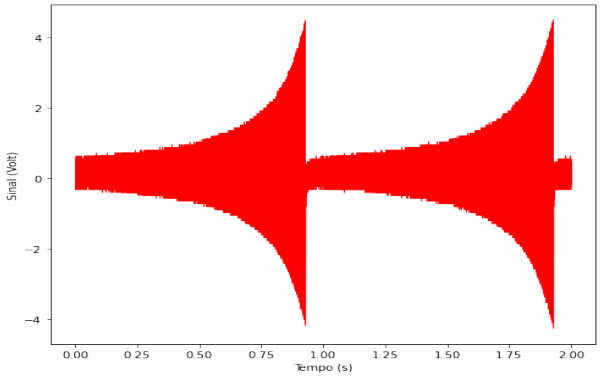

Fig. 3. Sinal selecionado aleatoriamente para se determinar um limiar de aceitação do LSH.

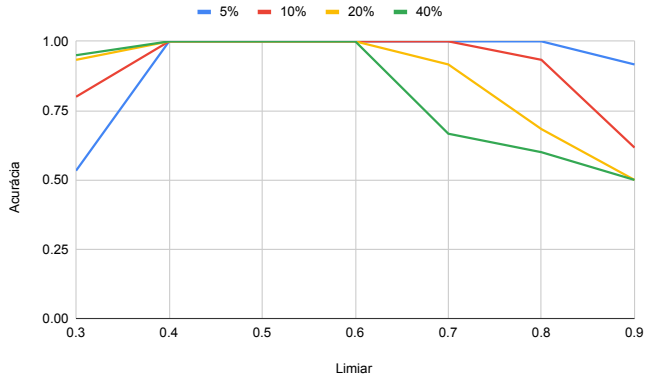

Fig. 4. Gráfico da acurácia do Minhash em razão do limiar selecionado.

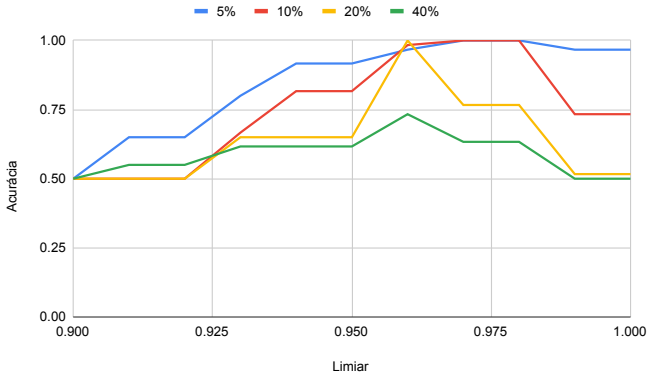

Fig. 5. Gráfico da acurácia do Simhash em razão do limiar selecionado.

algoritmo varia bastante em relação a taxa de perda e ao valor do limiar adotado. Ainda assim, o Simhash apresenta uma taxa de precisão alta para perdas de até 30%, sugerindo um limiar de 0,96.

E. Validação dos resultados do LSH

Para realizar a validação dos limiares propostos na seção anterior e verificar o desempenho dos *hashes* de similaridade, foi extraído de nosso *dataset* um novo conjunto de 30 pacotes de dados diferentes, sendo geradas em seguida 30 modificações de cada um deles. Para ter um teste mais conciso, foi escolhida apenas uma taxa de perda intermediária, de 30%, para todas as modificações. Com o novo conjunto de pacotes, comparamos o Simhash e o Minhash calculados dos 30 pacotes de dados íntegros com os de suas respectivas modificações. A Tabela I contém as métricas obtidas utilizando como limares os valores de 0,6 para o Minhash e de 0,96 para o Simhash.

TABLE I
RESULTADO OBTIDOS NA COMPARAÇÃO DE PACOTES USANDO LSH.

Métrica	Minhash	Simhash
FNR	0,0478	0,374
FPR	0,0256	0,388
Acurácia	0,9633	0,6189
f1-score	0,9629	0,6163

TABLE II
INSTÂNCIA EC2 UTILIZADA DURANTE OS TESTES.

Nome da instância	CPU (Núcleos)	Memória (GB)
m5.2xlarge	8	32

O Minhash apresentou excelentes resultados. As taxas de falsos negativos (do inglês, *False Negative Rate*, ou FNR) e falsos positivos (do inglês, *False Positive Rate*, ou FPR) foram consistentemente baixas, resultando em uma acurácia e *f1-score* próximos a 1. Os valores próximos de FNR e FPR, quando comparados ao número total de amostras, indicam que o limiar adotado resultou em um bom desempenho por parte do Minhash para comparar os pacotes de dados de forma precisa.

Ao contrário do Minhash, a aplicação do Simhash não obteve bons resultados, apresentando baixa acurácia e *f1-score*. Devido à variação baixa do valor apresentado pelo Simhash em relação a taxa de perda mostrada anteriormente na Seção IV, esse resultado já era esperado. FNR e FPR apresentaram valores similares, dando a entender que variar o limiar adotado provavelmente não resultaria em melhores resultados.

F. Avaliação de desempenho do blockchain

Assim como qualquer outra rede *blockchain*, o Fabric apresenta restrições quanto a sua escalabilidade. O melhor desempenho de *benchmark* nessa rede aponta para um limite teórico de aproximadamente duas mil transações por segundo [1]. Entretanto, com base em trabalhos anteriores que exploram ambientes mais realísticos, pode-se observar que o *throughput* raramente supera as 400 tps [15].

Neste teste, instanciamos a rede *blockchain* e um cliente *multithread* responsável por gerar uma carga de trabalho (i.e., *workload*) elevada de transações concorrentes. O *hardware* utilizado nesse teste corresponde a uma instância EC2 do serviço de nuvem da *Amazon Web Services* (AWS), com as especificações descritas na Tabela II. A rede *blockchain* instanciada tem duas organizações, cada uma possuindo apenas um *peer*, para um teste mais conciso. O protocolo de consenso escolhido foi o RAFT. A ferramenta de monitoramento Hyperledger Explorer[2] foi usada para monitorar as transações

[2]https://www.hyperledger.org/use/explorer

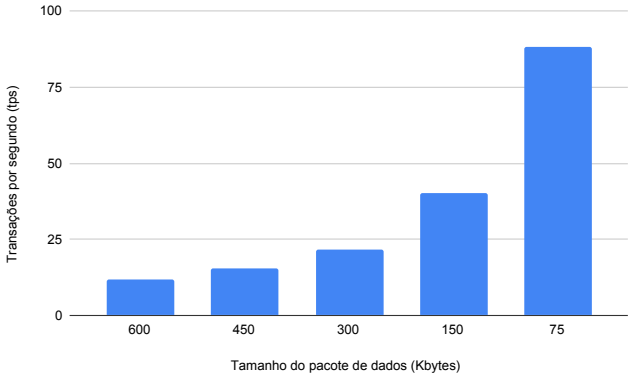

Fig. 6. Desempenho da rede ao variar o *payload*.

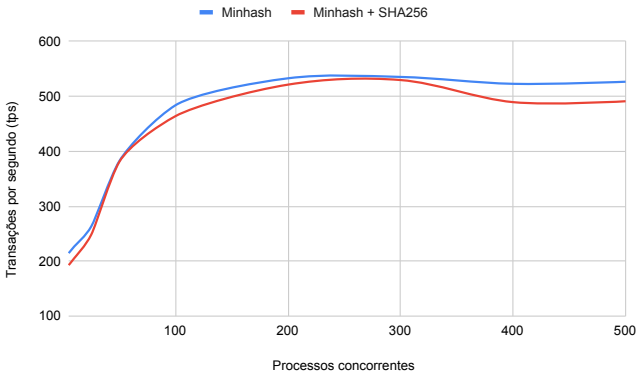

Fig. 7. Gráfico do desempenho variando o *workload*.

e avaliar o desempenho da rede. Os testes de desempenho realizados consistem em uma análise simples do *throughput* da rede blockchain, que inclui dois cenários:

- Impacto do tamanho do *payload*: verificar o desempenho da rede *blockchain* ao variar o tamanho do pacote de dados inserido por meio de um *smart contract*;
- Desempenho de uma aplicação *off-chain* usando LSH: verificar o desempenho do *blockchain* ao inserir pacotes de dados contendo o *hash* criptográfico SHA-256 e também um LSH, aumentando-se o *workload* até atingir o limite teórico do Fabric.

A Figura 6 mostra que ao diminuirmos o tamanho do *payload* das transações, há um ganho significativo no *throughput* do Fabric. Com isso é possível concluir que, de fato, o tamanho do pacote de dados inserido no *blockchain* influencia significativamente seu desempenho, o que justifica a abordagem *off-chain* adotada por diferentes autores.

A última etapa de nosso experimento foi avaliar o desempenho do Fabric ao invocar um *smart contract* associado ao armazenamento *off-chain* que insere no *blockchain* um *hash* criptográfico SHA-256 e um LSH para cada pacote de dados verificável. Devido ao baixo desempenho do Simhash, optou-se por realizar o teste de desempenho em modo *off-chain* apenas com o Minhash, sendo este gerado com um tamanho de 80 *bytes* por pacote de dados. Na Figura 7, é possível ver que o *throughput* resultante permanece elevado, ultrapassando a faixa de 500 tps, que constitui um excelente desempenho para o Fabric quando comparado àquele obtido em outros trabalhos que também exploram casos de uso realísticos.

V. Discussão dos Resultados

Nesta seção, fazemos uma discussão complementar dos resultados obtidos em nosso trabalho, enfatizando as principais contribuições decorrentes da proposta apresentada e dos experimentos desenvolvidos.

Primeiramente, temos que para a proposta apresentada, o Minhash apresentou um resultado muito mais promissor que o Simhash. Tal resultado provavelmente se deve à forma como os dois algoritmos foram pensados e também à arquitetura da aplicação onde foram testados. É importante relembrar que nossa proposta se destina em um primeiro momento a sistemas de monitoramento de infraestruturas críticas com ativos físicos, os quais se baseiam fortemente na análise de dados gerados por sensores. Deste modo, tais dados são essencialmente séries temporais descritas por grandezas físicas.

Considerando tais aspectos, temos que o Simhash gera *shingles* a partir de sua entrada (no nosso caso, pacotes de dados contendo séries temporais), o que nada mais é do que dividir pedaços de informação em uma quantidade fixa de *tokens* que são sobrepostos. A partir desses *shingles*, são gerados *hashes* criptográficos parciais e, no fim, de acordo com quão repetitivos esses *hashes* são, o Simhash é obtido como uma combinação deles. Por utilizarmos um *dataset* formado por pacotes de dados onde o sinal da frequência varia entre 20 e 1KHz, é possível que haja muitos valores repetidos na escala de digitalização do sinal, fazendo com que o Simhash mostre que há grande similaridade entre eles em todos os casos.

Em contrapartida, entendemos que o Minhash, por se basear na medida de Similaridade de Jaccard, sugere a ideia de álgebra de conjuntos para estabelecer relações entre os elementos de dados presentes tanto no pacote de dados original quanto no *hash* extraído. Tal relação pode se beneficiar do fato de que pacotes de dados com perdas parciais são na verdade aproximações de subconjuntos dos pacotes de dados originais, apresentando uma medida elevada de intersecção (i.e., o pacote de dados original de fato contém todos os elementos dos pacotes de dados com perdas) em relação à união dos dois conjuntos de dados, que por sua vez resulta no pacote de dados original. Assim, existe a hipótese de que o bom desempenho obtido na aplicação do Minhash em nosso caso de estudo derive diretamente dessa relação observada entre os dados.

Por fim, observamos que um resultado importante deste artigo consiste na análise de desempenho do Fabric em função do tamanho do *payload* das transações submetidas ao *blockchain*. Embora tal resultado seja esperado, não encontramos até então em nossas pesquisas outros trabalhos da literatura analisando este aspecto em particular. Entretanto, os resultados obtidos mostram que o tamanho do *payload* tem um impacto evidente sobre o *throughput* desta plataforma, sugerindo inclusive que esse aspecto pode ser explorado no

projeto de soluções implementadas no Hyperledger Fabric, além dos parâmetros relacionados ao tamanho dos blocos e limite de transações por bloco, já amplamente discutidos por outros autores.

VI. Conclusão

Neste trabalho abordamos o uso de *hashes* sensíveis à localidade para complementar a literatura atual a respeito de soluções de auditoria e verificação de integridade para sistemas *Big Data*. Considerando o cenário onde utiliza-se a técnica *off-chain* para incrementar o desempenho do *blockchain*, foi possível demonstrar através dos resultados obtidos que o uso do Minhash em conjunto com um *hash* criptográfico torna o processo de auditoria mais robusto, ajudando na verificação de integridade de pacotes com perdas parciais. Como trabalho futuro, temos em vista o estudo de LSH mais recentes, como o MinMaxHash, sugerido como uma possível melhoria do Minhash. Novos cenários de ataque também serão explorados, bem como mecanismos de auditoria alternativos que possam ser combinados para lidar com situações de perda parcial ou total dos dados.

References

[1] Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart, Christopher Ferris, Gennady Laventman, Yacov Manevich, et al. Hyperledger fabric: a distributed operating system for permissioned blockchains. In *Proceedings of the Thirteenth EuroSys Conference*, pages 1–15, 2018.

[2] Paulo Assumpcao, Carlos Oliveira, Wilson Melo, and Luiz Carmo. Sensors fingerprints using machine learning: a case study on dam monitoring systems. In *2022 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, pages 1–6. IEEE, 5 2022.

[3] Christian Cachin et al. Architecture of the hyperledger blockchain fabric. In *Workshop on distributed cryptocurrencies and consensus ledgers*, volume 310, page 4, 2016.

[4] Jian Chen, Zhihan Lv, and Houbing Song. Design of personnel big data management system based on blockchain. *Future Generation Computer Systems*, 101:1122–1129, 2019.

[5] Hong-Ning Dai, Hao Wang, Guangquan Xu, Jiafu Wan, and Muhammad Imran. Big data analytics for manufacturing internet of things: opportunities, challenges and enabling technologies. *Enterprise Information Systems*, 14(9-10):1279–1303, 2020.

[6] Christian Esposito, Alfredo De Santis, Genny Tortora, Henry Chang, and Kim-Kwang Raymond Choo. Blockchain: A panacea for healthcare cloud-based data security and privacy? *IEEE Cloud Computing*, 5(1):31–37, 2018.

[7] Aristides Gionis, Piotr Indyk, Rajeev Motwani, et al. Similarity search in high dimensions via hashing. In *Vldb*, pages 518–529, 1999.

[8] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613, 1998.

[9] Omid Jafari, Preeti Maurya, Parth Nagarkar, Khandker Mushfiqul Islam, and Chidambaram Crushev. A survey on locality sensitive hashing algorithms and their applications. *arXiv preprint arXiv:2102.08942*, 2021.

[10] Chenxin Li, Peilun Li, Dong Zhou, Zhe Yang, Ming Wu, Guang Yang, Wei Xu, Fan Long, and Andrew Chi-Chih Yao. A decentralized blockchain with high throughput and fast confirmation. In *2020 USENIX Annual Technical Conference*, pages 515–528, 2020.

[11] Jiaxing Li, Jigang Wu, Guiyuan Jiang, and Thambipillai Srikanthan. Blockchain-based public auditing for big data in cloud storage. *Information Processing & Management*, 57(6):102382, 2020.

[12] Wilson S Melo Jr, Lucas S Dos Santos, Lucila MS Bento, Paulo R Nascimento, Carlos AR Oliveira, and Ramon R Rezende. Using blockchains to protect critical infrastructures: a comparison between ethereum and hyperledger fabric. *International Journal of Security and Networks*, 17(2):77–91, 2022.

[13] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review*, page 21260, 2008.

[14] Ana Carolina de Oliveira Neves, Flávia Peres Nunes, Felipe Alencar de Carvalho, and Geraldo Wilson Fernandes. Neglect of ecosystems services by mining, and the worst environmental disaster in brazil. *Natureza & Conserva o*, 1(14):24–27, 2016.

[15] Daniel Peters, Artem Yurchenko, Wilson Melo, Katsuhiro Shirono, Takashi Usuda, Jean-Pierre Seifert, and Florian Thiel. It security for measuring instruments: confidential checking of software functionality. In *Future of Information and Communication Conference*, pages 701–720. Springer, 2020.

[16] Rameshwar Pratap, Karthik Revanuru, Ravi Anirudh, and Raghav Kulkarni. Efficient compression algorithm for multimedia data. In *2020 IEEE Sixth International Conference on Multimedia Big Data (BigMM)*, pages 245–250. IEEE, 2020.

[17] Arun Raman and Fei Liu. An investigation of the brumadinho dam break with hec ras simulation. *arXiv preprint arXiv:1911.05219*, 2019.

[18] Kexin Rong, Clara E Yoon, Karianne J Bergen, Hashem Elezabi, Peter Bailis, Philip Levis, and Gregory C Beroza. Locality-sensitive hashing for earthquake detection: A case study of scaling data-driven science. *arXiv preprint arXiv:1803.09835*, 2018.

[19] Anshumali Shrivastava and Ping Li. In Defense of Minhash over Simhash. In Samuel Kaski and Jukka Corander, editors, *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, volume 33 of *Proceedings of Machine Learning Research*, pages 886–894, Reykjavik, Iceland, 22–25 Apr 2014. PMLR.

[20] Joao Sousa, Alysson Bessani, and Marko Vukolic. A byzantine fault-tolerant ordering service for the hyperledger fabric blockchain platform. In *2018 48th annual IEEE/IFIP international conference on dependable systems and networks (DSN)*, pages 51–58. IEEE, 2018.

[21] Jin Sun, Xiaomin Yao, Shangping Wang, and Ying Wu. Blockchain-based secure storage and access scheme for electronic medical records in ipfs. *IEEE Access*, 8:59389–59401, 2020.

[22] Baskut Tuncak. Lessons from the samarco disaster1. *Business and Human Rights Journal*, 2(1):157–162, 2017.

[23] Taylor Rodriguez Vance and Andrew Vance. Cybersecurity in the blockchain era: a survey on examining critical infrastructure protection with blockchain-based technology. In *2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T)*, pages 107–112. IEEE, 2019.

[24] Marko Vukolić. The quest for scalable blockchain fabric: Proof-of-work vs. bft replication. In *International workshop on open problems in network security*, pages 112–125. Springer, 2015.

[25] Marko Vukolić. Rethinking permissioned blockchains. In *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts*, pages 3–7, 2017.

[26] Hao Wang and Yujiao Song. Secure cloud-based ehr system using attribute-based cryptosystem and blockchain. *Journal of medical systems*, 42(8):1–9, 2018.

[27] Wei Wu, Bin Li, Ling Chen, Junbin Gao, and Chengqi Zhang. A review for weighted minhash algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 34(6):2553–2573, 2020.

[28] Yulei Wu, Hong-Ning Dai, and Hao Wang. Convergence of blockchain and edge computing for secure and scalable iiot critical infrastructures in industry 4.0. *IEEE Internet of Things Journal*, 8(4):2300–2317, 2020.

[29] Yanwei Xu, Lianyong Qi, Wanchun Dou, and Jiguo Yu. Privacy-preserving and scalable service recommendation based on simhash in a distributed cloud environment. *Complexity*, 2017, 2017.

[30] Jiachen Yang, Jiabao Wen, Bin Jiang, and Huihui Wang. Blockchain-based sharing and tamper-proof framework of big data networking. *IEEE Network*, 34(4):62–67, 2020.

[31] Ma Zhaofeng, Wang Lingyun, Wang Xiaochang, Wang Zhen, and Zhao Weizhe. Blockchain-enabled decentralized trust management and secure usage control of iot big data. *IEEE Internet of Things Journal*, 7(5):4000–4015, 2019.

[32] Qiheng Zhou, Huawei Huang, Zibin Zheng, and Jing Bian. Solutions to scalability of blockchain: A survey. *Ieee Access*, 8:16440–16455, 2020.