IEEJ Journal of Industry Applications
Vol.6 No.4 pp.252–257 DOI: 10.1541/ieejjia.6.252

Paper

A Decentralized Trusted Timestamping Based on Blockchains

Yuefei Gao[*a)] Non-member, Hajime Nobuhara[*] Member

(Manuscript received May 30, 2016, revised Jan. 10, 2017)

Decentralized trusted timestamping based on blockchains is used to protect a large variety of digital data. At present, in the case of trusted timestamping services, such related information is not included in the OP_RETURN field of a Bitcoin's transaction chain, which has a limited size (40 bytes). When OP_RETURN is extended, (e.g., with multiple OP_RETURN fields) the transaction is rejected by the Bitcoin network. We propose storing data in the blockchain by encoding into Bitcoin addresses. The transactions created by the proposed method are similar to the usual transactions; therefore, they will not be rejected. The proposed method expands the storage space to a maximum of N*20 bytes, thereby enabling the storage of additional information (e.g., file names, creator names, and keywords) as well as file hashes. We performed experiments with a picture and its copyright information. We set N = 3, resulting in storage of 60 bytes of data. The experimental results indicate that the proposed method can timestamp a file in an average of 24 min at a possible cost of 0.24 USD. We believe that the proposed method can prove the existence and integrity of a digital file, which is helpful in copyright protection.

Keywords: trusted timestamping, blockchain, digital data protection

1. Introduction

Over the past few decades, the Internet has become increasingly popular in people's daily lives. Hundreds and thousands of digital data (such as articles, songs, and videos) are created and shared on the Internet by people every day. However, digital data easily be tampered with by hackers, which makes it difficult to prove the original time of creation and the creator of the digital data [(1)–(3)]. To solve this problem, a technique called "trusted timestamping" based on a central timestamping authority (TSA) has been discussed [(4)]. Trusted timestamping is a process that could securely track the creation and modification times of digital data [(5)–(7)]; in addition, it guaranties the existence and integrity of the data [(6)]. Users send a digital file to the TSA, where it is digitally signed with the current time. However, trusted timestamping entirely relies on TSA, a central third party, which may lead to safety problems. For example, if TSA's server is hacked, the service may have to be stopped for some time. In addition, the timestamp can be tampered with by hackers. To prevent such problems and increase reliability, multiple TSAs could be a solution [(5)]. However, multiple TSAs may lead to high costs. Another method, called decentralized trusted timestamping, has been implemented and is based on the opposite concept of centralization.

At present, there are several decentralized timestamping services called "blockchains" based on Bitcoin's peer-to-peer digital currency infrastructure. Essentially, a blockchain is a distributed database that does not rely on central servers, i.e., it is decentralized. Blockchain stores all confirmed Bitcoin transactions [(8)]. The existence and integrity of transactions are protected, i.e., the transactions cannot be tampered with or forged. Based on such features, web services, such as BTProof [(9)] and Proof of Existence (PoE) [(10)], have implemented decentralized timestamping. However, the details of their implementations are not all the same and there is a significant problem with such applications: only 40 bytes of data can be stored in a transaction in a blockchain.

Existing web-based trusted timestamp services embed a maximum 40-byte hash in the blockchain. The hash data cannot be reversed, which makes determining the creators and other related information difficult. To store such information, we propose a methodology for storing a maximum of N*20 bytes in the blockchain. For creators who prefer not to anonymously timestamp their creations, the proposed method can embed related information (e.g., the file name, the creator's name, and comments) and the hash of the digital data. Note that the related information can be decrypted in plain text. Our proposed method creates transactions with N output addresses; therefore, one transaction can store at most N*20 bytes. Keeping in mind the bloating problem of blockchains, we set N = 3 and conducted an experiment to evaluate the proposed method and verify the existence and integrity of a digital file. In addition, we evaluated the cost and computation time (the broadcast time) of the proposed method. The experimental results show that the proposed method can prove the existence of digital data at a particular time and that the digital data has not been modified since that time (integrity). We also find that the proposed method implements decentralized trusted timestamping in an average time of 24 min at a possible cost of 0.24 USD.

The remainder of this paper is organized as follows. In Section 2, we provide an overview of Bitcoin and blockchain, and describe current decentralized trusted timestamping services. In Section 3, the proposed method is explained in

a) Correspondence to: Yuefei Gao. E-mail: kou@cmu.iit.tsukuba.ac.jp
[*] Department of Intelligent Interaction Technologies, University of Tsukuba
1-1-1, Tennodai, Tsukuba, Ibaraki 305-8573, Japan

detail. Experimental results are presented and discussed in Section 4, and the conclusions are presented in Section 5.

2. Related Work

2.1 Overview of Bitcoin and Blockchain Bitcoin is a peer-to-peer crypto digital currency system proposed in 2009 by an unknown person or entity using the pseudonym Satoshi Nakamoto [8]. Bitcoin is decentralized and does not rely on any government or other legal entitiy [11]. This means that users can make transactions directly and securely anywhere via the Internet.

Blockchain is Bitcoin's public ledger and has three important features.

1) The blockchain is a big record book that contains a large number of entries. In actuality, all of Bitcoin's transactions are recorded in the blockchain. Even though nearly all the entries describe Bitcoin transactions, it is actually possible to record anything in the blockchain.

2) The blockchain is shared between all Bitcoin users [12]. This means that the keeper of the blockchain is not just one person but rather multiple Bitcoin users. Further, the agreement of numerous record keepers is necessary whenever something is written into the blockchain. Therefore, no one user controls of the blockchain.

3) It is extremely difficult to tamper with or forge records once they are stored in the blockchain. The blockchain has multiple keepers and are public to everyone, including those who are not Bitcoin users. This means that the records stored in the blockchain are secure [13].

The existence and integrity of data in the blockchain is guaranteed by the above features. Existence means that the data have actually existed since a certain time. Integrity means that the data have not been altered after that certain time. Transactions are stored in chronological order in the blockchain. Figure 1 illustrates the features and data contained in the blockchain. The blockchain comprises nearly 400,000 blocks, and each block primarily contains 1) the previous block's hash, 2) a timestamp, 3) a nonce, and 4) hash strings for approximately 500 Bitcoin transactions.

Figure 2 shows the data contained in a Bitcoin transaction. Transaction data primarily include the version, input, output and locktime. The output field contains the number of transferred Bitcoins and the receivers' addresses.

Bitcoin has three primary types of transactions, i.e., common, aggregating, and distributing.

1) Common transactions are simple payments that have one input and two outputs (output 0 is to the receiver and output 1 sends the change back to the sender).

2) Aggregating transactions have multiple inputs and only one output.

3) Distributing transactions are transactions that distribute one input to multiple outputs [14].

In this study, we use distributing transactions.

2.2 Current Trusted Timestamping Services Trusted timestamps are issued by a central TSA. A user sends a digital file to the TSA, where it is digitally signed with the current time. The protocols of this technology were specified in RFC 3161 [15]. Before we explain how trusted timestamps are issued, we discuss the concept of a hash. Hash functions are cryptographically secure functions [16]. For digital data,

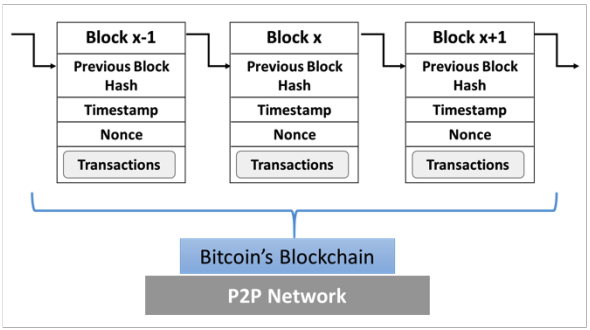

Fig. 1. The blockchain

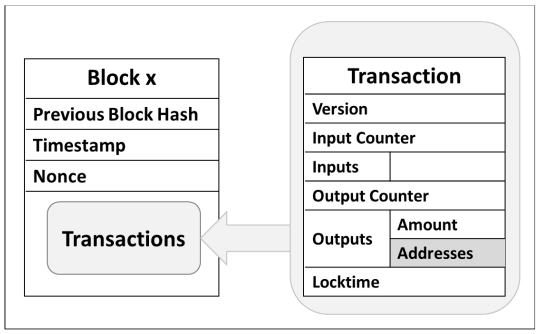

Fig. 2. Example Bitcoin transaction in the blockchain

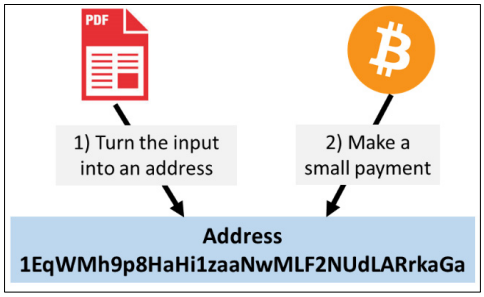

Fig. 3. The BTProof processes

the hash value is first calculated using a hash function. Hash functions are collision-resistant one-way functions [17], which means that obtaining the original digital data from the hash values is impossible. Using hash functions increases security when sending data to the TSA. Note that the SHA-256 and RIPEMD-160 hash functions are used in Bitcoin's protocol. A 256-bit hash is calculated using the SHA-256 hash function, and a 160-bit hash is calculated using RIPEMD-160 when a shorter hash value is required.

There are several web applications that have been implemented using different methods for decentralized timestamps based on the blockchain, e.g., BTProof and PoE. These applications help prove that a digital file existed at a point in time (existence) and that it has not being altered since that given time (integrity). Note that BTProof and PoE require a digital file as input.

In BTProof's trusted timestamping, the input file is first hashed and converted into a Bitcoin address. As shown in Fig. 3, an address is a string of characters and numbers. Then, by making a small payment to the address, the transaction and therefore the hash are stored in the blockchain. Transactions in the blockchain are extremely difficult to tamper with or

Fig. 4. The P2PK H and OP_RETURN scripts

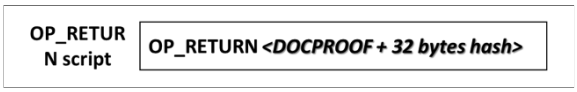

Fig. 5. PoE's OP_RETURN scripts

forge; therefore, the digital file is stored securely [9]. However, the hashed digital content cannot be reversed, which means that determining if any special data were stored in the transaction is impossible. If additional information can be stored in part of a transaction and converted to plain text, such information can be identified and searched.

PoE implements Bitcoin's script language to embed digital files in the blockchain. Bitcoin transactions are validated by executing a script written in a Forth-like scripting language. At present, most Bitcoin transactions have the form "A pays B". Such transactions are based on scripts referred to as Pay-to-Public-Key-Hash (P2PKH) scripts. However, Bitcoin transactions are not limited to the "A pays B" form. There are five standard types of transaction scripts: P2PKH, public-key, multi-signature (limited to 15 keys), pay-to-script-hash (P2SH), and data output (OP_RETURN). PoE applies two of these types, i.e., P2PKH and OP_RETURN [8]. Figure 4 shows these two types of script.

In PoE, a document is also hashed; however, it is hashed to a 32-byte string rather than a Bitcoin address. Then, the 32-byte string is embedded in a data field of the transaction. Therefore, a special transaction is constructed. After broadcasting this special transaction, the 32-byte hash is stored in the blockchain with the special transaction, making it difficult to tamper with or forge, PoE also uses "DOCPROOF" as a marker for their transactions by placing it at the beginning of the 32-byte string, which makes it easier to search for in transactions [10]. Figure 5 shows the format of the embedded information created by PoE.

2.3 Problems with Current Trusted Timestamping Services and the Proposed Solutions Both of the above two services provide trusted timestamping based on Bitcoin's the blockchain and can help verify the integrity and existence of digital data. However, knowing what type of data are in the transactions is impossible because only the hash value of the digital data is embedded. Present trusted timestamping methods only include one field for less than 40 bytes of included data. Therefore, storing related data is difficult. To enable including additional data, we propose a new type of transaction with N data fields for at most N*20 bytes of stored data. Using the proposed method, not only the hash value of the digital data but also the creator's name and additional related data can be included. We chose an origami diagram as an example and applied the proposed method to them. The goal was to add trusted timestamping to the origami diagrams including creator and related information.

3. Proposed Method

Even though similar web services (e.g., BTProof and PoE) have been implemented, they are limited by the size of the data recorded in a transaction. The primary objective of this study is to implement decentralized trusted timestamping to help protect the existence and integrity of digital data using the blockchain by expanding the storage space from 40 bytes to N*20 bytes. The proposed method can be used to protect a large variety of digital data such as documents, music and videos. In this study, we set N to 3 and applied the proposed method to a picture.

3.1 The Proposed Method In Section 2.2, the current decentralized trusted timestamping services were introduced. Their methods are defined as follows.

Define $\boldsymbol{W}$ to be a set containing all the alphanumeric characters. $\boldsymbol{W^x}$ is a set containing all the alphanumeric characters with a length of $\boldsymbol{x}$, and $\boldsymbol{W^*}$ represents $\boldsymbol{W}$ without 0 (zero), O (capital o), I (capital i), and l (lower case L). We use $\boldsymbol{U_{16}{}^y}$ to represent hexadecimal numbers with a length of $2\boldsymbol{y}$. A digital file is represented by $\boldsymbol{F}$.

Next, we define functions based on the definitions above.

The hash functions SHA-256 and RIPEMD-160 are defined as

$$\boldsymbol{SHA\text{-}256: W\ or\ F \rightarrow U_{16}{}^{32}}, \quad (1)$$

$$\boldsymbol{RIPEMD\text{-}160: W\ or\ F \rightarrow U_{16}{}^{20}}. \quad (2)$$

The hex function converts alphanumeric characters into hexadecimal numbers. Because only the hexadecimal numbers with the length of 20 bytes can be converted to Bitcoin addresses, if a < 20, zeros are added to make it 20 bytes, that is,

$$\boldsymbol{Hex: W^{20} \rightarrow U_{16}{}^a \quad (\rightarrow U_{16}{}^{20})}. \quad (3)$$

Base58 is the encoding scheme for Bitcoin addresses [18].

$$\boldsymbol{Base58: U_{16}{}^{20} \rightarrow W^*}. \quad (4)$$

The proposed method has N inputs. We define a set $\boldsymbol{I}$ to represent the inputs:

$$\boldsymbol{I} = \{\boldsymbol{W_1, W_2, W_3, \ldots W_{N-1}, F_N}\}, \quad (5)$$

$$\boldsymbol{I'(n)} = \begin{cases} \boldsymbol{Hex}(\boldsymbol{W_n}), & \boldsymbol{1 \le n \le N-1} \\ \boldsymbol{RIPEMD\text{-}160}(\boldsymbol{F_n}), & \boldsymbol{n = N} \end{cases} \in \boldsymbol{U_{16}{}^{20}}, \quad (6)$$

$$\boldsymbol{Bitcoin\ Address}(n) = \boldsymbol{Base58}(\boldsymbol{I'}(n)), \quad 1 \le n \le N. \quad (7)$$

The above two formulas, Eqs. (6) and (7), show how the proposed method converts N inputs into N Bitcoin addresses.

We can also represent the current services BTProof and PoE in a simple way. BTProof uses

$$\boldsymbol{Base58}(\boldsymbol{RIPEMD\text{-}160}(\boldsymbol{SHA\text{-}256}(\boldsymbol{W^x} \text{ or } \boldsymbol{F}))) \quad (8)$$

to convert the information users input or the files submitted into Bitcoin addresses. PoE hashes the files from users and adds its website marker at the beginning such that

$$\boldsymbol{W^8} + \boldsymbol{SHA\text{-}256}(\boldsymbol{F}). \quad (9)$$

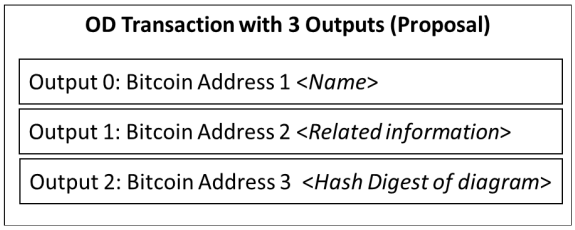

Fig. 6. Outputs of the proposed Origami Transaction

3.2 Application to Digital Data

As introduced in Section 1, a Bitcoin transaction has one or more inputs and outputs. In common transactions, output addresses are often bank accounts for transfers. However, in the proposed method, a transfer is not the goal. The goal is to implement trusted timestamping and apply it to digital data. Therefore, the output addresses have a special meaning. In the proposed method, the output addresses are special because the additional information (e.g., file names, creator names, and keywords) are embedded in them.

Normal Bitcoin addresses are generated randomly without any special information. In the proposed method, each address contains special data. First, the special data are turned into a hexadecimal number. Zeros are then added to make the hexadecimal number 20 bytes. Then, the 20-byte hexadecimal number is turned into a Bitcoin address with the encoding scheme Base58. At last, a transaction is created and the addresses are transferred. The time when the transaction is recorded in the blockchain is the trusted timestamping.

In the proposed method, the N addresses contain N*20 bytes of data. However, if a transaction has too many outputs, it may have problems in broadcasting and too many such transactions will cause bloating problems for the blockchain. To avoid such problems, we set N = 3 and propose a three-output transaction. We applied the proposed method to a picture, i.e., an origami diagram. We call this an "Origami Diagram Transaction" (abbr. OD Transaction, Fig. 6). An OD Transaction contains three outputs (addresses). The first address contains the name of the diagram, the second address contains the origami diagram's related information, and the third address contains the hash value of the origami diagram. Figure 7 shows an example of the proposed OD Transaction and the decoded result. Outputs 0 and 1 can be decoded in plain text, while Output 2 is messy code. This is because the hash digest in Output 2 is irreversible. Therefore, the original diagram is safe.

(Advantage 1) The file name and the creator can be derived directly from the decoded result. Figure 7 shows an example of the result after decoding an OD Transaction. In Outputs 0 and 1, the origami diagram's name, the creator's name or the use permission are in plain text. The hash of the diagram is embedded in Output 2. To minimize the OD Transaction's size, we chose the RIPEMD-160 hash algorithm to produce the 20-byte hash value. Accordingly, an OD Transaction is approximately 260 bytes.

(Advantage 2) The web application based on the proposed method is easy to use. Simply input the related data and click on a button and the diagram will be stored in the blockchain forever, without being tampered with or forged.

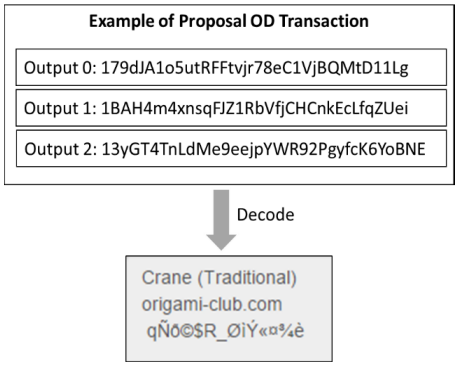

Fig. 7. Example of the proposed OD Transaction and the decoded result

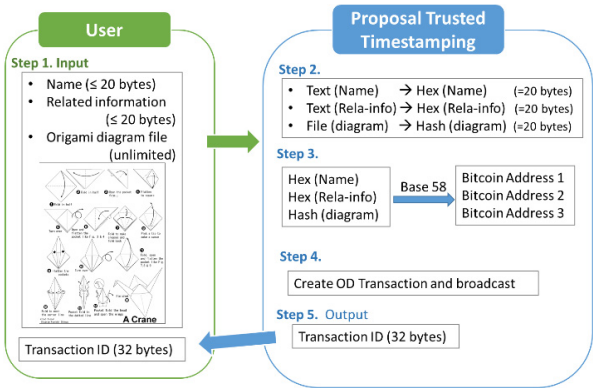

Fig. 8. Steps of the proposed method in the case of an origami diagram

4. Evaluation of the Proposed Method

To evaluate the proposed method in terms of time and cost, we conducted two experiments and investigated the possible lower bound of the cost. Figure 8 shows the steps of the proposed method. Suppose a proposed OD transaction is broadcast at time A and included in the blockchain at time B. We record the time period between times A and B. The cost is the fee that is required to record a transaction in the blockchain.

Step 1 shows what the user needs to input to get a trusted timestamp. Steps 2–5 shows how the proposed method works. The steps will be explained in detail.

Step 1: The user fills in three inputs: name, related information and the origami diagram file that will be timestamped. The name may contain both the diagram's name and the creator's name. Related information may include use permissions, a link, or other information about the diagram. The chosen origami diagram can be a JPED, PDF, or other format. The crane diagram in Fig. 8 is a 42KB GIF file. After clicking the submit button, the user receives a 32-byte transaction ID.

Step 2: Name and related information are turn into two 20-byte hexadecimal strings and a 20-byte hash value for the origami diagram file is calculated.
Hex of Name:
4372616e65202854726164974696f6e616c2900
Hex of Related information:
6f726967616d692d636c75622e636f6d00000000

Hash of the diagram file:
2094711d9bd1f0a924525fd81fecddaba4bee89d

Step 3: The three 20-byte strings are turned into three special Bitcoin addresses.
Address 1:
179dJA1o5utRFFtvjr78eC1VjBQMtD11Lg
Address 2:
1BAH4m4xnsqFJZ1RbVfjCHCnkEcLfqZUei
Address 3:
13yGT4TnLdMe9eejpYWR92PgyfcK6YoBNE

Step 4: An OD Transaction is created using the three special addresses (Fig. 8) and the transaction is broadcast to the Bitcoin network.

Step 5: The transaction ID is sent to the user. The transaction ID can be used to search and verify the special transaction that contains the origami diagram in the blockchain. Figure 9 is one of transactions recorded in the blockchain.

In Fig. 9, 1) the addresses with related information, 2) the timestamp (2015-09-18 05:51:29), and 3) the fee (0.00001 BTC) are shown in the search result. If we decode the three addresses, the result will be Fig. 7. Then, the name of the origami diagram and the website can be known.

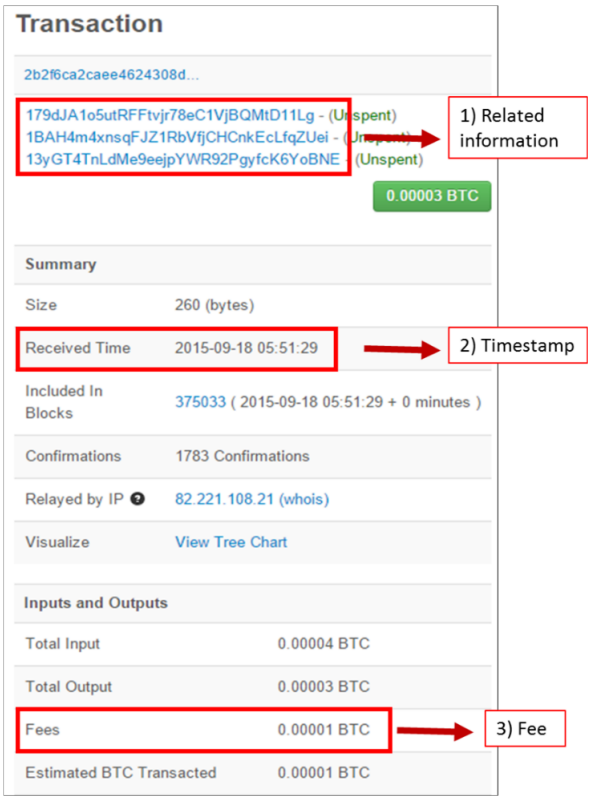

Fig. 9. An OD Transaction recorded in the blockchain

According to the steps explained above, two experiments were conducted. The first experiment was to determine the lower bound cost of sending a proposed OD transaction. The second experiment was to calculate the average time to complete such a transaction. The experimental environment was implemented on a computer with OS Ubuntu 14.04, the platform Node.js, and Bitcoin's client Bitcoin Core v0.9.3.0.

4.1 Investigation of the Minimum Fee for the Proposed Method The goal of the first experiment was to determine the lower bound on the possible transaction fee. Even though according to the protocol, the smallest fee could be 0, such transactions may not be accepted. As shown in Table 1, the six fees ranged from 0.000005 BTC to 0.001 BTC. We see that five values are succeeded and the last one failed. Therefore, the possible lower bound of the fee is approximately 0.00001 BTC.

4.2 Comparison between Costs and Time In Experiment 2, we measured the time for five different costs. For each cost, the time was measured 10 times, and 50 experiments were conducted in total. Table 2 shows the results. The average time is approximately 24 min. This means that the trusted timestamping to the origami diagram can be finished in approximately 24 min.

From the two experiments, we know that the proposed method can get a trusted timestamp at approximately 24 min with a possible cost of 0.0005 BTC.

5. Conclusions

In this study, we implemented trusted timestamping based on Bitcoin's infrastructure to store additional related information concerning a piece of digital data. Instead of inserting the data into the OP_RETURN field, we proposed to encode and turn the related data into N addresses. N was set to three and the proposed method was applied to origami diagrams. By performing two experiments, the proposed method was evaluated with respect to time and cost. After

Table 1. Result of Experiment 1

Fee (10^{-4} BTC)	10	5	1	0.5	0.1	0.005
Result	Succeed					Failed

(1 BTC ≈ 447.71 USD, 2016/5/26)

Table 2. The results of Experiment 2

Cost (BTC)	Time (min)										
	Group1	Group2	Group3	Group4	Group5	Group6	Group7	Group8	Group9	Group10	Average
0.001	15	80	3	32	58	8	30	1	4	11	**24.2**
0.0005	15	80	3	32	58	8	30	4	4	11	**24.5**
0.0001	1407	1342	2671	1002	860	796	1000	1242	1247	1015	**1258.2**
0.00005	1407	1342	2672	1002	861	796	1000	1242	255	1015	**1159.2**
0.00001	1407	1342	3657	1002	2363	796	1000	1246	1247	1015	**1507.5**

being broadcast, the proposed transactions were recorded into the blockchian in an average of approximately 24 min with a minimum cost of 0.0005 BTC. The proposed method has benefits for both authors and users. Authors who create digital files can protect the existence and integrity of their creation by applying the proposed method. This provides them with evidence to certify their creations. As for the users, they can discover what is stored in a certain transaction. When they want to use these files, they can find additional information such as permissions and restrictions in the copyright information. In future, we will improve and apply the proposed method to additional intellectual properties to protect copyrights and patents.

References

(1) P. Hartel, L. Abelmann, and M. Khatib: "Towards Tamper-evident Storage onPatterned Media", Consulted 2015/02/25. http://doc.utwente.nl/64661/1/secstor.pdf

(2) C. Lu and H. Liao: "Structural Digital Signature for Image Authentication", Consulted 2015/02/25. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.31.8070&rep=rep1&type=pdf

(3) N. Pandey, S. Nandy, and S. Choudhury: "Alternative Shift Algorithm for Digital Watermarking on Text", International Journal of Scientific and Research Publications, Vol.2, No.10. Consulted 2015/02/25. http://www.ijsrp.org/research-paper-1012/ijsrp-p1058.pdf (2012)

(4) A. Bonnecaze, P. Liardet, A. Gabillon, and K. Blibech: "Improving Time Stamping Schemes: A Distributed Point of View", Consulted 2015/02/25. http://pages.upf.pf/Alban.Gabillon/articles/AnnalTelecom.pdf

(5) CRYPTOMAThIC: "Cryptomathic Time Stamping Authority", Consulted 2015/02/25. http://www.cryptomathic.com/media/12284/cryptomathic%20tsa-product%20sheet.pdf

(6) CSS Technical Team: "Time Stamping Authority", Consulted 2015/02/25. http://www.css-security.com/blog/time-stamping-authority/

(7) B. Gipp, N. Meuschke, and A. Germandt: "Decentralized Trusted Timestamping using the Crypto Currency Bitcoin", Consulted 2015/02/25. http://www.gipp.com/wp-content/papercite-data/pdf/gipp15a.pdf

(8) S. Nakamoto: "Bitcoin: A Peer-to-Peer Electronic Cash System", Consulted 2015/02/03. https://bitcoin.org/bitcoin.pdf (2008)

(9) BTProof. Consulted 2015/02/03. https://www.btproof.com/

(10) Proof of Existence. Consulted 2015/02/03. http://www.proofofexistence.com/about

(11) R. Grinberg: "Bitcoin: An Innovative Alternative Digital Currency", Consulted 2015/02/03. http://heinonline.org/HOL/LandingPage?handle=hein.journals/hascietlj4&div=6&id=&page=

(12) Bitcoin.org: "The blockchain", Consulted 2015/02/03. https://bitcoin.org/en/vocabulary#the blockchain

(13) Blocksign. Consulted 2015/02/25. https://blocksign.com/about/theblockchain

(14) M.A. Andreas: "Mastering Bitcoin", O'Reilly Media, Inc. (2014)

(15) C. Adams, P. Cain, D. Pinkas, and R. Zuccherato: "Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP)", Consulted 2015/02/25. https://www.ietf.org/rfc/rfc3161.txt (2001)

(16) S. Haver and W. Stornetta: "How to Time-Stamp a Digital Document", Consulted 2015/03/18. https://www.anf.es/pdf/Haber_Stornetta.pdf

(17) NIST: "Descriptions of SHA-256, SHA-384, and SHA-512", Consulted 2015/03/18. http://csrc.nist.gov/groups/STM/cavp/documents/shs/sha256-384-512.pdf

(18) "Base58", Consulted 2015/10/06. https://en.wikipedia.org/wiki/Base58

Yuefei Gao (Non-member) received the M.E. degree from the University of Tsukuba, Ibaraki, Japan in 2016. She is currently working towards the Ph.D. degree in the Department of Intelligent Interaction Technologies, University of Tsukuba, Ibaraki, Japan. Her current work concerns the blockchain technology of Bitcoin.

Hajime Nobuhara (Member) received the Ph.D. degree from Tokyo Institute of Technology, Japan in 2002. He worked as a post doctoral fellow in University of Alberta, Canada from April to September in 2002. From 2002 to 2006, he has been affiliated with Tokyo Institute of Technology, Japan. From 2006 until now, he has been affiliated with University of Tsukuba, Japan. IEEE, Japan Society for Fuzzy Theory and Intelligent Informatics, the Institute of Electronics, Information and Communication Engineers member.