International Journal of Advanced Culture Technology Vol.9 No.4 392-401 (2021)
DOI https://doi.org/10.17703/IJACT.2021.9.4.392

IJACT 21-12-46

A study for system design that guarantees the integrity of computer files based on blockchain and checksum

Minyoung Kim

Assistant professor, Research Institute of ICT Fusion and Convergence, Dong-Eui Univ., Korea
kmyco@deu.ac.kr

Abstract

When a data file is shared through various methods on the Internet, the data file may be damaged in various cases. To prevent this, some websites provide the checksum value of the download target file in text data type. The checksum value provided in this way is then compared with the checksum value of the downloaded file and the published checksum value. If they are the same, the file is regarded as the same. However, the checksum value provided in text form is easily tampered with by an attacker. Because of this, if the correct checksum cannot be verified, the reliability and integrity of the data file cannot be ensured. In this paper, a checksum value is generated to ensure the integrity and reliability of a data file, and this value and related file information are stored in the blockchain. After that, we will introduce the research contents for designing and implementing a system that provides a function to share the checksum value stored in the block chain and compare it with other people's files.

Keywords: *File Integrity, Checksum, Hash, Blockchain, JavaScript*

1. INTRODUCTION

A data file is created by computer software with information tailored to the purpose of the user who created it. The data files created in this way are shared with other users by the user using various functions of the Internet. For example, a user uploads a document data file created by himself to a bulletin board of an Internet website and others download it, or transmits the file to others as an e-mail attachment. The data generated in this way is further used as a raw material for big data analysis and A.I learning. Therefore, it is important to protect the integrity of the data to secure the reliability of the information contained in the data. Otherwise, it may produce undesirable results for users in big data analysis and A.I learning.

Data files circulating on the Internet are likely to change the data due to various factors. Typically, hackers use steganography [1] to add malicious code that can attack the user's computer to the data of the file and distribute it over the Internet. To prevent this, reputable websites provide checksum values of downloadable files (Figure 1. (a)). Users first download the file and then obtain the checksum value of the downloaded file through checksum verification software (Figure 1. (b)). Finally, this checksum value is compared with the checksum value provided by the homepage, and if it is the same, it is confirmed that the file is the same as the file provided by the homepage. If a hacker attacks a web page that provides checksum value data and changes

Manuscript received: November 30, 2021 / revised: December 2, 2021 / accepted: December 7, 2021
Corresponding Author: kmyco@deu.ac.kr
Tel: +82-51-890-4267
Assistant professor, Research Institute of ICT Fusion and Convergence, Dong-Eui University, Korea

the content, it causes a problem in that the integrity and reliability of the file are not secured.

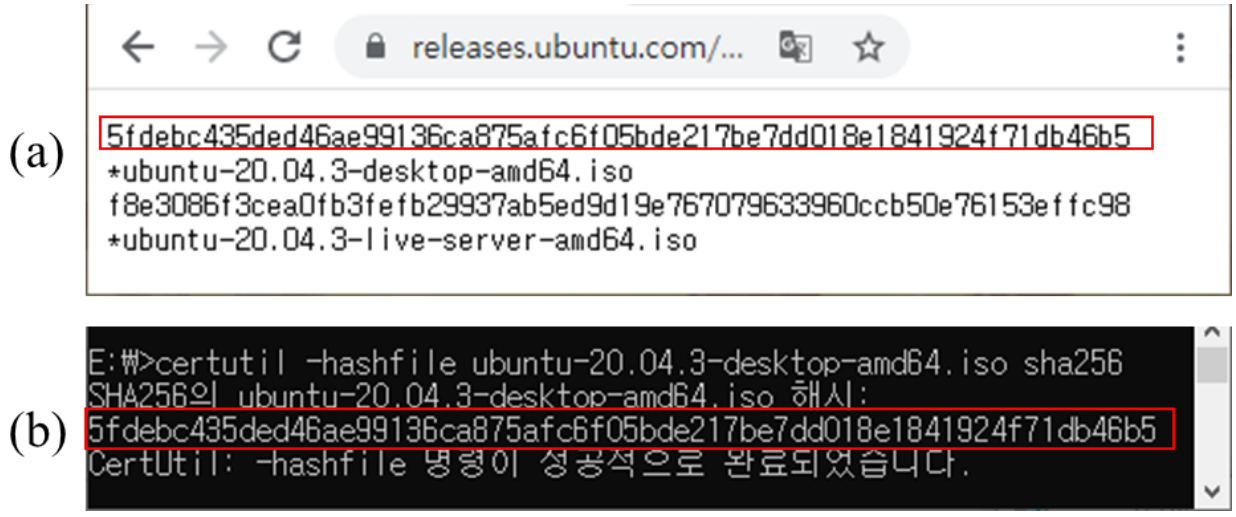


Figure 1. (a) Checksum of iso file provided from ubuntu official website [2], (b) Checked the checksum of iso file downloaded from ubuntu official site

The checksum data value can be used a specific hash algorithm to generate the data file's unique checksum value data. Checksums are utilized in many fields to ensure the integrity of critical files by taking advantage of these characteristics. However, the generated checksum value data is kept in plain text, and the same problem mentioned in the above sentence occurs. If there is a system in which the checksum value of the data file is stored in a secure storage and the checksum value data stored there is completely guaranteed, the integrity of the data file can be maintained from the problems mentioned above. This system is calculated a checksum value from the target file, compares that value with the value stored in this system, and proves the integrity of the target file if the checksum values of the two files are the same. If this can be done, the integrity of the data files distributed on the Internet is verified, the reliability of the information contained in the data is secured, and various problems caused by data corruption mentioned in the above section are solved.

In this paper, the checksum value of the target file is obtained from the website and the checksum value is stored in the block chain. After that, it deals with the research contents for designing and implementing a system that can obtain the checksum value of another target file and compare the two values to ensure the integrity of the corresponding file. So, in this paper, I introduce the contents for designing a system to solve the problems mentioned in the above sentence. First, in this paper, the ICT technology used in the system will be introduced and similar related academic research will be explained. Then, the main components of the system and its definitions will be presented. In addition, the contents of the study on how this system can be implemented using what kind of technology is also presented. Finally, we discuss what the system presented in this paper suggests to us and introduce future research plans for it.

2. RELATED RESEARCH

A hash encryption algorithm is used to extract the data file checksum value. This is because the probability that hash values collide is close to 0%. This is because hash values from the same hash algorithm do not overlap. Currently used typical algorithms are MD5 (Message-Digest algorithm 5) [3] and SHA-256 (Secure Hash Algorithm 256 bytes) [4]. The reason is that it has a short hash value, so the comparison time can be shortened compared to other hash algorithm values. In the case of MD5, the result value has a length of 128 bytes, and

in the case of SHA-256, it has a length of 64 bytes which is shorter than that of MD5 [5].

Techniques for verifying the integrity of data files on the Internet using checksum values are still being studied. Typically, MEYLAN [6] checks the integrity of data files by automatically performing checksum verification when downloading data files from the Internet. Unfortunately, this study presented a good technique to use in a single system. In this paper, the checksum value of the file registered in the system presented in this paper is shared so that it can be used by other sites as well. This is because the checksum value for which the user will objectively compare the checksum data of the data file downloaded from the website is stored in the blockchain, thereby guaranteeing that the two files are identical. The reason is that, due to the nature of the block chain, data stored in the block chain once cannot be changed. Unfortunately, there is currently no system provided by the method presented in the introduction of this paper.

The checksum value of the data file to be stored in the system presented in this paper is stored in the blockchain. This is to ensure reliability by protecting the integrity of data because the checksum value stored in the blockchain of this system is not changed when data files are collated. Blockchain stores a single block by tying multiple transaction data together. In addition, the hash value of the previous block is stored. In this way, if the contents of the previous block are modified later, the hash value of the next block does not match, so the reliability of the block is forgotten. This ensures the reliability of the data because the data stored in the blockchain is not modified. On the other hand, if an existing database is used, it is easy for hackers to modify data as long as they have administrator rights.

3. SYSTEM CONFIGURATION

Figure 2. System configuration proposed in this paper

The overall configuration of the system presented in this paper is Figure 2. The user can generate the checksum value of the selected file through the web service of the system and record the contents in the blockchain of the system. The blockchain of this system is responsible for the role of a database that stores all checksum data and can be inquired afterwards.

3.1 Web service

The web service of this system (hereinafter referred to as 'web service') is in charge of the interface through which users can interact with this system. This web service consists of a web server and a database as shown in Figure 3. Among web services, the web server plays an important role in this system and consists of functions as shown in Fig. 4.

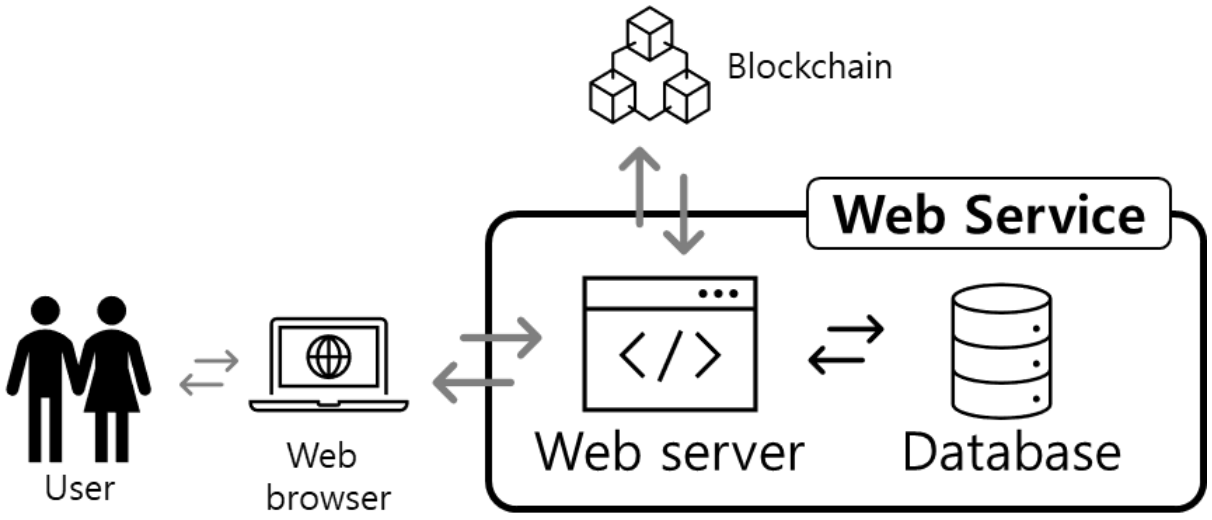


Figure 3. Web service internal configuration

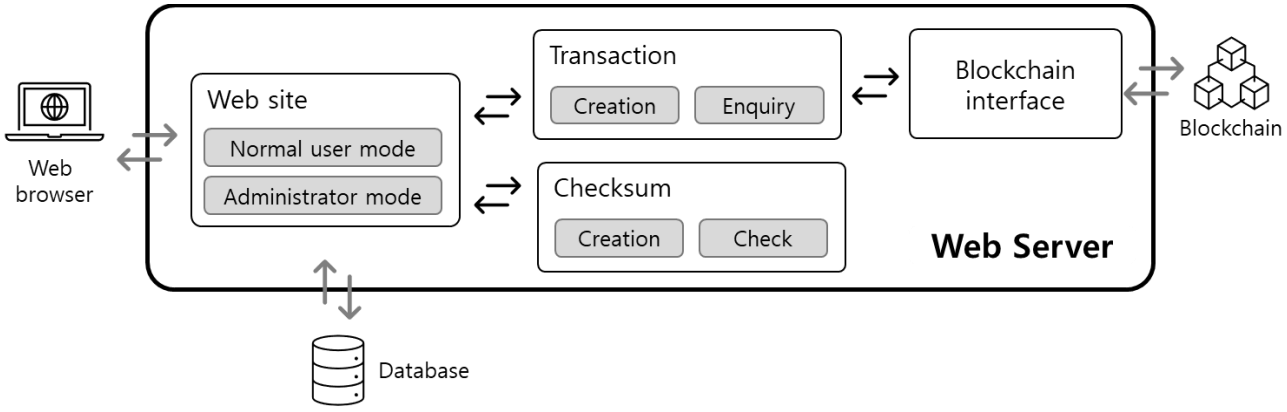


Figure 4. Web service function diagram

First, the website function is provided in general user mode and administrator mode. In the general user mode, the user accesses the system through a web browser, and provides a function to generate and save the checksum value of the selected file. At this time, the checksum value uses the ‘Checksum’ function of this web server. In addition, the user provides a function to inquire the checksum file stored in the system and to compare it with the selected file. User mode does not require a separate membership registration for convenience. The administrator mode provides the ability to manage the system as a whole. Typically, you can configure the website screen, database related, and block chain related settings. In order to use these functions, only the administrator user registered in this system is given the right to use these functions.

The second is the transaction function. The transaction creation function is responsible for generating one piece of data by collecting the checksum value, file information, and event occurrence information generated for the file selected by the user on the website. The information to be saved at this time is listed in Figure 5. It is then passed to the ‘blockchain interface’ function for storage on the blockchain. In addition, there is a transaction inquiry that can inquire and retrieve checksum values and various information of files stored in the block chain. At this time, the function uses the 'Blockchain interface'.

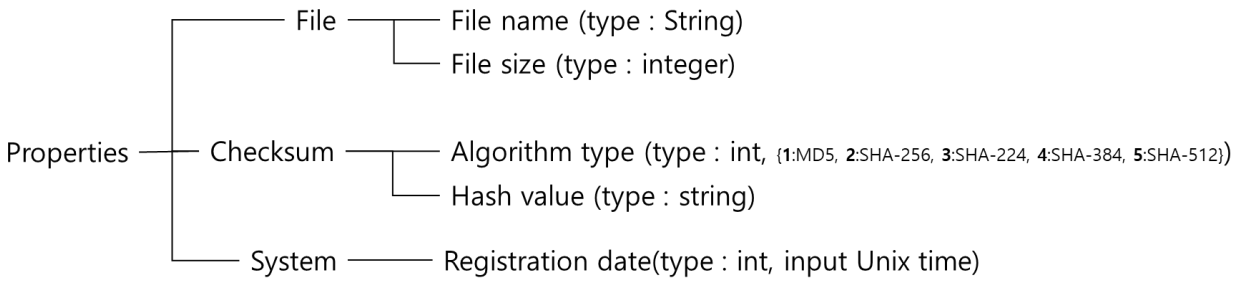


Figure 5. Structure of information to be stored in blockchain transaction

The third is the blockchain interface function. In the blockchain built in this system, it is responsible for storing the transaction created on the website in the blockchain and inquiring the stored data. Based on the data transfer protocol used in the built blockchain platform, this blockchain interface performs data transformation according to the situation when creating and inquiring transactions. Since most blockchain platforms transmit data in JSON format, a function to convert to JSON (JavaScript Object Notation) format must be implemented in this function.

Lastly, the checksum function is mainly responsible for generating and providing the checksum value of the file selected by the user on the website. The checksum generation function provides several cryptographic hash functions to obtain various checksum values, allowing the user to choose an algorithm. The checksum algorithm provided in this function basically provides the MD5[3] algorithm and all SHA family algorithms [4] suggested by NIST (National Institute of Standards and Technology). After that, algorithms suggested by NIST and relevant organizations in each country can be added. And the Check function extracts the checksum value from the given file and compares the checksum value to be compared and tells whether the two values are the same.

```
{
  "file": {
    "name": "5.jpg",
    "size": 138273
  },
  "Checksum": {
    "algorithm": 0,
    "value": "40a0e85faf3317975ad215b581c62ff4"
  },
  "System": {
    "create": 1638109931
  }
}
```

Figure 6. Example of transaction data designed in JSON

3.2 Blockchain

It is the block chain that acts as a database in the system of this paper. In the blockchain, data with checksum values and various information are stored and stored as a single transaction. It retrieves the stored data and provides the target transaction data. A distributed ledger that can store the transaction data mentioned in Section 3.1 is implemented. In this case, the blockchain uses an open-source platform. And configure the system according to the blockchain platform you use. If you build a blockchain with Hyperledger Fabric, you need at least 4 or more node server computers to share the distributed ledger with each other. In addition, a server is needed to collect transaction data sent by nodes connected in the same channel and distribute the organized transaction data to the nodes in the same channel. And in order to connect with the above-mentioned blockchain interface, the program to be loaded on the node must be implemented through Chaincode [7-10].

In this paper, we design the transaction data to be stored in the adopted blockchain platform. At this time, the contents mentioned in Section 3.1 are stored in the transaction and designed in JSON format as shown in Figure 6.

3.3 System usage scenarios

First, the user accesses the web service website in the system of this paper. After that, when a screen like Figure 7 (a) appears, select the file whose checksum value needs to be checked and click the OK button. At this point, the hash algorithm to be used for the checksum must be selected. After that, if the checksum value comes out successfully, as shown in Figure 07 (b), the checksum value and file information registered in the system, the identification number (transaction ID) that manages this data in the system, and the information at the time of upload are displayed together.

Afterwards, when the user sends the information of the file to another person through the attachment in the mail (Figure 7 (b)), the URL link provided can be sent, and the attachment file and the checksum registered in the system as shown later (Figure 8) Provides the ability to collate values.

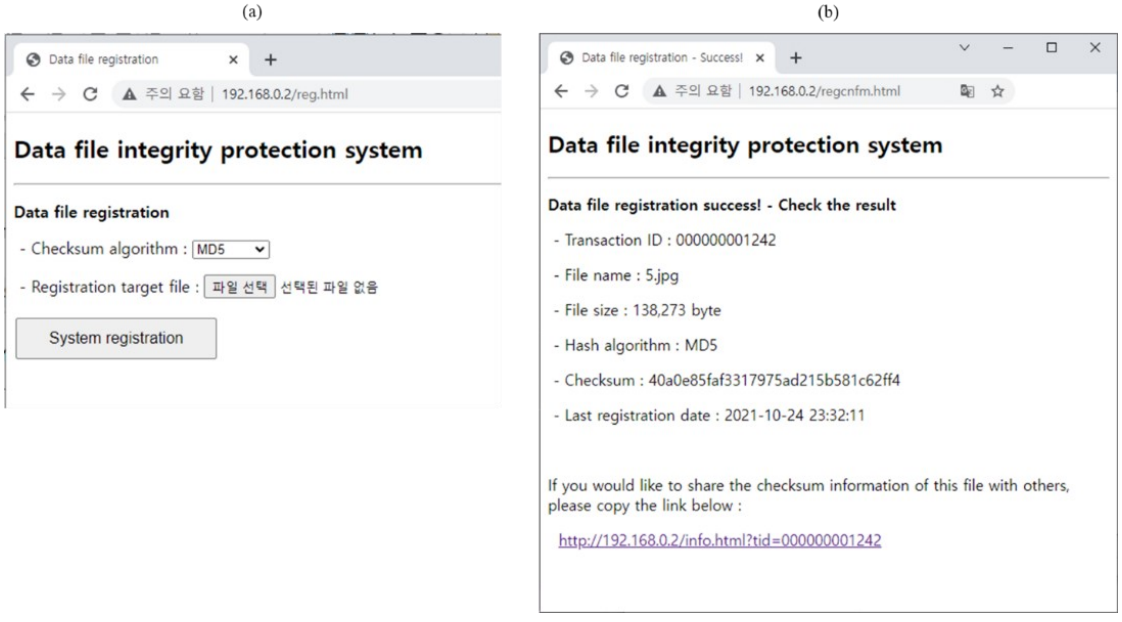


Figure 7. (a) The screen where the target file can be registered, (b) The screen where the target file is registered in this system

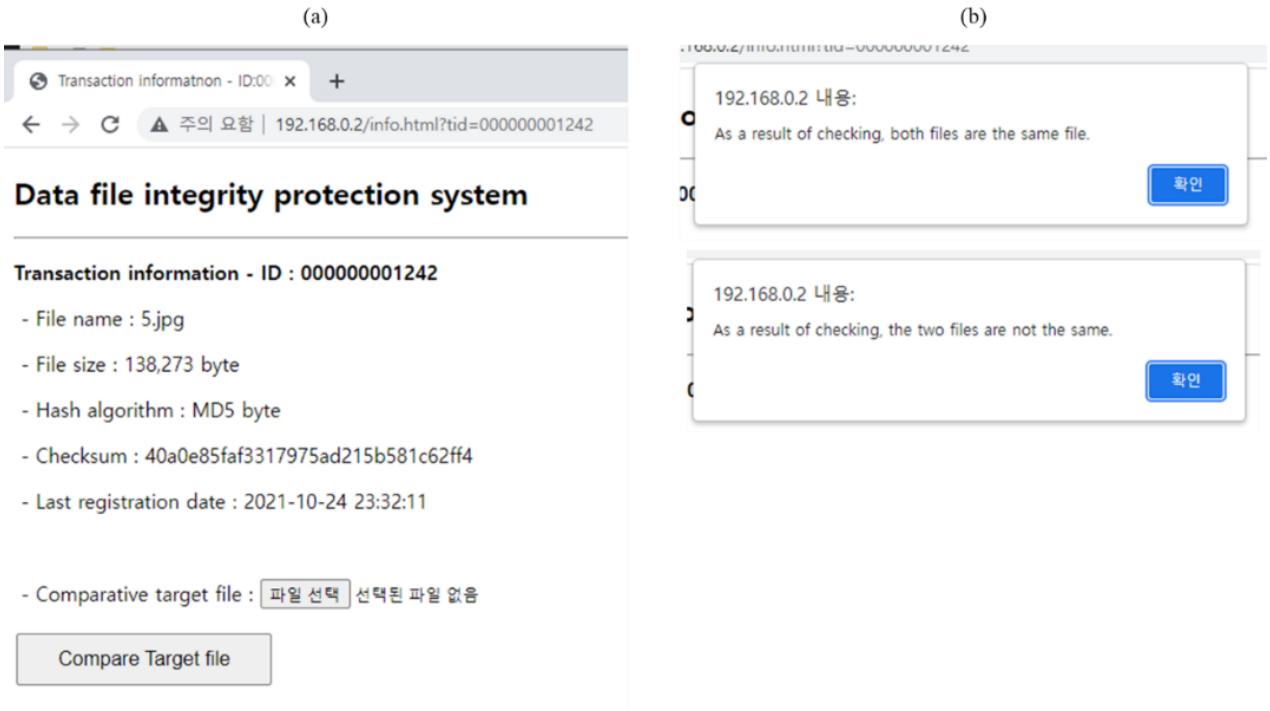


Figure 8. (a) Checksum information screen of the file searched by transaction ID, (b) Result message dialog through the checksum verification function provided on the page

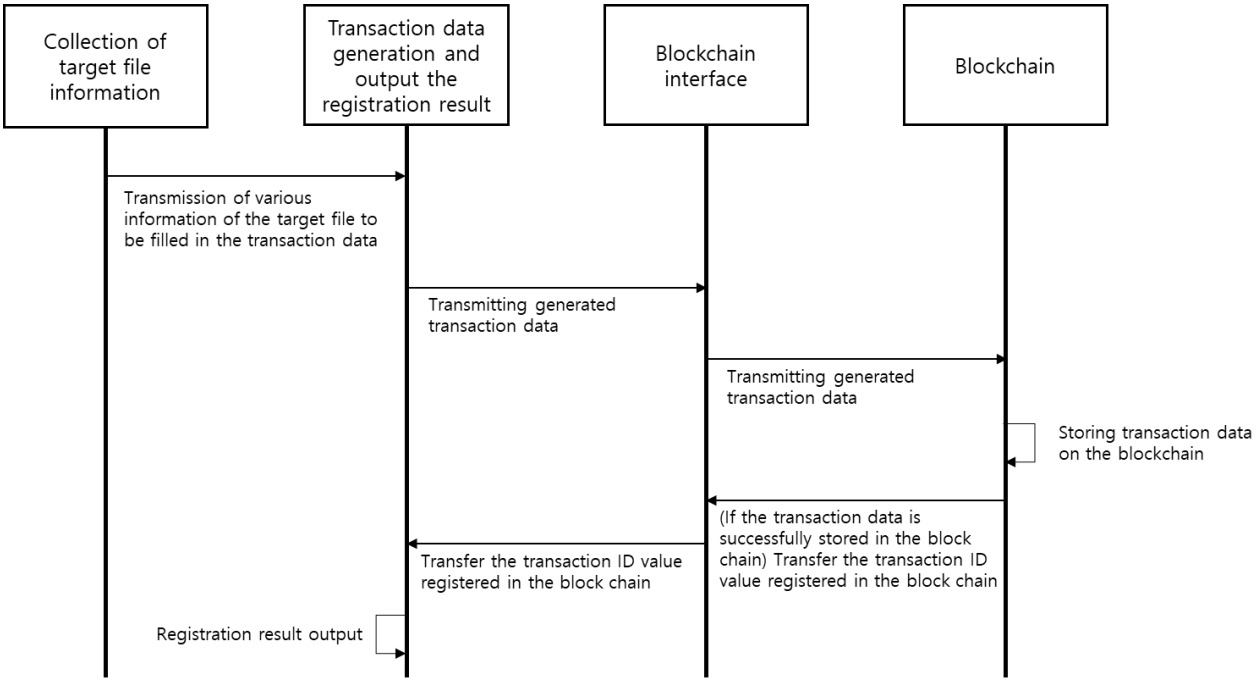


Figure 9. The website sequence diagram

4. HOW TO IMPLEMENT

This chapter presents the methodology for implementing the system design content presented in Chapter 3 of this paper. The first is the 'Web service' area. In the case of a website that is used directly by the user, the

website is implemented using HTML5, php, and JavaScript. At this time, in Figure 7(a), the target file information and other information to be stored in the transaction data to be stored and stored in the block chain are obtained. Then, in Figure 7(b), it is received various information sent from Figure 7(a), converts it into JSON format as shown in Figure 6, and sends it to the block chain interface function to send transaction data. After that, if the blockchain interface is registered this transaction data, it delivered the registered and secured transaction ID to Figure 7(b). The transferred transaction ID is output as shown in Figure 7(b). Fig. 9 is expressed as a sequence diagram of the contents mentioned here. After that, the transaction ID is also used to verify the transaction data in Figure 8.

In Figure 7(a), target file information (name and size (byte)), checksum value of target file, and registration time information are implemented in JavaScript. The part obtained by calculating the checksum value of the target file is implemented using the CryptoJS [11] library. To use the hash encryption function provided by CryptoJS, plaintext data (target file) must be converted into a *CryptoJS.lib.WordArray* object as shown in Figure 10. To convert to a *CryptoJS.lib.WordArray* object, the target file, which is plain text data, must be converted into a binary value.

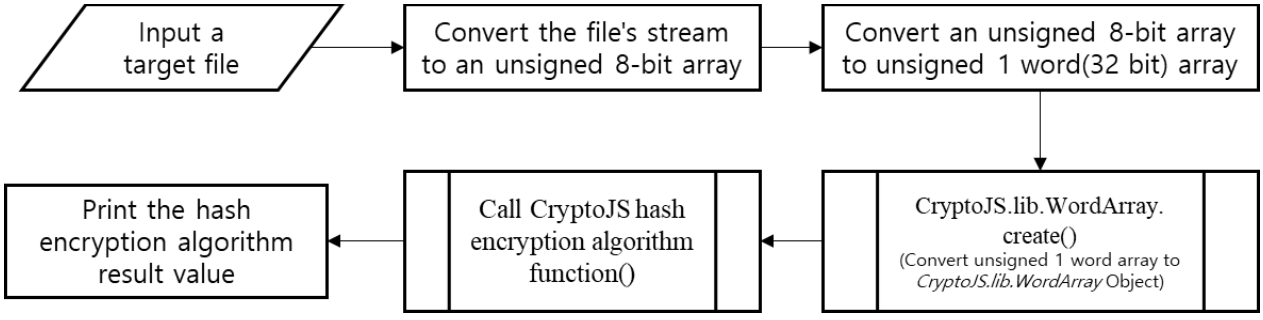


Figure 10. Flowchart of converting a plain data file stream to a CryptoJS.lib.WordArray object in JavaScript

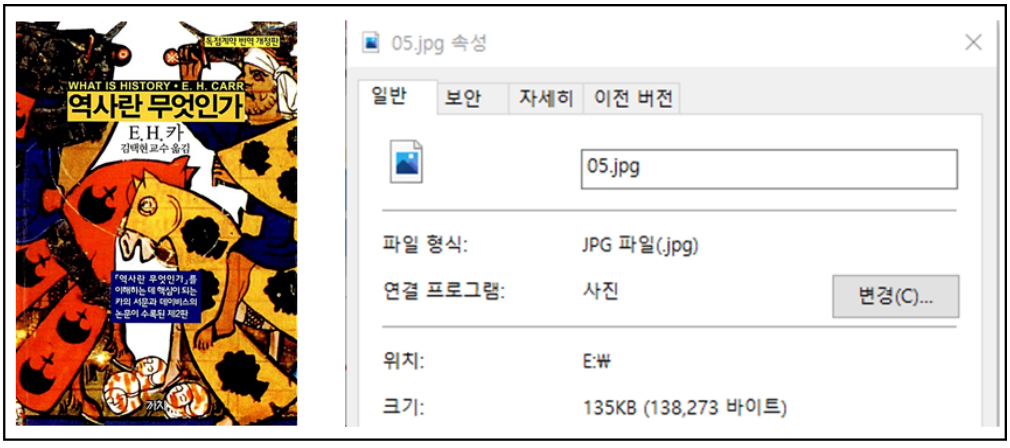
In this method, it is first converted into an array of Unsigned Integer 8-bit type, and then converted into Unsigned Integer 1 word (32 bit) type. The reason that file data needs to be converted into binary format in this way is that the JavaScript file stream basically uses a text-only string set such as UTF-8. This is because if you just hash the target data file without such a process, you will get a hash value that is completely different from the existing data file hash program. Convert the binary 1-word type array obtained as described previously into a *CryptoJS.lib.WordArray* object using the *CryptoJS.lib.WordArray.create()* function. This is because the hash encryption function provided by CryptoJS takes the form of this object as an argument. Afterwards, the function of the hash encryption algorithm selected in Figure 7(a) is executed using the function provided by the CryptoJS library [12].

The checksum value in Figure 7(b) is actually implemented as mentioned here. To prove that, Figure 11(a) shows the file information used in the test, and Figure 11(b) shows the result of calculating the checksum value of the file through Windows' CertUtil. It has been proven that the values of Figure 7(b) and Figure 11(b) are the same, so the checksum value calculation function of the actual system can be implemented in the manner described above.

It is also necessary to implement a function to check the checksum value and related information stored as a transaction in the blockchain. At this time, it is implemented using the RESTful API provided by the blockchain. Most of the RESTful APIs offered by blockchain platforms are provided as APIs that can be used in JavaScript considering the web environment, and this function is implemented using this. Some blockchain

platforms require additional implementations of necessary functions producing Smart Contracts to store and inquire transaction data.

(a)



(b)

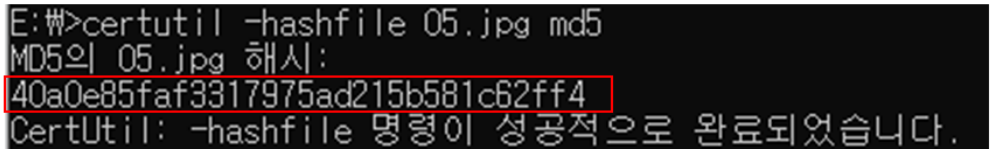


Figure 11. File information used for checksum functional verification experiment

5. CONCLUSION

In this paper, a checksum value is generated to ensure the integrity and reliability of a data file, and this value and related file information are stored in the blockchain. Afterwards, the research contents were introduced to design a system that provides the function that allows others to contrast with the target file by sharing the checksum values stored in the blockchain.

In this paper, a method using CryptoJS is adopted to obtain the checksum value of the target file. This can reduce the performance load of the system presented in this paper by processing everything on the user's computer and collecting only the information to be included in the actual block chain transaction data. And it was proposed to build a block chain based on Hyperledger Fabric. In addition, i was introduced the concept of a JavaScript-based ‘blockchain interface’ that stores transaction data and inquires transaction data in the built blockchain.

By providing a system that can guarantee the integrity of the information stored in the data file, various advantages are provided to users of this system. Typically, it can protect the user's computer from steganography attacks, and it prevents accidents that occur in various ways because the information stored in the data file is changed. This utilizes the characteristic of the block chain in which stored data cannot be changed. In the future, various assurance services using file integrity are possible in the system presented in this paper.

Efforts are being made to implement and operate this system in the future. Among the contents suggested by MEYLAN [6], additional research is being conducted to provide a ‘method that allows users to easily compare the target file with the checksum stored in this system’. And we are researching how to build a block chain based on Hyperledger Fabric 2.x. In addition, we are exploring ways to implement a 'blockchain interface' that can insert and inquire transaction data into this blockchain based on JavaScript. Additionally, we are devising a way to exercise of rights between users over transaction data stored in this blockchain. If

this is possible, NFT (Non-Fungible Token) is possible through this system [13-15].

REFERENCES

[1] D. Artz, "Digital steganography: hiding data within data," *IEEE Internet Computing*, Vol. 5, No. 3, pp. 75-80, May 2001. DOI: *https://doi.org/10.1109/4236.935180*

[2] Ubuntu 20.04.03 iso SHA256 Checksum, Ubuntu Official website, *https://releases.ubuntu.com/20.04.3/SHA256SUMS*

[3] The MD5 Message-Digest Algorithm, RFC (Request for Comments) Editor, *https://www.rfc-editor.org/info/rfc1321*

[4] Hash Functions, NIST, *https://csrc.nist.gov/projects/hash-functions*

[5] Hash result size, Hashnet Wiki, *http://wiki.hash.kr/*

[6] A. MEYLAN, M. CHERUBINI, B. CHAPUIS, M. HUMBERT, I. BILOGREVIC, K. HUGUENIN, "A Study on the Use of Checksums for Integrity Verification of Web Downloads," *ACM Trans. Priv. Secur.*, Vol. 24, No. 1, pp. 1-36, Jan. 2021. DOI: *https://doi.org/10.1145/3410154*

[7] Hyperledger Fabric, Read the Docs*, https://hyperledger-fabric.readthedocs.io/en/release-2.2/whatis.html*

[8] M.Y. Kim, H.S. Lee, J.D. Kim, "A Blockchain Copyright Information Registration System for Content Protection of Online Sharing Platforms,*" Journal of the Korea Institute of Information and Communication Engineering*, Vol.24, No. 12, pp.1718-1721, Dec 2020. DOI: *https://doi.org/10.6109/jkiice.2020.24.12.1718*

[9] M.Y. Kim, "A Study on the Design of LoRaWAN-based Public Blockchain Cryptocurrency Payment System," *The Journal of the Convergence on Culture Technology (JCCT),* Vol. 7, No. 1, pp.608-614, Jan 2021. DOI: *https://doi.org/10.17703/JCCT.2021.7.1.608*

[10] *S. Han, S. Kim, S. Pack*, "A GDPR based Approach to Enhancing Blockchain Privacy," *The Journal of The Institute of Internet, Broadcasting and Communication (JIIBC)*, Vol. 19, No. 5, pp.33-38, Oct. 2019. DOI*: https://doi.org/10.7236/JIIBC.2019.19.5.33*

[11] CryptoJS, *https://code.google.com/archive/p/crypto-js/*

[12] How to read a binary file with FileReader in order to hash it with SHA-256 in CryptoJS?, Stackoverflow, *https://stackoverflow.com/questions/33914764/how-to-read-a-binary-file-with-filereader-in-order-to-hash-it-with-sha-256-in-cr*

[13] Y. Lee, "Analysis on Trends of Artworks Blockchain Platform," *The International Journal of Advanced Culture Technology (IJACT)*, Vol.7, No.3, pp.149-157, Aug 2019. DOI: *https://doi.org/10.17703/IJACT.2019.7.3.149*

[14] K. C. Kim, "The Impact of Blockchain Technology on the Music Industry," *International journal of advanced smart convergence (IJASC)*, Vol. 8, No. 1, pp.196-203, Mar 2019. DOI: *https://doi.org/10.7236/IJASC.2019.8.1.196*

[15] Y. Lee, "Modeling of Artworks Blockchain Platform Using Colored Petri Net," *The International Journal of Advanced Culture Technology (IJACT)*, Vol. 8, No. 4, pp.242-248, Nov. 2020. DOI: *http://dx.doi.org/10.17703/IJACT.2020.8.4.242*