

Blockchain based Removable Storage Device Log Management System

Gyuseong Lee*, Minsung Son*, Nakhoon Choi*, Seongho Hong*, Heeyoul Kim*

* Department of computer Science and engineering, Kyonggi University, Korea

kfc7799@naver.com, sns2831@gmail.com, skg4463@gmail.com, 20191101100@kyonggi.ac.kr, heeyoul.kim@kyonggi.ac.kr

Abstract— since the use of removable storage devices, security of the storage devices has been required as needed. In response to this demand, a method of managing/controlling removable storage devices through software is being utilized. However, because common management software stores logs about the use of the storage device on a personal PC or on a central server, there may be situations in which insiders tamper with the logs after malicious behavior. This paper proposes a blockchain-based log management system to solve the problem of log tamperability of existing removable storage management software.

Keywords— blockchain, removable storage device, log, Hyperledger Fabric, log management system

I. INTRODUCTION

With the development of computers, the use of electronic documents increased explosively. The use of removable storage devices, which can easily copy and manage multiple materials, including these electronic documents, has also increased. This led to security incidents, such as the unauthorized copying of important data to and from mobile storage devices, and attacks such as Tickusb[1], which planted malicious codes to leak confidential information. Companies use methods such as Data Loss Prevention (DLP) [2] to protect themselves from these threats and prevent external leaks. In other words, it prevents insiders in the enterprise from leaking confidential information through malicious actions. However, security policies alone cannot ensure that all malicious behavior can be prevented, such as when insiders are highly empowered or able to control DLP through vulnerabilities, etc. In the event of a security accident caused by such a problem, the removable storage device use log is checked to find the cause of the accident and the main cause of the accident. The problem is to tamper with or delete the logs after a malicious act so that you don't know who caused the security accident where. The proposed system saves the collected logs in the ledger of the Hyperledger Fabric in real time using the existing DLP program. This ensures log integrity and persistence.

II. BACKGROUND

This chapter describes the basic knowledge of the blockchain and Hyperledger Fabric needed to understand the content.

A. Blockchain

Blockchain [3] is a data distribution processing technology that all users participating in the network distribute and store all data. Blocks become ledger that records transactions, which store the hash of the previous block. The blockchain connects these blocks sequentially and has a chain structure. The more participants store and share blocks, the more secure the data recorded in the blockchain. This is because in order to tamper with a blockchain network, it is necessary to attack a number of participants, not a central server or a few participants. For this reason, tampering with blocks is virtually difficult, and the data stored in the blockchain has integrity and permanence.

B. Hyperledger Fabric

Hyperledger Fabric [4] is an implementation of a distributed ledger platform for running smart chaincodes. As a private blockchain, only authorized participants can participate in the network. The ledger that stores the data may be kept open or confidential, depending on the identity of the participants.

- **Channel:** Channel is the intergroup communication mechanism of Hyperledger Fabric. Channel-specific ledger is accessible only to members of the consortium who have been granted permission to use the channel. This enables data separation and confidentiality.
- **Chaincode:** Chaincode initializes and manages the ledger state through transactions submitted by the application. It can be considered smart contract [5].
- **Membership Service:** Permissioned membership provides a trusted blockchain network, where participants know that all transactions can be detected and traced by authorized regulators and auditors.
- **Peer:** Peer is an entity that stores ledger and chaincode. The peer is assigned an identity from the CA and participates in the channel.
- **Fabric-CA:** Fabric CA issues X.509 [6] certificates of participants so that they can verify that they are permissioned by authenticating their digital identities.

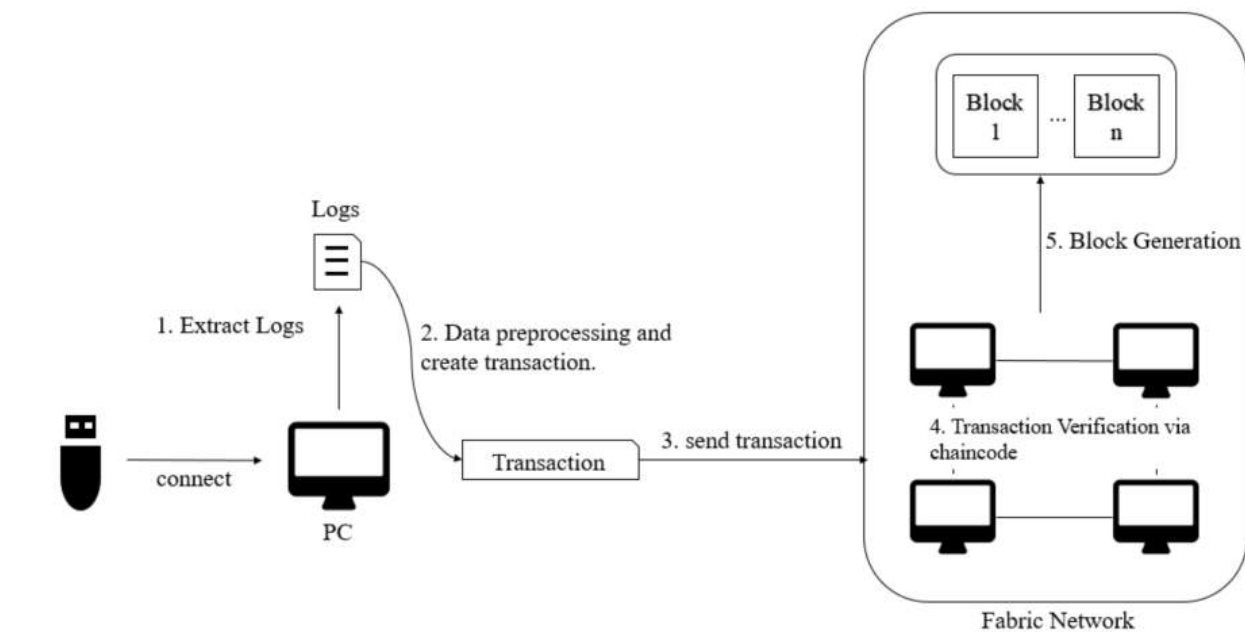


Figure 1. Log management system architecture

Since the consensus algorithm of the Hyperledger Fabric is pluggable, the consensus algorithm that satisfies the Byzantine fault tolerance [7] can be applied as desired such as PBFT (Practical byzantine fault tolerance) and CFT (Crash Fault Tolerance). We aim for a internal network, so we use Hyperledger Fabric, a private blockchain that limits participants rather than a public blockchain that anyone can participate freely.

III. LOG MANAGEMENT SYSTEM

This chapter describes the log management system and hyperledger fabric network configuration.

A. Hyperledger Fabric Network

The organization that wants to use the proposed system configures nodes for building blockchain networks. A node consists of an organization's trusted servers and assumes they are safe.

Users act as clients of blockchain networks. To satisfy this, users receive a certificate from the Certificate Authority (CA), which is responsible for issuing keys to the Hyperledger Fabric. The issuance of the certificate is made by the administrator, which gives only authorized users the right to access the blockchain network.

B. Log Management System

The proposed log management system ensures the integrity of the logs by propagating the user's use logs from the removable storage to the blockchain.

The use logs of removable storage can be obtained through global hooking and software, and in this paper the logs are extracted using the Endpoint Protector Basic, an endpoint security solution, and the blockchain platform is to be used. The overall flow of the proposed log management system is

shown in Figure 1. When a user connects a removable storage to a computer, global hooking, or using software, extracts the activity record and model name of a connected removable storage device. The data extracted through the pre-processing process is processed to fit the parameters of the chain code and then propagated through the transaction to the blockchain network. At this point, the above process is automatically carried out. Transmitted transactions are recorded in the ledger through the consensus process, and recorded data is guaranteed to be integrity. The data stored in the ledger is shown in Table 1, and the data transmitted as parameters of the chain code is the same.

TABLE 1. PARAMETERS

parameter	explain
Timestamp	Log occurrence time
Action	User behavior
UserInfo	The information of the user
FileInfo	The information of the file
DeviceInfo	The information of the removable device

Figure 2 shows the pseudocode of the chaincode used in the proposed system. getUserInfo() is a function provided by Hyperledger and identifies the user's identity through the signature of the transaction. This is possible because the CA issued the certificate. putState() means a function that records data in the ledger. By recording the activity logging of the user's information and files received from the signature, users can record what they have done about the file.

Input : params // all parameters in table 1

Output : None

Algorithm :

```
UserInfo := getUserInfo() // from Signature
putState(params, UserInfo)
```

Figure 2. pseudocode of chaincode

7 functions are implemented below to look up the logs stored in the ledger for each input value.

- getAllLogs()
- getLogByTimestamp(timeFrom, timeTo)
- getLogByWallet(wallet)
- getLogByFilename(filename)
- getLogBySerial(serial)
- getLogByAction(action)
- getLogByFiletype(filetype)

Inside each function, the getState or getStateByRange function is used to query the data as opposed to the putState function used when recording data in the ledger. In addition to the above functions, additional log lookup functions for the desired value can be created.

IV. IMPLEMENT

This section shows implementation based on the content of this paper. In real-world systems, whenever new logs are created, the chaincode must read the logs in real time to send transactions, but here it is not implemented in real time to check for simple operation. When used in practice, it should be implemented in real time. Otherwise, there is a risk that logs will be logged and that the logs will be tampered with before the transaction is sent. The consensus algorithm used SOLO. The test was conducted in Ubuntu 16.04 and the source code was javascripted.

A. Invoke Transactions

```
hyperledger@ubuntu:~/fabric-log/fablog/javascript$ node invoke.js
Wallet path: /home/hyperledger/fabric-log/fablog/javascript/wallet
Transaction has been submitted
Transaction has been submitted
Transaction has been submitted
Transaction has been submitted
```

Figure 3. Invoke transaction

Read the Logs.csv file and run the invoke.js file to send the transaction containing the logs in Figure 3.

Wallet is created with the name user1 and permissioned on the network. Transfers transactions with parsed data for username and logs.csv.

B. Query Transactions

Figure 4 describes the transaction query recorded in the ledger. The transaction contains information such as action, fileInfo, userInfo, timestamp, deviceInfo. If you look at the log, you can see that LOG99 recorded in the ledger that user1 read the Grab.g4b file on 2019-09-25 20:19.

```
hyperledger@ubuntu:~/fabric-log/fablog/javascript$ node query.js user1 LOG99
Transaction has been evaluated, result is:
[{"Key": "user1LOG99",
  "Record": {
    "docType": "log",
    "action": "read",
    "timestamp": "2019-09-25 20:19",
    "userInfo": [
      { "wallet": "user1",
        "user": "DESKTOP-H1LE11A$",
        "userDesc": ""
      }
    ],
    "fileInfo": [
      {
        "filename": "J:_ISO\\e2b\\grub\\Grab.g4b",
        "filetype": ".g4b file"
      }
    ],
    "deviceInfo": [
      {
        "serial": "11AJFH5RBMHC3AV0",
        "pid": "TRANSCEND_64GB (1000)",
        "vid": "JETFLASH (8564)",
        "desc": "JETFLASH TRANSCEND_64GB"
      }
    ]
  }
}]
```

Figure 4. Query transactions

V. CONCLUSIONS

The existing system uses DLP to prevent critical data leakage and, if this data is leaked through a removable storage device, check the logs to determine where and who leaked it. However, it was possible to obscure the source of leakage by deleting or tampering with these logs, and to prevent this, we proposed a blockchain based removable storage log management system that guarantees the integrity of logs. This system will be useful to prevent leakage of confidential information through removable storage devices in an environment that uses internal networks such as companies.

ACKNOWLEDGEMENT

This work was supported by the GRRC program of Gyeonggi province. [GRRC KGU 2017-B04, Image/Network-based Intellectual Information Manufacturing Service Research]

REFERENCES

- [1] (2019)AhnLab ASEC REPORT website. [Online].Available: https://global.ahnlab.com/global/upload/download/asecreport/ASEC%20REPORT_vol.95_ENG.pdf
- [2] Liu, S., & Kuhn, R, "Data Loss Prevention", *IT Professional*, Vol.12, pp. 10 – 13, 2010.
- [3] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system, 2009," 2012. [Online]. Available: <http://www.bitcoin.org/bitcoin.pdf>
- [4] Christian Cachin, Architecture of the Hyperledger Blockchain Fabric. [Online]. Available:https://www.zurich.ibm.com/dccl/papers/cachin_dccl.pdf, 2016
- [5] SZABO, Nick. Smart contracts: building blocks for digital markets. *EXTROPY: The Journal of Transhumanist Thought*,(16), 1996, 18: 2.
- [6] Greck, E. et al., "Overview of Certification Systems: X.509, Infrastructure and Its Application." *IEEE Comput. Soc. CA, PGP and SKIP*". Aug. 1, 1998, MCG, pp. 2-15.[Online].Available : https://www.researchgate.net/profile/Ed_Gerck/publication/318700731_original_web_site_Overview_of_Certification_Systems/data/597829520f7e9b277721d8ce/certhm.pdf

- [7] C. Miguel and L. Barbara, "Practical byzantine fault tolerance," in Proceedings of the Third Symposium on Operating Systems Design and Implementation, vol. 99, pp. 173–186, 1999.



Gyuseong Lee is in the undergraduate course of Computer Science Department, Kyonggi University , His major research interests include security and blockchain.



Minsung Son is in the master course of Computer Science Department, Kyonggi University , His major research interests include security and blockchain.



Nakhoon Choi is in the undergraduate course of Computer Science Department, Kyonggi University , His major research interests include security and blockchain.



Seongho Hong is in the master course of Computer Science Department, Kyonggi University , His major research interests include security and blockchain.



Heeyoul Kim received the B.E. degree in Computer Science from KAIST, Korea, in 2000, the M.S. degree in Computer Science from KAIST in 2002, and the Ph.D. degree in computer science from KAIST in 2007. From 2007 to 2008, with the Samsung Electronics as a senior engineer. Since 2009 he has been a faculty member of Department of Computer Science at Kyonggi University. His major research interests include cryptography, security and blockchain.