BCDP: A Blockchain-based Credible Data Publishing System

Fei-Qiang Liao[1], Jian-Feng Wang[1], Jian Shen[2]
[1] State Key Laboratory of Integrated Service Networks (ISN), Xidian University, China
[2] School of Computer and Software, Nanjing University of Information Science and Technology, China
fqliao@yeah.net, jfwang@xidian.edu.cn, s_shenjian@126.com

Abstract

With the advent of the era of big data, how to publish electronic data in a manner of credible become a challenge. Traditional data publishing schemes either require a trusted third party (TTP) that is responsible to collect and publish data, or cannot support similar data duplicate detection. The result is that they are vulnerable to distributed denial of service (DDoS) attacks and suffer from an unnecessary waste of storage space. To address the above issue, in this paper, we propose a blockchain-based credible data publishing model (BCDP), which enjoys the properties of decentralized, tamper resistant and anti-DDoS attacks. Then, we construct a two-layer near duplicate detection algorithm with counting bloom filter (CBF) that can detect near duplicate data based on BCDP. In addition, we exploit an academic publishing system based on BCDP, which introduces an academic scoring mechanism to achieve reward and punishment for academic publishing.

Keywords: Data publishing, Blockchain, Near duplicate detection, Counting bloom filter

1 Introduction

Data publishing is an important part of medical, academic and commercial applications [1-3]. For example, in California, licensed hospitals are required to publish their patient records on the online system for general medical research [4]. The model of traditional data publishing system [5] is illustrated in Figure 1. In the data collection stage, the data publisher collects data from the data owner, such as Alice and Bob. And in the data publishing stage, the data publisher publishes the collected data to the public or to data miners, referred to the data recipient, who can utilize the published data. For example, a journal collects research data from authors and publishes the research data to online systems to serve researchers. In this example, the journal is the data publisher, the author is the data owner and the researcher is the data recipient.

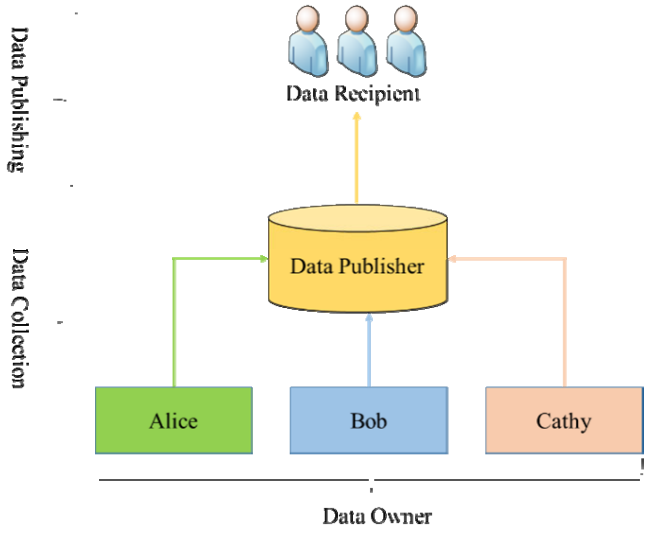


Figure 1. The model of traditional data publishing system

Despite the fact that there are a lot of data publishing online systems, most of them demand a TTP and control over published data [6]. There are some issues need to be discussed. First of all, published data in centralized online systems is costly to maintain, especially when large capital expended in hardware and software deployment. Besides, a single point failure and DDoS attacks occur inevitably. Second, administrators will be given privileges to have complete control over published data that may be corrupted and lost intentionally or unintentionally. At last but not least, near duplicate detection prior to data publishing needs to be considered, such as academic publishing needing to deal with the issue of duplicate publication.

With the understanding of Bitcoin, blockchain has gradually aroused people's attention. Blockchain serves as an innovative technology with the properties of decentralized and tamper resistant, which can provide a new solution to reconstruct the data publishing system. In this paper, the main contributions can be summarized as three folds:

*Corresponding Author: Fei-Qiang Liao; E-mail: fqliao@yeah.net
DOI: 10.3966/160792642019032002001

- We propose a new blockchain-based credible data publishing scheme, which has the properties of decentralized, tamper resistant and anti-DDoS attacks.
- In order to save storage overhead and avoid near duplicate data being stored multiple times, we construct a two-layer near duplicate detection algorithm with counting bloom filter. Finally, we provide simulation tests to prove that the algorithm has high efficiency and stability.
- To present a specific BCDP-based Dapp, we construct and exploit an academic publishing system. Furthermore, we propose an academic scoring mechanism to achieve reward and punishment for academic publishing.

1.1 Related Work

Data publishing goes back to the early days of the public network and has been well studied ever since. Publius [7] is a web-based publishing system which provides methods for updating or deleting the published data. In Publius, an automatic tamper checking mechanism is implemented, but a static list of documents on available servers and the index is not protected. Moreover, many servers need to be maintained since its way of development is centralized. Free Haven [8] is a publishing system that provides a dynamic network and ensures the availability of each document for a publisher-specified lifetime. Because of inefficient broadcasts for communication inherently implemented, it is unsuitable for wide deployment. Freenet [9] is used to solve survivable issues and information privacy problem. However, Freenet need to provide search mechanisms and develop more protection against DDoS attacks. Another approach used anonymity tunnels for each different user in RTAP [10]. The system provides anonymity to all participants which is defined in RTAP. Recently, a prediction-based scheme for data stream with time constraint over Internet of Things (IoT) was proposed [11], which presented a new methodology for deploying real-time data stream based on prediction mechanism in IoT environments.

However, the issues of near duplicate detection have received no attention in these systems. On the other side, there are some algorithms proposed for near duplicate detection. A shingling algorithm was firstly proposed to measure the similarity between two web pages by a sequence of fingerprints, namely, shingles [12]. The works [13-14] presented the use of sentence level features in near duplicate detection. Mitzenmacher et al. [15] proposed an odd Sketches which is a compact binary sketch to estimate Jaccard similarity to provide a highly space-efficient estimator for sets of high similarity. Varol and Hari [16] proposed a hybrid approach to embed Jaro distance and statistical results of word usage frequency for near duplicate detection. Recently, Liu et al. [17] proposed a data deduplication scheme, which measured data similarity by taking advantage of counting bloom filter. In our scheme, we utilize the advanced technology to construct our two-layer near duplicate detection algorithm.

1.2 Organization

The rest of this paper is organized as follows. We present some preliminaries in Section 2. We propose the detailed description on BCDP in Section 3. The concrete academic publishing system based on BCDP is given in Section 4. We provide security and efficiency analysis in Section 5. Finally, a brief conclusion is given in Section 6.

2 Preliminaries

2.1 Blockchain

Blockchain is a decentralized public ledger and consists of block that is made of block header and a long list of transactions. The block header contains version, timestamp, nonce, difficulty, previous block hash (Prehash) and the root hash of Merkle tree which is computed by all transaction records stored in the block. One block is mined by miners who fiercely compete on the peer to peer network to solve hard problems in the most efficient way, then the new block will be concatenated to blockchain by a Prehash [18], as illustrated in Figure 2. When global network mining power increases, so does the difficulty for mining a new block. Therefore, blockchain as a secure public ledger is maintained by all participants through the distributed network without a TTP, such as a central bank or a financial institution.

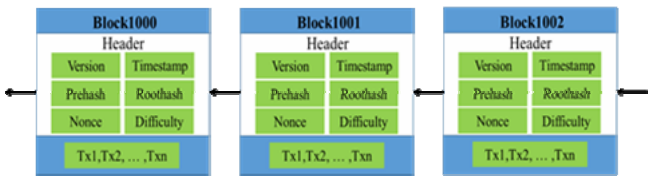


Figure 2. Blockchain structure

Ethereum is a blockchain-based decentralized platform which runs smart contracts in a decentralized virtual machine known as Ethereum Virtual Machine (EVM). Smart contracts are written in EVM bytecode that is a Turing-complete bytecode language rather than a simple scripting language implemented in Bitcoin [19], which extends the capabilities of blockchain to support developers to build Decentralized Applications (DApps) beyond financial applications.

2.2 Partition Technology

Partition technology can be categorized into two types: Fixed-Sized Partition technology and Variable-Sized Partition technology. Zhang et al. [20] proposed Asymmetric Extremum Content-Defined Chunking (AE-CDC), which belongs to Variable-Sized Partition technology and is known as the state-of-the-art partition technology. In AE-CDC scheme, chunk boundaries are determined only based on local content and bytes are regarded as numbers. The properties of AE-CDC can be summarized as follows:

(1) AE-CDC can reduce chunk-size variance and eliminate low-entropy strings.

(2) Fewer operations are performed on AE-CDC scheme, which reduce computational overhead compared with previous CDC schemes [21-22].

2.3 Bloom Filter

Bloom 0 proposed a kind of data structure, namely, bloom filter (BF), which has already been widely used to check whether an element is in a set or not. A bloom filter is initialized by setting all bits in a bit array of m to 0 and k independent hash functions $h_i : \{0,1\}^* \rightarrow [1,m]$, where $i \in [1,k]$, as is shown in the left part of Figure 3. To insert an element into the bloom filter, feed it to all of the k independent hash functions to obtain k array positions. Then, all these positions will be set to 1. We note that some positions have been set to 1 over one time. Given any element x, checking whether it belongs to the set, when bits in all positions of $h_i(x)$ are 1, we can determine that x is an element in the set with a false positive. Otherwise, x is not an element of the set. As given in 0, the false positive of bloom filter is $P_f(1-e^{-km/m})^k$, where k is the number of hash functions, m is the length of the bloom filter and n is the number of elements.

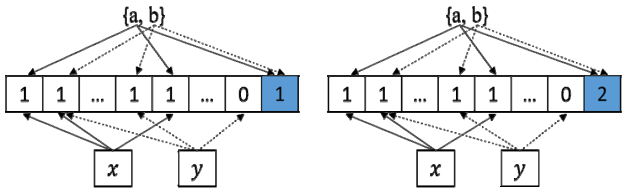


Figure 3. Traditional bloom filter and counting bloom filter

However, an obvious drawback of traditional bloom filter is that an element cannot be deleted from bloom filter. Fan et al. [25] proposed a counting bloom filter (CBF) to solve this problem. In the proposed scheme, a counter is introduced to support the element deletion operation, which indicates the number of times that a position is set to 1. The operation of checking whether an element belongs to the set in the counting bloom filter is very similar to that in the traditional bloom filter, as is illustrated in the right part of Figure 3.

3 Problem Formulation

3.1 System Model

The proposed BCDP consists of three actors: the data owner, the data publisher, and the data recipient. The model of BCDP is shown in the Figure 4.

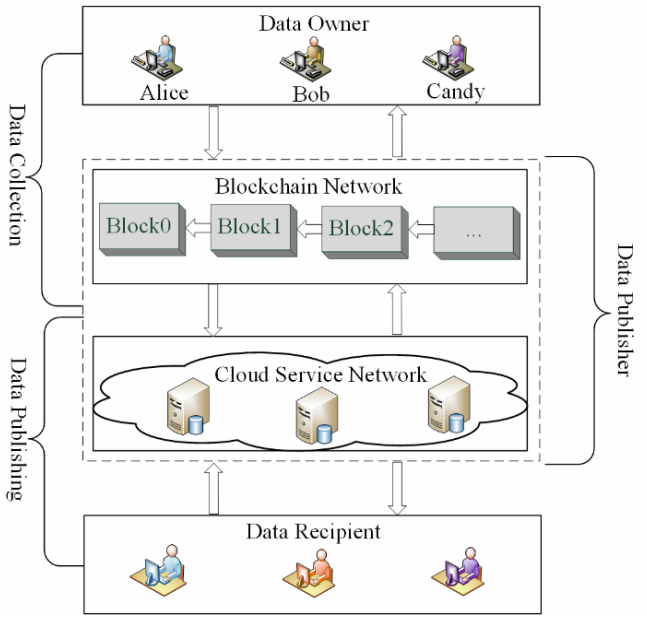


Figure 4. The model of BCDP

Data owner. The data owner is an actor who wants his data to be published. He cannot be deemed completely trusted. In order to gain some benefits, he may engage in publishing their duplicate data.

Data publisher. The data publisher is responsible for publishing data. In BCDP, the data publisher is the blockchain-based data publishing platform which consists of blockchain network and cloud service network. Data will be stored in cloud service network and its associated metadata will be stored in blockchain network.

Data recipient. The data recipient refers to the actor who can query metadata from blockchain network and download data from cloud service network.

3.2 The Main Idea of BCDP

The high description for BCDP can be formulated as follows: The data owner provides his data along with associated metadata to the data publisher. The data publisher will carry out near duplicate detection. If the detection is not passed, data will be rejected. Otherwise, the metadata will be published in blockchain network, whilst the data will be published in cloud service network. In this paper, our main idea is that we take advantage of the above technologies of blockchain and Ethereum to provide a global decentralized and permanent ledger. In order to reduce the burden of data

storage for blockchain, we solve this challenge by only publishing a small amount of metadata about data in blockchain network and data will be published in cloud service network. This is done by sending a light transaction, including metadata information. The content of metadata depends on specific applications, for example, data title, data tag, data link, and timestamp. Therefore, the data recipient can obtain data from cloud service network by querying blockchain transaction including data link.

4 The Proposed Blockchain-based Academic Publishing System

In this section, we firstly present a data duplicate detection method used in the proposed construction. Then, we present the proposed academic publishing system based on BCDP.

4.1 Two-layer Near Duplicate Detection Algorithm with CBF

We construct a two-layer near duplicate detection algorithm with CBF to detect near duplicate data. Consider that the metadata is published in blockchain network and data content is published in cloud service network, our data detection algorithm is designed with two layers structure as follows:

Metadata detection. In order to reduce computation overhead, data title will be detected firstly by blockchain network.

Data detection. Data detection serves as the second layer of the algorithm, which mainly detects data tag and data similarity by cloud service network.

The concrete construction of detection algorithm involves the following steps:

TitleTest (TiT). The data owner inputs the data title TiT, whether there is a duplicate data on the blockchain network will be detected by comparing data title. If there is $TiT' = TiT$, the output of the algorithm is True. Otherwise, the output of the algorithm is False.

TagGen (m_i). The data owner divides data M into blocks $\{m_i\}_{i=1}^{n}$ by AE-CDC algorithm. Then each block tag $T_i = H(m_i)$ will be computed, where H is a cryptographic hash function. Finally, the data owner utilizes each T_i as the input of a CBF, then the CBF serves as the data tag T. As is shown in Algorithm 1.

SimTest ($(T, \{T_j'\}_{j=1}^{s}, d)$. While receiving the data tag T from the data owner, the cloud server firstly detects whether there is the same data on cloud by comparing data tag. If there is not $T_j' = T$, the cloud server will compute the data tag similarity $D_j = sim(T_j', T)$ (We will analyze concrete similarity algorithms sim in Section 5). Then the cloud server compares D_j with a preset trapdoor d. If $D_j > d$, the output of algorithm is True. Otherwise, the output of algorithm is False. As is shown in Algorithm 2.

ConTest (T, M). If $SimTest(T, T_j', d) =$ False, the data owner will be requested for uploading data M to check consistency. The cloud server reconstructions $\{T_i\}_{i=1}^{n}$ based on M, then puts $\{T_i\}_{i=1}^{n}$ into a CBF to get data tag T'. Only if $T' = T$, the output of the algorithm is True. Otherwise, the output of the algorithm is False. If the output is True, the cloud server will return a data link L which is used to retrieve the data. Otherwise, the data owner will be rejected.

Algorithm 1. The work of the data owner

Input: Data M

Output: the data block $\{m_i\}_{i=1}^{n}$, the data tag T

1. $\{m_i\}_{i=1}^{n} \leftarrow AE - CDC(M)$
2. **for** $i = 1$ to n **do**
3. $T_i \leftarrow H(m_i)$
4. **end for**
5. $T \leftarrow CBF(\{T_i\}_{i=1}^{n})$
6. **return** $\{m_i\}_{i=1}^{n}$, T

Algorithm 2. The work of the cloud server

Input: data tag T, existing data tags $\{T_j'\}_{j=1}^{m}$, preset trapdoor s

Output: the outcome of near duplicate detection

```
if T in {T'_j}_{j=1}^{s} then
reject to the data owner
else
  for j = 1 to m do
    if SimTest(T, T'_j, d) = True then
      reject to the data owner
    else
      requests for uploading data M
      if ConTest(T, M) = True then
        return data link L to the data owner
      else
        reject to the data owner
      end if
    end if
  end for
end if
```

4.2 The Concrete Construction

In this section, we present an academic publishing system based on BCDP. The goal of the system is to provide a decentralized, credible and global academic publishing service. The architecture of the system is shown in the Figure 5. There are three different entities involved in the system: the authors, the reviewers, and the researchers.

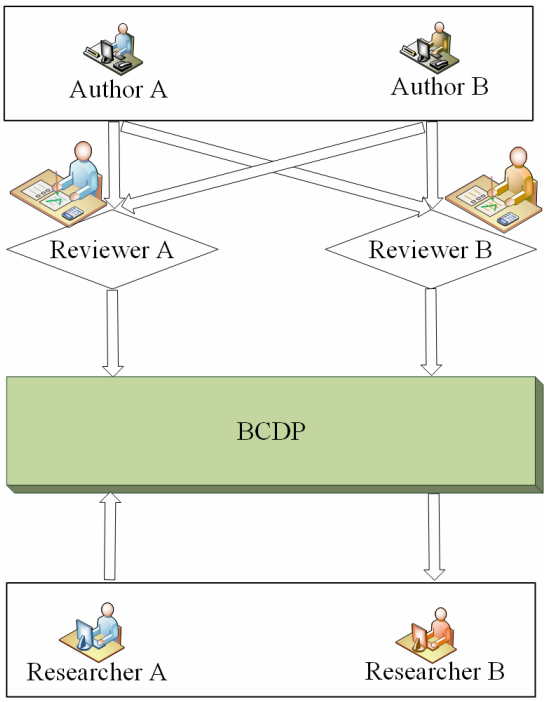


Figure 5. Architecture of academic publishing system

Author. The author will serve as the data owner, who expects his paper to be published and cannot be deemed completely trusted. In order to increase the number of papers or gain economic benefits, he may engage in duplicate publication or academic fraud.

Reviewer. The reviewer who belongs to some specific journals as part of the data publisher. He is responsible for reviewing papers. If a paper is passed in reviews, it will be delivered to the system for publishing.

Researcher. The researcher belongs to the data recipient. He can obtain papers by querying from the system.

In order to deal with the issue of duplicate publication, we propose an academic scoring mechanism. We mainly introduce academic scores for the author, which can evaluate one author's academic level. The higher academic score you get, the greater academic contribution you made.

The system settings are as follows: s denotes the author's academic scores, k_1 denotes academic scores for publishing one paper, k_2 denotes academic scores for duplicate publication and k_3 denotes academic scores for academic fraud. The value of k_1, k_2 and k_3 depend on the level of journal. In addition, k_1, k_2 and k_3 should be satisfied:

$$k_3 < k_3 < k_3 \tag{1}$$

The procedure of academic scoring mechanism as follows:

(1) Initialize: let $s = 0$.

(2) If academic scores less than zero, the author will be banned from publishing.

(3) If a paper is successful published, then $s = s + k_1$.

(4) If duplicate publication is detected, then $s = s - k_2$.

(5) If academic fraud is confirmed, then $s = s - k_3$.

The details construction of academic publishing system are given as follows.

System setup. The necessary parameters are initialized in system setup phase. The author gets $\text{author}_{\text{id}}$, $\text{author}_{\text{name}}$, academic scores s (initialized to 0), $\text{author}_{\text{password}}$ and the reviewer gets $\text{reviewer}_{\text{id}}$, $\text{reviewer}_{\text{password}}$, and other system parameters are built in.

Publishing phase. In the system, the metadata about paper will be stored in blockchain network and the full text will be stored in cloud service network. The metadata contains paper title iT, paper tag T, paper link L, author id Ω_{id}, journal name N and timestamp t. When a paper is to be published, whether the author has permission to publish papers will be detected firstly. That is, at least one author's academic scores $s < 0$, then the paper will be rejected. Otherwise, duplicate publication detection will be performed on the paper. The duplicate publication detection is based on two-layer near duplicate detection algorithm. According to the following three conditions:

$$\begin{cases} TiT' = TiT \\ T' = T \\ SimTest(T, T'_j, d) = True \end{cases} \tag{2}$$

If any of the three conditions are satisfied, the paper will be identified as duplicate publication.

Querying phase. The author can query their published papers and academic scores. The researcher can query published papers and download relevant full text from cloud service network by TiT or N. If academic fraud behavior is found by the researcher, the published paper will be revoked and its associated authors' academic scores $s = s - k_3$.

5 Security and Efficiency Analysis

5.1 Security Analysis

In this section, we present security analysis of academic publishing system based on BCDP.

Theorem 1. *The academic publishing system satisfies the security requirement of fairness.*

Proof. The system is based on blockchain technology. Therefore, a global decentralized academic dataset can be provided to support for duplicate publication detection. When academic fraud is found, the author will be uniformly punished with academic scoring mechanism, which provides an open and transparent way to deal with academic misconduct.

Theorem 2. *The academic publishing system satisfies*

the security requirement of tamper resistant.

Proof. Tampering with the data stored in blockchain network requires significant proportion of the computation power (typically 51%), which is almost impossible to do [26]. Hence, the metadata cannot be corrupted or lost. We note that the full text stored in cloud service network. Assume that cloud servers tamper or delete the full text, while the paper tag stored in blockchain remains unchanged. As a result, the malicious behavior will be found by the comparison of paper tag.

Theorem 3. *The academic publishing system satisfies the security requirement of Anti-DDoS attacks.*

Proof. The execution of smart contracts is restricted to execution fees, which can reward miners for maintaining the Ethereum network and protect against DDoS attacks [27]. Meanwhile, cloud service network could be resistant to DDoS attacks by some defense solutions [28]. Hence, our system can achieve the corresponding security for publishing data.

5.2 Comparison

In this section, we compare the proposed BCDP with Publius [7] and RTAP [10]. Table 1 presents comparison of the three data publishing systems.

Table 1. Comparison of three data publishing systems

	Publius [7]	RTAP [10]	BCDP
Decentralized	No	Yes	Yes
Near duplicate detection	No	No	Yes
Tamper resistant	Yes	No	Yes
Anti-DDoS attacks	No	Yes	Yes

Firstly, all of the three systems can provide the function of data publishing. Secondly, our proposed BCDP is based on blockchain that is a decentralized network without a TTP, and tamper resistant is built-in. However, all the other ones can only obtain one the property for decentralized or tamper resistant. Besides, Publius and RTAP cannot provide the mechanism of near duplicate detection. At last, BCDP also has the ability of anti-DDoS attacks.

5.3 Performance Evaluation

We provide an experimental evaluation for the academic publishing system. The configuration of software and hardware environment are shown in the Table 2.

We implement the system as a Dapp running on the blockchain-based Ethereum testnet. Ethereum client is EthereumJS TestRPC and Ethereum development framework is Truffle. The front-end programming language is HTML, CSS, JavaScript and the back-end programming language is Solidity that is used to write smart contracts. In addition, cloud service is simulated with Python and MongoDB.

Table 2. Configuration of software and hardware environment

Category	Configuration
CPU	Intel (R) Core (TM) CPU i3-3240 @3.4 0GHz, 4 core
OS	Linux (x86 64, kernel version: 4.4.0-31-generic)
RAM	4GB
HDD	TOSHIBA DT01ACA050
Software	TestRPC v3.0.3, Truffle v2.2.1, Python3.4, MongoDB2.4.9

In order to provide the entrance of user interaction with the smart contract, we have exploited a web interface for the Dapp. Users need establish an Ethereum node in their computer to access it. The web interface uses JavaScript functions to interact with the user's local node, which stores the blockchain with a record of all transactions and contracts. In the Dapp, the full text will be uploaded to cloud service network and metadata will be recorded in blockchain network. We provide an academic scoring mechanism for dealing with duplicate publication. Each journal can deal with the relevant papers published when any academic fraud is found.

In the system, the metadata information takes up about 360 bytes, we get about 1 KB per transaction by adding some extra data. In that case, we could store one thousand papers when the metadata information reached 1 MB in size, as illustrated in Figure 6.

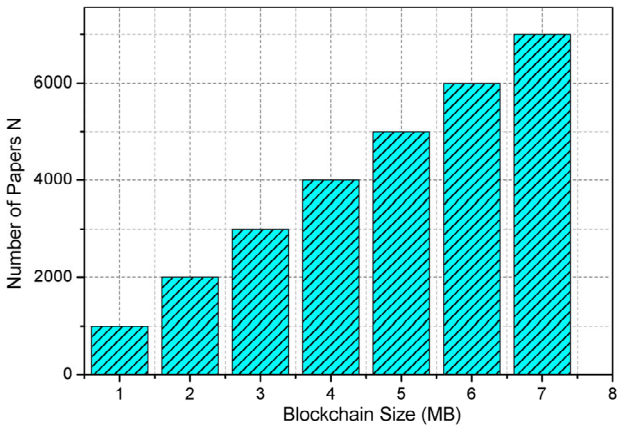
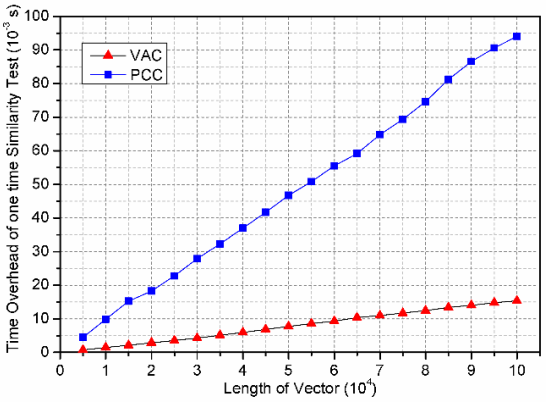


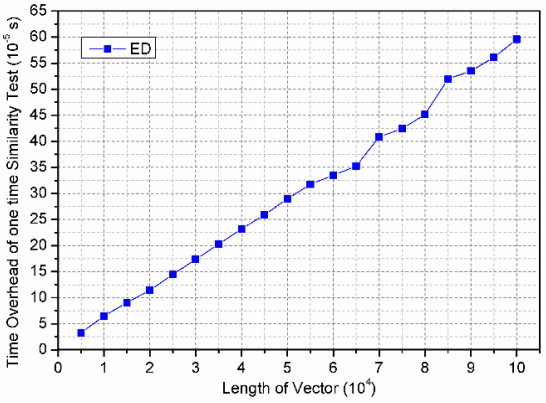
Figure 6. Metadata information stored vs blockchain sizes

As we mentioned in Section 4, similarity estimation is another kernel in our two-layer near duplicate detection algorithm. There are several similarity algorithms to estimate similarity between two vectors, such as Vectorial Angle Cosine (VAC), Pearson Correlation Coefficient (PCC), Euclidean Distance (ED) and Normalized Euclidian Distance (NED). We first evaluate the time overhead of these four similarity algorithms. In our experiments, we can randomly generate vectors of length 5000 to 100000 and each dimension of 0 to 1 by vector generation function of Python, then run these four similarity algorithms

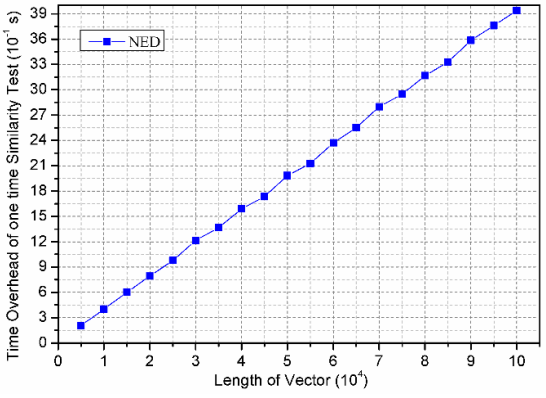
multiple times to get the average run time for each algorithm, as indicated in Figure 7. It is obvious that the time overhead of VAC is faster than that of NED by two orders of magnitude. Meanwhile, VAC is much less than PCC in time overhead. We also note that ED has the least time overhead in these similarity algorithms. However, it is very hard to estimate the similarity by ED with data in different size, since the output of ED is greatly influenced by the data size in lack of normalization. As a result, VAC is used as similarity estimation algorithm in our two-layer near duplicate detection algorithm.



(a) Time overhead on VAC and PCC



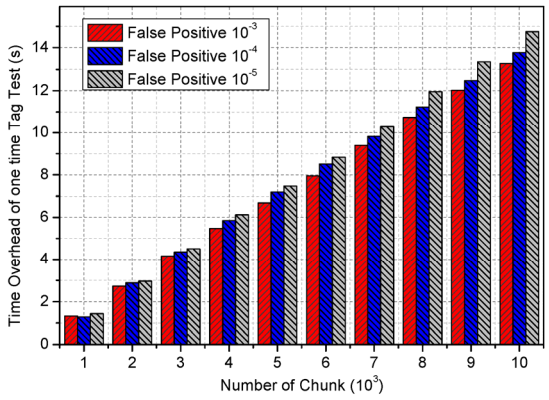
(b) Time overhead on ED



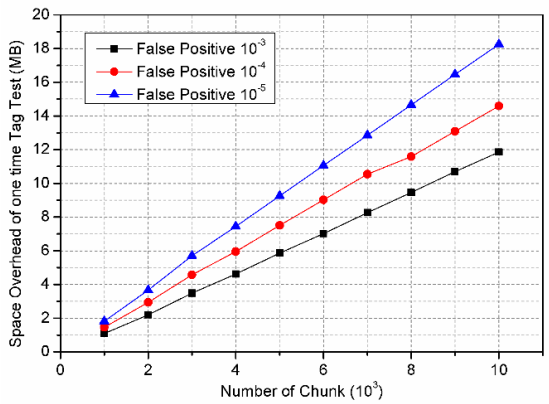
(c) Time overhead on NED

Figure 7. Time overhead for four similarity algorithms

Next, we do simulations to evaluate the effect of number of chunks n and false positive P_f on the tag generation algorithm. The result is illustrated in Figure 8. We can notice that time and space overhead will increase along with the increase of n, while increases with the decrease of P_f. The reason is that the lower false positive leads to the larger CBF. As a result, we should choose a suitable value for P_f to get the balance between overhead and accuracy according to our needs.



(a) Time overhead for Tag Generation Algorithm



(b) Space overhead for tag generation algorithm

Figure 8. Time and space overhead for tag generation algorithm

Finally, we also adopt a realistic dataset Fslhomes [29] that has been widely used for other near duplicate data deletion studies. We compare the value of similarity for different window size in AE-CDC and false positive in CBF, as is shown in Figure 9. It is important to note that the similarity value fluctuates around 30% according to window size and false positive. Hence, our near duplicate algorithm satisfies high efficiency and stability.

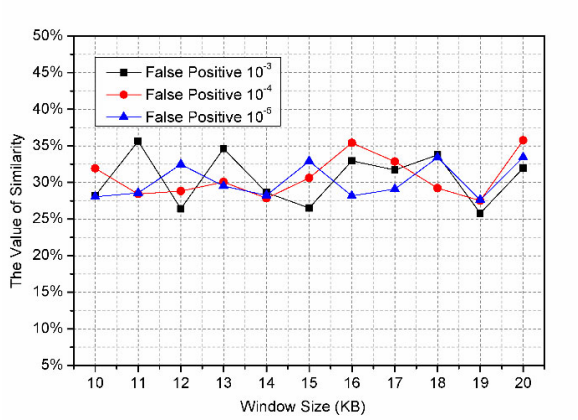


Figure 9. Comparison of the value of similarity for different window size and false positive

6 Conclusion

In this paper, we propose a blockchain-based credible data publishing scheme. Specifically, we adopt blockchain network and cloud service network as the data publishing platform, while a two layer near duplicate data detection algorithm is constructed. Then, we construct an academic publishing system on BCDP with a academic scoring mechanism, which can achieve reward and punishment for academic publishing. Finally, the academic publishing system is implemented on the blockchain-based Ethereum testnet.

References

[1] B. J. Yolles, J. C. Connors, S. Grufferman, Obtaining Access to Data from Government-sponsored Medical Research, *N Engl J Med*, Vol. 315, No. 1, pp. 1669-1672, December, 1986.

[2] G. Hughes, Sharing Research Data, *Emergency Medicine Australasia Ema*, Vol. 29, No. 1, pp. 4-5, February, 2017.

[3] Z. Fan, Dynamic Transmission Power Switch for Fast Data Collection in Duty-cycled Sensor Networks, *International Journal of Ad Hoc and Ubiquitous Computing*, Vol. 24, No. 3, pp. 173-182, May, 2017.

[4] G. Sondermeyer, L. Lee, D. Gilliss, D. Vugia, Coccidioidomycosis-associated Hospitalizations, *Emerging Infectious Diseases*, Vol. 19, No. 10, pp. 1590-1597, October, 2013.

[5] K. Wang, R. Chen, B.-C. Fung, P.-S. Yu, Privacy-preserving Data Publishing: A Survey of Recent Developments, *ACM Computing Surveys (CSUR)*, Vol. 42, No. 4, pp. 14, June, 2010.

[6] V. Jacynycz, A. Calvo, S. Hassan, A. A. Sánchez-Ruiz, Betfunding: A Distributed Bounty-based Crowdfunding Platform over Ethereum, *Proceedings of the Distributed Computing and Artificial Intelligence, 13th International Conference*, Sevilla, Spain, 2016, pp. 403-411.

[7] M. Waldman, A. D. Rubin, L. F. Cranor, Publius: A Robust, Tamper-evident, Censorship-resistant Web Publishing System, *Proceedings of the 9th USENIX Security Symposium*, Denver, CO, 2000, pp. 59-72.

[8] R. Dingledine, M. J. Freedman, D. Molnar, The Free Haven Project: Distributed Anonymous Storage Service, *Designing Privacy Enhancing Technologies, Springer Berlin Heidelberg*, New York, NY, 2001, pp. 67-95.

[9] I. Clarke, S. G. Miller, T. W. Hong, O. Sandberg, B. Wiley, Protecting Free Expression Online with Freenet, *IEEE Internet Computing*, Vol. 6, No. 1, pp. 40-49, August, 2002.

[10] O. Hermoni, N. Gilboa, E. Felstaine, S. Dolev, Rendezvous Tunnel for Anonymous Publishing, *Peer-to-Peer Networking and Applications*, Vol. 8, No. 3, pp. 352-366, May, 2015.

[11] D.-J. Chiang, Real-Time Data Delivering Based on Prediction Scheme over Internet of Things, *Journal of Internet Technology*, Vol. 18, No. 2, pp. 395-405, July, 2016.

[12] A. Z. Broder, S. C. Glassman, M. S. Manasse, G. Zweig, Syntactic Clustering of the Web, *Computer Networks and ISDN Systems*, Vol. 29, No. 8, pp. 1157-1166, July, 1997.

[13] Y.-S. Lin, T.-Y. Liao, S.-J. Lee, Detecting Near-duplicate Documents Using Sentence-level Features and Supervised Learning, *Expert Systems with Applications*, Vol. 40, No. 5, pp. 1467-1476, April, 2013.

[14] J.-B. Feng, S.-L. Wu, Detecting Near-duplicate Documents Using Sentence Level Features, *Proceedings of the International Conference on Database and Expert Systems Applications*, Valencia, Spain, 2015, pp. 195-204.

[15] M. Mitzenmacher, R. Pagh, N. Pham, Efficient Estimation for High Similarities Using Odd Sketches, *Proceedings of the 23rd International Conference on World Wide Web*, Seoul, Korea, 2014, pp. 109-118.

[16] C. Varol, S. Hari, Detecting Near-duplicate Text Documents with a Hybrid Approach, *Journal of Information Science*, Vol. 41, No. 4, pp. 405-414, August, 2015.

[17] J.-F. Liu, J.-F. Wang, X.-L. Tao, J. Shen, Secure Similarity-based Cloud Data Deduplication in Ubiquitous City, *Pervasive and Mobile Computing*, Vol. 41, pp. 231-242, October, 2017.

[18] M. Pilkington, *Blockchain Technology: Principles and Applications*, Social Science Electronic Publishing, 2016.

[19] G. Wood, *Ethereum: A Secure Decentralised Generalised Transaction Ledger*, Ethereum Project Yellow Paper, 2014.

[20] Y.-C. Zhang, H. Jiang, D. Feng, W. Xia, M. Fu, F.-T. Huang, Y.-K. Zhou, AE: An Asymmetric Extremum Content Defined Chunking Algorithm for Fast and Bandwidth-efficient Data Deduplication, *Proceedings of the 34th IEEE Conference on Computer Communications*, Kowloon, Hong Kong, 2015, pp. 1337-1345.

[21] N. Bjørner, A. Blass, Y. Gurevich, Content-dependent Chunking for Differential Compression, the Local Maximum Approach, *Journal of Computer & System Sciences*, Vol. 76, No. 3, pp. 154-203, May, 2010.

[22] M. O. Rabin, *Fingerprinting by Random Polynomials*, Center for Research in Computing Techn, Aiken Computation Laboratory, Univ, 1981.

[23] B. H. Bloom, Space/time Trade-offs in Hash Coding with Allowable Errors, *Communications of the ACM*, Vol. 13, No. 7, pp. 422-426, July, 1970.

[24] B. Andrei, M. Michael, Network Applications of Bloom Filters: A Survey, *Internet Math*, Vol. 1, No. 4, pp. 485-509, November, 2004.

[25] L. Fan, P. Cao, J. M. Almeida, A. Z. Broder, Summary Cache: A Scalable Wide-area Web Cache Sharing Protocol, *IEEE/ACM Trans*, Vol. 8, No. 3 pp. 281-293, June, 2000.

[26] R. Dennis, G. Owen, Rep on the Block: A Next Generation Reputation System Based on the Blockchain, *Proceedings of the 2015 10th International Conference for Internet Technology and Secured Transactions*, London, England, 2015, pp. 131-138.

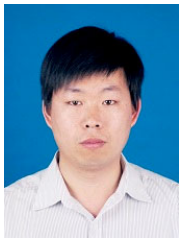
[27] N. Atzei, M. Bartoletti, T. Cimoli, A Survey of Attacks on Ethereum Smart Contracts (SoK), *Proceedings of the International Conference on Principles of Security and Trust*, Uppsala, Sweden 2017, pp. 164-186.

[28] O. Osanaiye, K. K. R. Choo, M. Dlodlo, Distributed Denial of Service (DDoS) Resilience in Cloud: Review and Conceptual Cloud DDoS Mitigation Framework, *Journal of Network & Computer Applications*, Vol. 67, pp. 147-165, May, 2016.

[29] Fslhomes, http://tracer.filesystems.org/traces/fslhomes/2014.

Biographies

Fei-Qiang Liao received his bachelor in communication engineering from Hubei University, China. He is currently working toward the master degree in information security in Xidian University, China. His research interests include applied cryptography and blockchain.



Jian-Feng Wang received his M.S. degree on mathematics and Ph.D. degree in cryptography from Xidian University in 2013 and 2016, respectively. He is a lecturer at School of Cyber Engineering of Xidian University. His research interests include applied cryptography and data security.

Jian Shen received the B.E. degree from Nanjing University of Information Science and Technology, China and the M.E. degree in Computer Science from Chosun University, Korea. He is a professor at Nanjing University of Information Science and Technology. His research interests include network security, mobile computing and networking.