Garantindo autenticidade de conteúdos Web com Blockchain

Meirylene R. E. Avelino[1], **Antônio Augusto de A. Rocha**[1]

[1]Universidade Federal Fluminense, Departamento de Ciência da Computação, Niterói, Brasil, 24210-310

meirylenerea@id.uff.br, arocha@ic.uff.br

Abstract. *The article's purpose is introduce a Blockchain use proposal for Web content records, in order to allow the auditability regarding authenticity and integrity version, besides enabling all records history consultation made for a specific URL.*

Resumo. *O objetivo desse artigo é introduzir uma proposta de uso de Blockchain para registros de conteúdos Web, com o objetivo de permitir a auditabilidade quanto a autenticidade e integridade de versão, além de possibilitar a consulta do histórico de todos os registros realizados para uma dada URL.*

1. Introdução

A internet, que surgiu em meados dos anos 60 como um meio de comunicação com propósitos militares, foi se ampliando e sendo disseminada por todo o mundo, até que nos anos 90 a internet começou a ser utilizada por usuários comuns de forma mais acentuada.A sociedade foi se acostumando a essa forma de consumir informações, serviços e conteúdo, de forma rápida em qualquer hora e lugar. Atualmente, qualquer pessoa com acesso a internet possui a facilidade de publicar alguma informação e os provedores de conteúdo, empresas sérias que são responsáveis por distribuir não somente as informações como também tecnologias e serviços, estão cada vez mais adotando meios de zelar pela sua idoneidade as regulamentações propostas pelo Marco Civil.

Blockchain possui um conceito que visa a descentralização como medida de segurança. São bases de registros e dados distribuídos e compartilhados, que possuem a função de criar um índice global para todas as transações que ocorrem em uma determinada rede. Esse trabalho visa apresentar o *BlockProof, um framework de uma aplicação que utiliza a tecnologia Blockchain, com o objetivo de armazenar, de forma unívoca, o conteúdo de qualquer página Web, independente da página possuir conteúdos dinâmicos ou estáticos, que pode ser utilizado tanto pelo consumidor do conteúdo quanto pelo provedor de conteúdo.* Se utilizado pelo consumidor do conteúdo, o usuário pode consultar todo o histórico de atualizações de uma página Web com todas as informações daquele conteúdo, além do próprio conteúdo em si; se utilizado pelo provedor de conteúdo, o provedor pode registrar na *Blockchain* o conteúdo que está sendo divulgado por ele na internet.

2. Fundamentação Teórica

Através da Internet a ubiquidade se torna possível, o ciberespaço se torna um lugar fictício, que só temos acesso pelo computador, e ainda assim ele está ligado à realidade

pelo uso que temos feito dele, transformando em um espaço intermediário entre duas realidades [Pinheiro 2006]. Essa possibilidade que a internet nos traz, acaba abrindo brechas para que pessoas mal intencionadas façam uso dessa abertura e cometam ações maliciosas, pois acreditam que o anonimato é garantido. Crime cibernético pode ser considerado como uma forma de desvio on-line utilizando tecnologia [Sieber 1998]. Tais crimes podem ser caracterizados a partir de uma ação de *hackers* ou mesmo da disseminação de *"fake news"* atribuídas a blogs, redes sociais, sites de relacionamentos, agências de notícias, ou e-mails, por exemplo [Junior and Alavarse 2014], o que pode levantar suspeitas (e denigrir a reputação), colocando em dúvida a idoneidade de pessoas ou empresas.

Na literatura podemos encontrar diversas pesquisas para auxiliar no combate aos crimes digitais, com o intuito de provar a integridade e autenticidade de um documento, imagem ou vídeo. Segundo [da Silva Pereira and de Oliveira 2019], a perícia forense computacional é a área responsável pela investigação de crimes digitais, análise de seus fatos e coleta de dados para usar como evidência e essas evidências precisam ser coletadas com certa destreza para não comprometer a informação obtida. Como um provedor pode provar (em juízo ou perante à sociedade), de forma incontestável, que uma notícia/informação não estava sendo disponibilizada em seus servidores? Por outro lado, como um usuário, consegue provar que ele viu uma determinada informação em um site, que pode conter conteúdos inverídicos, de cunho pessoal, ou alguma informação que ele gostaria de provar que fora divulgada?

A busca por uma solução que responda de forma positiva às questões colocadas acima estimulou os estudos deste trabalho. Para isso, buscou-se a utilização da tecnologia *Blockchain*, a qual está em crescente popularidade nos últimos tempos, classificada com o potencial de modificar a economia e os negócios nos próximos anos [Nakamoto 2019].

3. Trabalhos Relacionados

A proliferação de informações enganosas disseminadas nos meios de comunicação diariamente (tais como os feeds de mídia social, blogs de notícias, jornais online) se tornou um desafio para identificar fontes confiáveis. A maioria das abordagens encontradas para detecção de conteúdos falsos ou manipulados de alguma forma, que são divulgados na internet, geralmente utilizam técnicas de crowdsourcing, como apresentado no trabalho de Pérez at.al [Pérez-Rosas et al. 2017], ou protocolos que evitem a disseminação de noticias falsas, como o proposto por Qayyum at.al [Qayyum et al. 2019]. Neste último, os autores descrevem, em alto nível, sem detalher ou definir um sistema, como a tecnologia *Blockchain* pode ser usada para detectar e mitigar a disseminação de *fakenews*.

A plataforma Original.my[1] é um sistema que oference uma interface a uma plataforma *Blockchain*, que permite registros e verificações futuras de conteúdos. A plataforma permite registrar conteúdos como: declarações, relatórios, imagens ou qualquer outro tipo de documento e são geradas algumas informações associadas ao artefato, como uma assinatura digital do documento, incluindo data e hora dessa assinatura digital, por exemplo. Essas informações são enviadas para a Blockchain, incluindo data e hora do registro, código da transação e o certificado digital da transação. O ponto problemático é que, neste caso, a "chave" do registro é realizado a partir do *"permalink"* e ele pode ser facilmente manipulado e associado a outro conteúdo.

[1]https://originalmy.com

4. Descrição do framework BlockProof

O objetivo do framework *BlockProof* é compor um conjunto de métodos e processos para que seja possível registrar conteúdos Web, independente da página possuir conteúdo dinâmico ou estático, em uma Blockchain e com isso oferecer uma solução com todos os benefícios inerentes dessa tecnologia. Além disso, a solução proposta possibilita a consulta do histórico de todos os registros realizados para uma dada URL, onde a mesma será utilizada como chave de inserção e busca na *Blockchain*.

O contrato do *BlockProof*, escrito em Solidity, é composto por uma *struct* que nomeamos de *Transaction*, e ela contém todos campos que serão armazenados na Blockchain, juntamente com uma variável (*urlHash*) responsável por armazenar o hash da URL, que será informada pelo publicador ao realizar um registro na aplicação. Essa *struct* está associada a uma estrutura de dados em Solidity conhecida como *Mapping*, que é uma estrutura que relaciona uma chave à um valor. Essa estrutura de dados está associando à uma *string* (urlHash) toda a *struct* com os dados que serão enviados para a *Blockchain*.

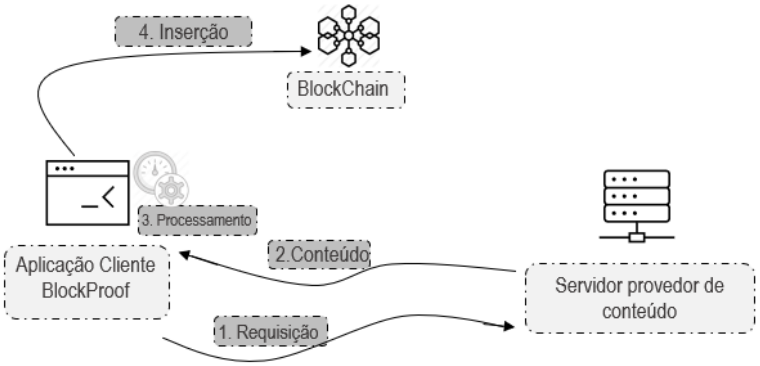


Figura 1. Fluxo de Inserção no BlockProof

Para um publicador realizar um registro no *BlockProof*, é necessário passar por quatro fases principais:

A **primeira fase (Requisição)** é a etapa na qual algum usuário, responsável pelo servidor de publicação, se autentica na ferramenta (utilizando dados de usuário e conta Ethereum existentes). Após a validação dos dados, o usuário é redirecionado para a tela de registro, onde ele deve informar o título e a URL que aponta para o conteúdo que deseja inserir na *Blockchain*. Após a normatização dos dados informados, são realizadas chamadas às funções que (i) calculam o *hash* (sha256) da URL recebida e (ii) recuperam todo código do conteúdo da página web que está associada a URL informada, o retorno da função utilizada será armazenado em uma variável que, após receber algum tratamento, será enviada para cadastro na blockchain.

Na **segunda fase (Conteúdo)**, o conteúdo da página Web recuperada (armazanada no objeto ao final da fase 1) também é submetido ao cálculo do *hash* (sha256) para que, além de reduzir o espaço de armazenamento que será alocado na Blockchain, possamos verificar se, dado o conteúdo original e o cálculo do hash apresentar o mesmo valor que fora calculado anteriormente, é possível afirmar que se trata do mesmo dado. Também estão sendo armazenadas outras informações pertinentente que irão auxiliar na validação da evidência caso necessário.

Para permitir que os registros na *Blockchian* pudessem diferenciar mudanças significativas no conteúdo da página Web, adicionou-se ao *BlockProof* o conceito de similaridade de conteúdos dos registros. Para isso, adicionou-se ao *framework* o uso do algoritmo do *Minhash*. Em [Christiani and Pagh 2017], Christiani descreve as provas matemáticas para corroborar a utilização do *Minhash* para o cálculo do grau de similaridade de conteúdos.

A **terceira fase (Processamento)** do *BlockProof*. Após a normatização dos dados são realizadas duas requisições ao conteúdo da página associada a URL para, após calcular o hash, verificar se os valores, são iguais (conteúdo estático). Caso os resultados desses *hashes* sejam diferentes, um processamento adicional para cálculo da similaridade é realizado utilizando o seguinte algoritmo [Christiani and Pagh 2017]:

- **Passo 0 - Gerar subconjuntos de tamanho fixo do conteúdo.** Neste caso, utilizamos subconjutnos de 30 caracteres. São, portanto, gerados inicialmente N subconjuntos de cada conteúdo recuperado, sendo o número de subconjuntos do conteúdo i dado por $N_i = \frac{T_i}{30}$, onde T_i representa o tamanho do conteúdo i, com $i = \{1, 2\}$;
- **Passo 1 - Transformar cada subconjunto usando uma função hash.** Cada um dos N subconjuntos são transformados em novos subconjuntos, a partir dos resultados do cálculo da função hash (neste caso foi usado o MD5);
- **Passo 2 - Armazenar o menor valor da função aplicada em cada conjunto.** Após aplicar o cálculo do hash (MD5) para cada subconjunto, é armazenado o menor dentre todos os N MD5 computados em uma lista de valores mínimos;
- **Passo 3 - Repetir os passos acima muitas vezes.** Os passos descritos acima são repetidos 100 vezes. O subconjunto transformado (contento os resultados da transformação a partir do hash na interação atual) é novamente submetido ao "Passo 1", até que 100 valores de hash mínimos sejam obtidos.

As duas listas resultantes do algoritmo descrito acima consistem de 100 valores de *hashes* mínimos, para cada um dos dois conteúdos distintos recuperados. Calcula-se então o coeficiente de semelhança entre essas listas, resultando no coeficiente de similaridade dos conteúdos. Esse coeficiente é obtido a partir do cálculo do índice de semelhança de *Jaccard*, que divide o tamanho da interseção entre as duas listas pelo tamanho a da união delas. Ou seja, considere L_1 e L_2, respectivamente, como sendo os dois conjuntos formados a partir dos elementos de *hashes* mínimos das listas de cada um dos conteúdos. O índica de semelhança de *Jaccard*, J é obtido da seguinte forma:

$$J = \frac{|L_1 \cap L_2|}{|L_1 \cup L_2|} \tag{1}$$

Por fim, na **quarta fase (Inserção)** é feito o registro dos dados na *BlockChain*.

Na Figura 2 podemos observar o fluxo de consulta. Quando o usuário deseja obter todo o histórico de um registro associado a uma determinada URL, é gerado o hash 256 dessa URL e enviada uma solicitação de consulta para a *Blockchain* para que seja obtido a quantidade de registros existentes relacionados ao *hash* 256 da URL. Esse valor é utilizado para consultar exatamente todos os dados armazenados na *Blockchain* associado à URL, e assim é obtido todo o histórico de registros associados a mesma.

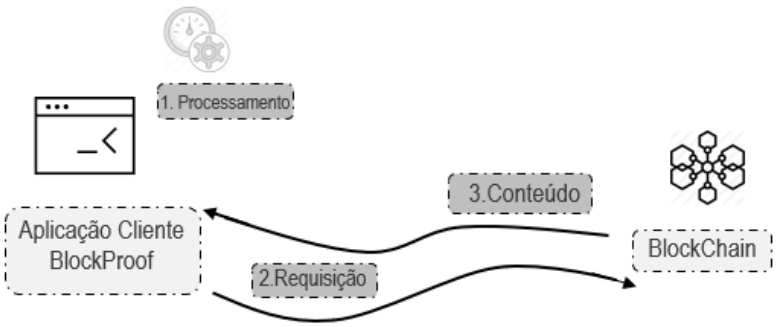


Figura 2. Fluxo de consulta no *BlockProof*

5. Resultados

Foram realizados testes de carga na aplicação *BlockProof* simulando inserções e consultas simultâneas de forma assíncrona. Em cada teste foram informados o número limite de requisições simultâneas (LRS) que se deseja simular juntamente com o incremento que será realizado a cada segundo (IRS). Foram realizadas duas simulações, uma de registros da URL e outra de consulta de todos os registros vinculados a uma dada URL, com o objetivo de simular interações de um usuário utilizando a aplicação de forma concorrente. Foram elencadas 1000 URLs, escolhidas de forma aleatória, englobando URLs estáticas e dinâmicas, e estas serviram de base para a construção dos seguintes cenários de teste:

- Cenário 1: 1000 novas inserções/consultas simultâneas, no primeiro segundo, e um incremento de 1000 updates por segundo, até chegar a (10 x 1000) inserções/consultas simultâneas.
- Cenário 2: 1000 + (9 x 1000 updates) novas inserções/consultas simultâneas, no primeiro segundo, e um incremento de 1000 updates por segundo, até chegar a (100 x 1000) inserções/consultas simultâneas.

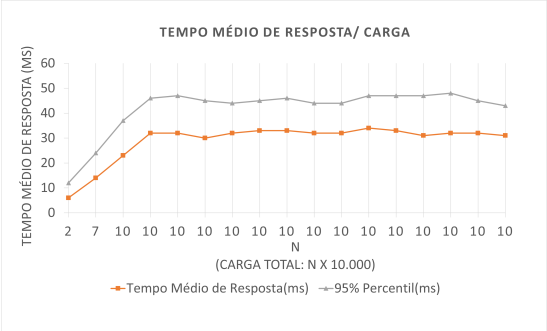
Na Figura 3 podemos observar os gráficos obtidos para os testes de carga de inserções e consultas para os cenários descritos acima. As curvas referentes ao tempo médio de resposta e ao 95% *percentil* estão plotadas nos gráficos e mostram o comportamento da aplicação ao longo dos testes. Os percentis são medidas que dividem uma amostra de valores, ordenados de forma crescente, em cem partes frequentemente utilizado para medir o grau de aceitação de algo. O tempo médio de resposta indica qual o tempo médio de retorno da requisição enviada para a aplicação ser efetivada para o IRS correspondente.

Para uma carga **N** igual a 2, por exemplo no gráfico correspondente ao Cenário Registro 1, é apresentado um tempo médio de resposta igual a 6ms e um 95% percentile de 12ms. Ou seja, para o servidor do BlockProof tentar atender a 2 x 10.000 usuários, 95% dos usuários, terá a resposta da sua requisição de registro atendida em 12ms. Análises similar podem ser realizadas para interpretar os demais pontos plotados nos gráficos resultantes subsequentes.

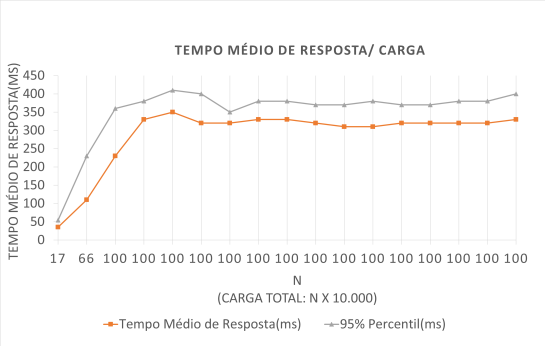
Durante o intervalo de tempo de execução dos testes, foram reportados em média 0 falhas por segundo (FS), ou seja todas as requisições/consultas realizadas foram suportadas pela aplicação.

Figura 3. Gráficos dos Resultados dos Registros e Consultas Realizados em 2 Cenários

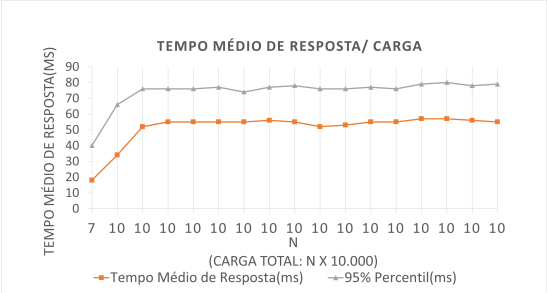
(a) Cenário Registro 1



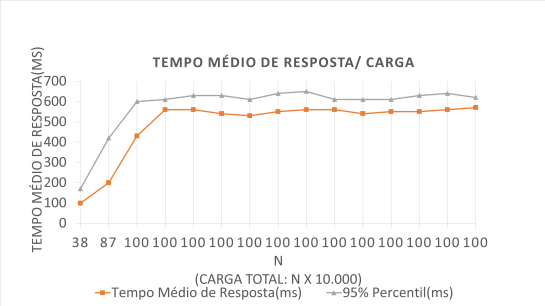
(b) Cenário Registro 2



(c) Cenário Consulta 1



(d) Cenário Consulta 2



Referências

Christiani, T. and Pagh, R. (2017). Set similarity search beyond minhash. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1094–1107.

da Silva Pereira, K. and de Oliveira, F. M. (2019). Perícia forense computacional e crimes cibernéticos. *Revista Interdisciplinar Pensamento Científico*, 5(2).

Junior, A. S. R. and Alavarse, G. M. A. (2014). Crimes virtuais: Um desafio para perícia. *Revista Diálogos & Saberes*, 9(1).

Nakamoto, S. (2019). Bitcoin: A peer-to-peer electronic cash system. Technical report, Manubot.

Pérez-Rosas, V., Kleinberg, B., Lefevre, A., and Mihalcea, R. (2017). Automatic detection of fake news. *arXiv preprint arXiv:1708.07104*.

Pinheiro, E. P. (2006). Crimes virtuais: uma análise da criminalidade informática e da resposta estatal. Vol. 5.

Qayyum, A., Qadir, J., Janjua, M. U., and Sher, F. (2019). Using blockchain to rein in the new post-truth world and check the spread of fake news. *IT Professional*, 21(4):16–24.

Sieber, U. (1998). Legal aspects of computer-related crime in the information society: Comcrime study. *European Commission*.