Blockchain-Based System and Methods for Sensitive Data Transactions

Xin Su[1], Inam Ullah[1], Meiling Wang[1], and Chang Choi[2*]

[1] College of Internet of Things (IoT) Engineering, Hohai University (HHU), Changzhou Campus, 213022, China, (e-mail: leosu8622@163.com; inam.fragrance@gmail.com; 2960661135@qq.com)
[2] Department of Computer Engineering, Gachon University, 13120, Republic of Korea (e-mail: changchoi@gachon.ac.kr)

***Abstract* - A blockchain-based processing framework for sensitive data is proposed. The smart network comprises a blockchain module, an electronic contract (E-contract) layer node, and a Software-as-a-Service (SaaS) layer module. The underlying blockchain module provides technical support, such as virtual machines, consensus algorithms, transaction verification mechanisms, and accounting mechanisms. The E-contract layer module provides a distributed application service and uses the blockchain technology (BT) to support it. In addition, it runs the code of the sensitive data transaction system and the code created by the algorithm for encoding generation at the E-contract layer. The SaaS layer module offers a cloud-platform service that allows each party to easily engage in business collaboration via web portals or clients. The proposed smart system is used by each party get involved in the production of sensitive data. The final sensitive data are produced by the final data generator, and other modules involved in the process of data production are unaware of the final data. This approach prevents the leakage of sensitive data into the circulation.**

I. INTRODUCTION

ANTI-counterfeiting traceability has been an essential production feature of traditional enterprises [1]-[2]. All goods produced by formal enterprises include anti-counterfeiting two-dimensional (2D) or one-dimensional (1D) code labels that are used for anti-counterfeiting traceability queries [3]. The label content is a digital encoding (DE) string. Currently, industrial DEs are typically generated by system integrators before being delivered to printing factories. The printing factories use the DEs to produce paper or plastic anti-counterfeiting labels and return them to industry clients or manufacturers. Throughout the label manufacturing process, industry clients use the hardware and software of the system integrators to gather information relevant to the DE on the labels before the products are shipped out. In the consumption phase, consumers can identify DEs in various ways (for example, by scanning) and query the commodity information in the database to assess the authenticity of the commodity. This information is provided by system integrators.

The aforementioned transaction involves a considerable risk of leakage during the DE production of the 2D code [4]. Machine integrators, which supply DE files directly to printing factories or industry consumers, constitute a significant source of risk of human leakage in the circulation of DE files. Once DE packets are leaked, it is impossible for system integrators, printing factories, and other parties to prove their innocence. In addition, industry clients typically authorize multiple agents to subcontract their businesses. The occurrence of fake agents can be highly problematic for system integrators, causing significant economic losses both upstream and downstream.

At present, in response to the privacy leakage phenomenon caused by DE files in the circulation process, researchers have introduced blockchain technology [5]. At the same time, homomorphic secret sharing [6] and secure multi-party computing [7] technologies are jointly used. However, most of the existing homomorphic secret sharing and secure multi-party computing technologies have the problems of massive communication rounds and too much traffic load. This inevitably causes the operating efficiency loss based on blockchain platform. Besides, the blockchain platform uses plaintext and ciphertext [8] methods for transactions. If the plaintext is used, the system face the risk of exposing data privacy. Otherwise, if the ciphertext is used, although data privacy is protected, it is difficult to support homomorphic computing. Accordingly, the phenomenon of DE circulation in modern industries encounter the following problems:

- There is a risk of leakage of sensitive data in the circulation.
- System integrators and printing factories cannot

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MCE.2021.3076985, IEEE Consumer Electronics Magazine

be well documented, and the reliability of upstream agents cannot be identified.

- After the DE data are generated, they are stored in the database of each system integrator for future reference and are susceptible to tampering by workers.
- The existing technology has high complexity in processing sensitive data and low operating efficiency.

In this article, we propose a transaction framework for sensitive data and a blockchain-based process [9-11]. The smart network comprises a blockchain module, an electronic contract (E-contract) layer node, and a Software-as-a-Service (SaaS) layer module. The blockchain module offers technological support, such as virtual machines, consensus algorithms, and mechanisms for transaction–verification. A distributed application service is provided by the E-contract layer module. In addition, this runs the sensitive data transaction framework code and the code generated at the E-contract layer by the algorithm to encode the generation. The SaaS layer module provides a cloud-platform service that facilitates each party to participate in business communication through web portals. The smart system uses the blockchain to record the actions and participation of each party in generating DE information. No party can access the DE information until it is printed as labels, thereby preventing any human leakage of the DE information. Specifically, the contributions of this paper are as follows:

- We propose a processing framework for sensitive data based on blockchain. The overall complexity of processing sensitive data in the existing blockchain technology is reduced.
- For chain privacy issues, we propose an encoding algorithm for processing sensitive data involved in transactions. The risk of leakage of sensitive data in the circulation is prevented.
- We introduced the calculation of the algorithm matrix using the computing resources of all nodes in the blockchain. The large-scale graphics processing unit (GPU) resources in blockchain are fully utilized, and the operating efficiency and applicability of the algorithm are improved.

II. Contents of The Smart system

The blockchain-based sensitive data transaction system includes the following modules: an underlying blockchain module, an E-contract layer module, and a SaaS layer module. The underlying blockchain layer module provides support for the blockchain technology, which includes virtual machines, consensus algorithms, transaction verification mechanisms, and accounting mechanisms [12]. The blockchain layer proposed in this article is not a public blockchain (to which Bitcoin and Ethereum can connect anonymously for transactions [13]-[15]), but rather a hyperledger-based consortium blockchain (CB). Therefore, each party involved in the blockchain requires the authentication and authorization of the smart system to store information on the blockchain for business and individual subjects. Enterprises can build their nodes to join the blockchain and lease the nodes provided by the smart system to access the blockchain. Furthermore, they can use the underlying blockchain services, such as the underlying distributed storage of the E-contract layer, the support of a consensus algorithm, and the provision of contract-based containers, particularly when using credit endorsement, operation, critical data records, and sensitive data transactions.

Blockchain technology is one of the biggest technological breakthroughs in this century. Bitcoin is the first blockchain technology that allows transactions by a network of users without needing the trust of others on the network or a third party. Everything is encrypted; as a result, nobody can tamper with the blockchain without others instantly knowing. A traditional blockchain network is composed of several nodes that do not fully trust each other. Some of the nodes exhibit Byzantine behavior, but most are truthful. The nodes together maintain a collection of mutual and global states and perform transactions that change the states. Blockchain is a special data system that stores transactions and historical states. All system nodes agree on the transactions and their orders.

The E-contract layer module provides a trusted distributed application service. There is a strong link between distributed applications and business. The application code is available to participants, who perform business according to the agreement algorithms. The E-contract module of this smart system focuses on sensitive data services, such as DE and time cards, which can be controlled by algorithms. In addition, it runs the code of sensitive data transactions and the code of the DE-generation algorithm. The SaaS layer module provides a cloud platform service [16] that allows each party to easily participate in business collaborations via web platforms or clients. Each aforementioned module provides a call service to the upper layer through the interfaces. All the participants are certified on the SaaS platform. The certification is a CA certificate issued by authorities and includes identity authentication through facial and social attributes (for example, identity cards or business subjects).

A. Other modules of the smart system

The smart system also includes a DE transaction business layer module. In this module, the SaaS platform calls the business E-contracts stored in the blockchain through the SDK or API [17], [18] to record the entire process of the transaction. The code-generation delivery module ensures that the printing factories have the final DE data. Finally, after the printing factories have delivered anti-counterfeiting labels, the code verification process calls for the query interface of the E-contract layer to verify the final DE. The smart system also includes a code-generation delivery module, which comprises the following submodules:

1) The industry-user submodule generates anti-counterfeiting label orders according to the requests of industry users. These orders call the E-contract interface through the SaaS layer. The blockchain E-contract of the anti-counterfeiting label order information is stored in the blockchain. According to the order information, the industry-user submodule is automatically assigned a randomly generated number, A_1, using a hash algorithm. This number is stored in the E-contract for subsequent code generation and is invisible to the public.
2) The agent–user submodule transfers the anti-counterfeiting label orders to the system integrator submodule through the platform according to the confirmation of the agent users. Subsequently, the agent-user submodule places a DE order with the system-integrator submodule, and the order information is stored in the blockchain E-contract through the DE order interface. Based on the order information, the smart system obtains a randomly generated number, A_2 , using the hash algorithm, which is stored in the E-contract for subsequent code generation.
3) The system-integrator submodule confirms the legality of the order, verifies that the customer is authorized by the industry, confirms the order to determine the agent, and performs the following transactions after receiving digital orders. First, if the system integrator has its own seed code, it can choose to upload it to the blockchain E-contract, which is automatically generated by the system integrator. If the system integrator does not have a seed code, it is automatically generated by the E-contract in the system. The DE order is then transferred to an agent. The seed code is denoted as $Seed_N$, where N is the number of digits.
4) The printing submodule generates a random number, A_3, for the order using the hash algorithm, according to the confirmed order. This number is stored in the E-contract. The printing submodule includes a code-generation submodule, a final anti-counterfeiting DE download submodule, and a direct-connection printer module. The code-generation submodule, which comprises an encoding-conversion module and code-generation-function submodule, generates code according to the stored random numbers (A_1, A_2, A_3) and function groups created by the code-generation algorithm of the E-contract. The conversion-encoding module converts the results of the order hash of each participant into numbers. This yields a set of parameters to transform the E-contract function. The final anti-counterfeiting DE download submodule provides the final anti-counterfeiting encoding download.
5) The agent-confirmation submodule transfers the order to the printing submodule to perform the production task only after the reception of the agent's order is confirmed.

III. SENSITIVE DATA ENCODING

The code-generation submodule comprises an encoding-conversion module and a submodule generation function. The encoding-conversion module converts the order hash results of each party into numbers that form the parameter set of the E-contract conversion function. Based on these numbers, the function-generation submodule uses an encoding algorithm to encode the sensitive data in the transaction. In this section, we provide the details of sensitive data encoding. To compare the proposed work, for security and safeguarding [19],[20], the authors presented sensitive data encoding for e-commerce. A BT is applied to smart contacts and electronic transactions, and a mechanism of association is introduced between digital and physical assets. The proposed technique is capable of data security and establishes trust among entities. The proposed E-contract for the generation of code is irreversible, and the specific steps are as follows:

Step 1 The length of the code to be generated is set to M, the length of the seed code to $N(M > N)$, and the length of a random number to L.

Step 2 $Seed_N$ is divided into $Z = M - N$ groups, each of which corresponds to a code-generation function. In each group, each code is split into N-dimensional variables. These are padded with "1" to the left, up to the M^{th} bit. This result, recorded as X , becomes the input of the code-generation function in group *Z*. In the code-generation function group, the function is a polynomial in X . In Eq. (1), j is randomly

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MCE.2021.3076985, IEEE Consumer Electronics Magazine

selected in the range $[0, Z+1]$, and k and b are M-dimensional vectors, which are selected from A_1, where A_1 is a set of random numbers A. Then, k and b are split into L-dimensional variables, from which the encoding algorithm selects to generate M bits according to the step size of the round ($M = N+1$). According to the code-generation function, the time complexity of the calculation depends mainly on X^j, and j is related to Z. The elements of the set can be selected randomly using either the linear congruential or middle-square methods. K and B are the digital matrices corresponding to k and b, respectively. The operations of K and B can be calculated using a matrix calculation.

Step 3 The seed and code-generation submodules of the functional groups are called. Then, all the outputs of $F^Z(X)$ are calculated. The final number is obtained by combining each dimension of the code-generation contract. This records the binding relationship between X and Y, which is then used to check the repeated code and subsequent quick-verification code. The above systematic algorithm is illustrated in Figure 1.

$$\begin{cases} F^1(X) = X^j \times k_1 + b_1 \\ \cdots \\ F^{Z-2}(X) = X^j \times k_{Z-2} + b_{Z-2} \\ F^{Z-1}(X) = X^j \times k_{Z-1} + b_{Z-1} \\ F^Z(X) = X^j \times k_Z + b_Z \end{cases} \quad (1)$$

Step 1 The length of the encoding to be generated is set to M, the length of the seed code to $N(N < M = 3)$, and the length of a random number to L. In each group, each code is split to give an N-dimensional variable that is then padded with "1" to the left, up to the M^{th} bit. This result, recorded as X, is then the input of the code-generation function in group Z. In the code-generation function group, the functional form is a linear monotonic function of X. In Eq. (2), k and b are separate N-dimensional vectors, and K and B are separate digital matrices corresponding to k and b, respectively. Each pair refers to a pair of k and b that is a randomly selected from a set of random numbers A. Then, k and b are split into L-dimensional variables, from which the algorithm to generate M bits according to the step size of the round $(M = N+1)$. For each code-generation function, a random M-dimensional number G is generated. Each dimension covers the range $[0, M]$, where "0" signifies that the corresponding code bit will not be exchanged, and a nonzero number indicates the bit that will be exchanged. G is used for the corresponding code bit exchange of the final code-generation combination in Step 3.

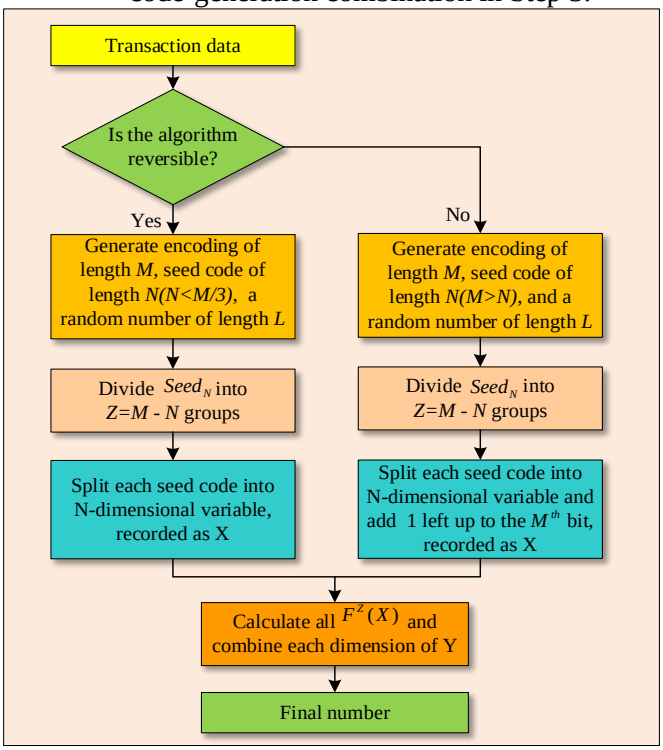

Fig. 1: The proposed systematic algorithm method.

Step 2 All the outputs of $F^Z(X) = Y$ are calculated, and the final results from each dimension of Y are modulated. In the modulation process, each dimension is shifted by three bits upward while padding with zeros to the left. The DE is modulated using contrast G to exchange bits of code to obtain the final DE. If a created DE matches an existing one, a new G is produced to ensure that no existing DE is repeated.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MCE.2021.3076985, IEEE Consumer Electronics Magazine

$$\begin{cases} F^{1}(X) = X \times k_1 + b_1 \\ \cdots \\ F^{Z-2}(X) = X \times k_{Z-2} + b_{Z-2} \\ F^{Z-1}(X) = X \times k_{Z-1} + b_{Z-1} \\ F^{Z}(X) = X \times k_Z + b_Z \end{cases} \quad (2)$$

The E-contract code generation retains the functions of G and code generation to measure X according to Y and for subsequent demand.

IV. BLOCKCHAIN-BASED TRANSACTION METHODS

This section provides a detailed description of blockchain-based transaction methods for sensitive data. The first approach is as follows.

Step 1 The first party can obtain a random number, A_1, using a hash algorithm based on order information. The random number is stored in the E-contract and is invisible to the public.

Step 2 The second party can obtain a random number, A_2, using the hash algorithm based on the order information stored in the E-contract. The random number is used for subsequent code generation.

Step 3 If a third party has a seed code, they can choose to upload it to the blockchain E-contract, which is generated automatically by the third party. If this party does not have a seed code, the system E-contract automatically generates one. The DE order is then transferred to the second party.

Step 4 The fourth party can obtain a random number, A_3, using the hash algorithm. Random number A_3 is stored in the E-contract. Subsequently, the fourth party generates code according to the random numbers A_1, A_2, A_3 and function groups, which are created by the code-generation algorithm in the E-contract.

The basic steps of the second approach are as follows.

Step 1 Industry users log into the industry user submodule and place an anti-counterfeiting label order with the agent by calling the E-contract interface through the SaaS layer. The order information in the blockchain E-contract is stored on a blockchain platform. According to the order information, the system generates a random number, A_1, using a hash algorithm, which is then stored in the E-contract.

Step 2 The agent logs into the agent-user submodule, and the anti-counterfeiting label order is transferred to the agent account. The agent confirms the order and transfers the order to the system integrator through the platform. Then, the agent places a DE order with the system integrator. The order information is also stored in the blockchain E-contract through the DE order interface. The system generates a random number, A_2, based on the order information hash, which is stored in the E-contract for subsequent code generation.

Step 3 The system integrator logs into the system-integrator submodule. After receiving the DE order, the system integrator confirms the legality of the order and checks whether it is authorized by industry users. Then, the system integrator confirms the orders of the agent and performs the following operations: First, if the system integrator has its own seed code, it can choose to upload it to the blockchain E-contract, which is automatically generated by the system integrator. If the system integrator does not have a seed code and the seed code is generated automatically by the system E-contract, then the DE order is transferred to the agent.

Step 4 After logging into the printing submodule, the printing factory confirms that the order is accepted. It then generates the order and also a random number, A_3, using a hash algorithm. The random number is stored in the E-contract and is invisible to the public. Subsequently, the code generation E-contract of the printing submodule generates code according to the random numbers A_1, A_2, A_3 and function groups created by the code generation algorithm in the E-contract. The printing factory downloads the final anti-counterfeiting code on the system and imports it into the printer to print the anti-counterfeiting label. By directly connecting to a printer using E-contract for printing, the printing factory completes the coding delivery, which is monitored by industry users.

V. SPECIFIC IMPLEMENTATION CASES

A. *Case 1*

A sensitive blockchain-based data transaction system comprised an underlying blockchain layer, an E-contract layer, and a SaaS layer. The underlying blockchain module provided support for the blockchain technology,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MCE.2021.3076985, IEEE Consumer Electronics Magazine

including virtual machines, consensus algorithms, transaction verification mechanisms, and accounting mechanisms. The E-contract layer module provided a distributed application service, which included using the blockchain technology endorsement, running the smart-system code, and the DE generation algorithm code on the E-contract layer.

Each party participated in a business collaboration either through a web platform provided by the SaaS layer module or through clients. Each layer provided a calling service to the upper layer via an interface. The participants clarified their role in participating in the blockchain by placing their identity on the SaaS platform. By calling
the SaaS platform business, they stored their identity information and operations in blockchain certificates. At the DE transaction business layer, the SaaS platform recorded the entire transaction process in the blockchain by calling the E-contract. This ensured that the printing factory had the final DE data.

Finally, after the printing factory delivered anti-counterfeiting labels, the code verification process verified the final DE by calling the interface of the E-contract layer. The code-generation delivery phase is as follows. The participants were industry users, agents, system integrators, and printing factories.

1) The user logged into the smart system and placed anti-counterfeiting label orders to the agent by calling the E-contract to the interface via the SaaS layer. The order information on the blockchain E-contract was stored on a blockchain platform. According to the order information, the system automatically obtained a random number, A_1, using a hash algorithm. The random number was stored in the E-contract, which was invisible to the public.
2) The anti-counterfeiting label order was transferred to the agent account. The agent logged into the agent–user submodule, confirmed the order, and forwarded it to the system integrator through the platform. The agent then placed a DE order with the system integrator. The order information was stored in the E-contract blockchain through the DE order interface. The system obtained a random number, A_2, based on the hash order information. This random number was stored in the E-contract for the next generation of code.
3) The system integrator logged into the submodule of the system integrator. Once the DE order was received, the system integrator checked to confirm the legality of the order and whether it was authorized by the users of the industry. The system integrator then determined the order of the agent and performed the following operations: First, if the system integrator had its seed code, it uploaded it to the blockchain E-contract, which was automatically generated by the system integrator. If the system integrator did not have a seed code, it was automatically generated by the system E-contract; the DE order was then transferred to the agent.
4) Upon receipt of the order, the agent confirmed its acceptance and forwarded it to the printing factory for production tasks.
5) The printing factory logged into the system, confirmed the acceptance of the order, and then generated a random number, A_3, for the order hash. This number was stored in the E-contract. Subsequently, the
printing factory performed the following actions: First, the printing plant downloaded the final anti-counterfeiting DE from the system and then imported it into the printer to print the anti-counterfeiting labels. Thus, although the printing plant cannot prove its innocence, the other parties involved can. Second, the process of printing factory operations on the system was connected directly to the printer by the E-contract. It completed encoding delivery under the supervision of the users. In the aforementioned two steps, the code-generation E-contract generated a random number code A_1, A_2, A_3 and function groups created by the code generation algorithm in the E-contract. A flowchart of the code-generation E-contract is shown in Figure 2.

The random numbers $A: A_1, A_2, A_3$ are obtained by the algorithm that converts the strings into the DE. The encoding scheme can be based on ASCII or other digital codebooks. The contract must store a one-to-one mapping of the seed code and the encoding result for the proposed irreversible code-generation algorithm. Reversible algorithms are more susceptible to cracking. However, the calculations are simpler than those of an irreversible algorithm, and the DE result does not need to be stored, which saves storage space.

Figures 3 and 4 illustrate the flow of the irreversible and reversible algorithms, respectively. In the mining process of blockchain-based trading, the aforementioned steps are followed. Using the proposed method, the algorithm matrix calculations can use the computing resources of all the nodes in the blockchain to perform calculations in parallel and then summarize the results. This completely exploits the massive GPU resources on the blockchain, which is a major benefit of the calculation.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MCE.2021.3076985, IEEE Consumer Electronics Magazine

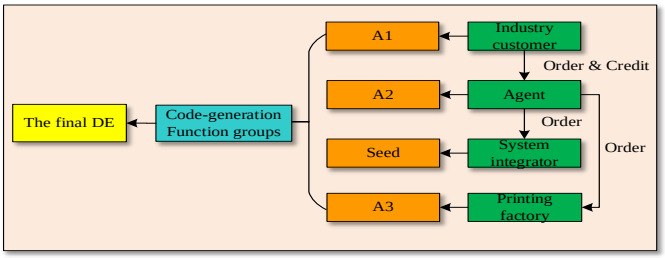

Fig. 2: Flowchart of the code-generation E-contract.

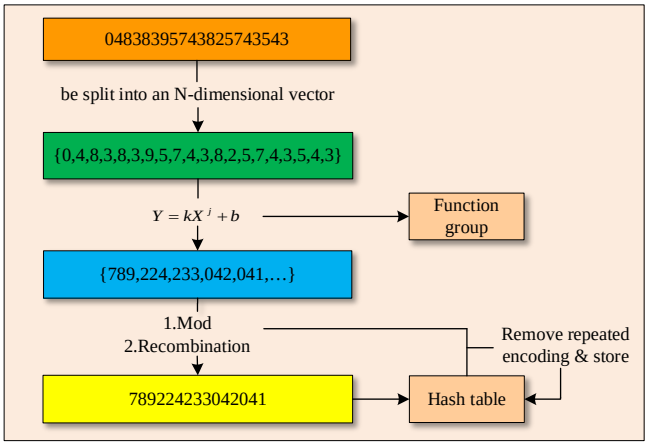

Fig. 3: Flow of the irreversible algorithm for sensitive data encoding.

In addition to the release of the authorization information in Figure 5, each role can only see its own input- or output-related information.

Approximately 50 students were tested at Hohai University to simulate the proposed blockchain-based P2P transaction method. The questionnaire survey performed on the students after a one-month test, summarized in Table 1, indicated that the students did not consider the conventional method of electronic trading as being fair and did not want to pay fees to a third-party service. Several students favored the decentralized framework and wanted a perfect transaction credit network to be set up. Most students stated that our proposed solution solved the problems of information leakage and data theft and would like to join the program to upgrade the current electronic trading framework.

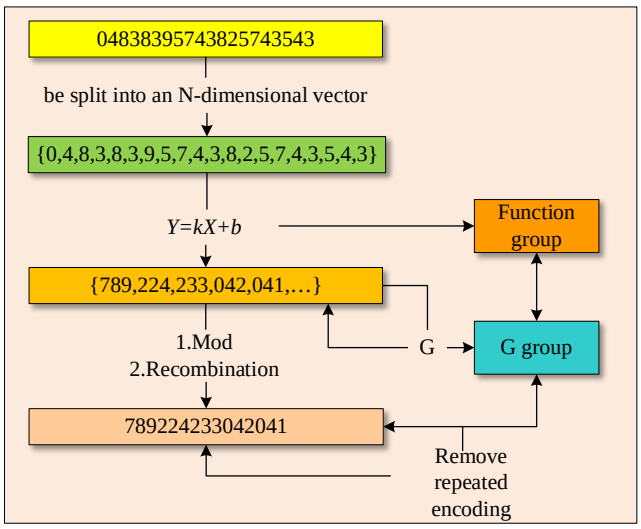

Fig. 4: Flow of the reversible algorithm for sensitive data encoding.

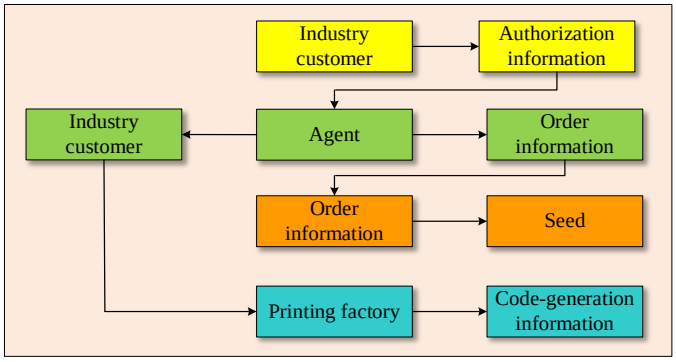

Fig. 5: Access privileges of each party to sensitive data.

B. Case 2

The industrial client, designated as User-A (for example, Liquor Enterprise), placed an anti-counterfeiting label order with the agent or directly with the device integrator and the printing plant/factory. The agent was designated as User-B for the industry customer and as User-A for the device integrator and printing factory. Industry accepted an anti-counterfeiting label order from User-A, which was outsourced to the system integrator and the printing factory. The system integrator was User-B, which supplied algorithms and raw data, produced DE based on measurements, and then delivered the digital packet to the printing plant or agent. The printing plant used was User-B. Upon receiving the digital packet from the system integrator, the 2D data was repackaged by the customer or agent, arranged for printing, and shipped to the customer or agent of the industry according to the request of the order edition.

Each role registered an account to join the smart system and perform the following actions. The industry customer placed an order with the agent. The agent placed an order with the system integrator. After confirming the receipt of the order, the system integrator uploaded the seed

encoding or used the seed encoding generated by the system. The agent placed an order with the printing factory for production. The printing plant produced the code and performed encoding. Then, the business client, contractor, network integrator, and printing plant performed the basic steps of the second approach to blockchain-based transaction methods.

TABLE I: Questionnaire and survey result.

Question asked during survey	Answers	Value
Is the current model of electronic commerce realistic?	Yes	3.0%
	No	97.0%
Is it necessary to integrate the digital economy and the real economy?	Yes	98.0%
	No	2.0%
Do you want to pay electronic transactions fees to a third-party platform?	Yes	3.0%
	No	97.0%
Would you like to collect your personal details from third-party platforms?	Yes	1.0%
	No	99.0%
Would you like to set up a perfect credit system for transactions?	Yes	98.0%
	No	2.0%
Is decentralization necessary for economic activities?	Yes	100%
	No	0.0%
Would you like to enter a program to upgrade the current e-trading system?	Yes	92.0%
	No	8%
Does our proposed approach solve information leakage problems and data tampering?	Yes	99.0%
	No	1.0%

VI. Conclusions

Current industrial settings are significantly susceptible to leakage during the manufacturing phase of 2D encoding. The proposed framework described the operation of each party through a blockchain and produced DE according to the participation of each party. This approach prevented the leakage of sensitive data in circulation. Meanwhile, each party could claim its innocence by invoking the traceability of blockchain operations. We proposed an encoding algorithm for treating the sensitive data involved in a transaction. We presented the calculations of the algorithm matrices that used the computational resources of all the nodes in the blockchain to perform the calculations in parallel and then summarize the results. This completely exploited the massive GPU resources in the blockchain, thereby improving the efficacy and applicability of the calculations.

Acknowledgment

This research was supported by Global Infrastructure Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Science and ICT(NRF-2018K1A3A1A20026485) and by the Gachon University Research Fund of 2019(GCU-2019-0795).

References

[1] P. J. Taylor, T. Dargahi, A. Dehghantanha, R. M. Parizi, and K. K. R. Choo, "A systematic literature review of blockchain cyber security," *Digital Communications and Networks*, vol. 6, no. 2, pp. 147-156, 2020.

[2] D. Liu and J. Lee, "CNN based Malicious Website Detection by Invalidating Multiple Web Spams," *IEEE Access*, vol. 8, no. 1, pp. 97258-97266, 2020.

[3] W. Martin, V. Friedhelm, and K. Axel, "Tracing manufacturing processes using blockchain-based token compositions," *Digital Communications and Networks*, vol. 6, no 2, pp. 167-176, 2019.

[4] D. Puthal, N. Malik, S. P. Mohanty, E. Kougianos and G. Das, "Everything You Wanted to Know About the Blockchain: Its Promise, Components, Processes, and Problems," *IEEE Consumer Electronics Magazine*, vol. 7, no. 4, pp. 6-14, 2018.

[5] L. Peng, W. Feng, and Z. Yan. (2020). Privacy preservation in permissionless blockchain: A survey. *Digital Communications and Networks*. [Online]. Available: https://doi.org/10.1016/j.dcan.2020.05.008.

[6] N. Kakade and U. Patel, "Secure Secret Sharing Using Homomorphic Encryption," in *Proc. 2020 11th International Conference on Computing, Communication and Networking Technologies*, 2020, pp. 1-7.

[7] S. Sundari and M. Ananthi, "Secure multi-party computation in differential private data with Data Integrity Protection," in *Proc. 2015 International Conference on Computing and Communications Technologies*, 2015, pp. 180-184.

[8] S. Jiao, T. Lei, Y. Gao, Z. Xie and X. Yuan, "Known-Plaintext Attack and Ciphertext-Only Attack for Encrypted Single-Pixel Imaging," *IEEE Access*, vol. 7, no.2, pp. 119557-119565, 2019.

[9] S. Kaushik, and S. Puri, "Online transaction processing using enhanced sensitive data transfer security model," in *Proc. 2012 Students Conference on Engineering and Systems*, 2012, pp. 1-4.

[10] W. Zheng, Z. Zheng, X. Chen, K. Dai, P. Li and R. Chen, "NutBaaS: A Blockchain-as-a-Service Platform," *IEEE Access*, vol. 7, pp. 134422-134433, 2019.

[11] F. Casino and C. Patsakis, "An Efficient Blockchain-Based Privacy-Preserving Collaborative Filtering Architecture," *IEEE Transactions on Engineering Management*, vol. 67, no. 4, pp. 1501-1513, Nov. 2020.

[12] D. Chkliaev, J. Hooman and P. van der Stok, "Mechanical verification of transaction processing systems," in *Proc. ICFEM 2000. Third IEEE International Conference on Formal Engineering Methods*, 2000, pp. 89-97.

[13] S. Zhang, and J H. Lee. "Mitigations on Sybil-based Double-spend Attacks in Bitcoin," *IEEE Consumer Electronics Magazine*, vol.7, no. 2, pp. 1-1, 2020.

[14] X. Wang, Q. Feng and J. Chai, "The Research of Consortium Blockchain Dynamic Consensus Based on Data Transaction Evaluation," in *Proc. 2018 11th International Symposium on Computational Intelligence and Design*, 2018, pp. 214-217.

[15] S. Zhang, and J. H. Lee, "A group signature and authentication scheme for blockchain-based mobile-edge computing," *IEEE Internet of Things Journal*, vol. 7, no. 5, 4557-4565, 2019.

[16] K. Chung and C. Keum, "Access control management of the cloud service platform," in *Proc. 2014 International Conference on Information and Communication Technology Convergence*, 2014, pp. 621-625.

[17] Q. Huang, X. Xia, Z. Xing, D. Lo and X. Wang, "API Method Recommendation without Worrying about the Task-API

Knowledge Gap," in *Proc. 2018 33rd IEEE/ACM International Conference on Automated Software Engineering*, 2018, pp. 293-304.

[18] P. Rodeghero, C. McMillan and A. Shirey, "API Usage in Descriptions of Source Code Functionality," in *Proc. 2017 IEEE/ACM 1st International Workshop on API Usage and Evolution*, 2017, pp. 3-6.

[19] X. Su, Y.M. Liu, and C. Choi, "A blockchain-based P2P transaction method and sensitive data encoding for E-commerce transactions," *IEEE Consumer Electronics Magazine*, vol. 9, no. 4, pp. 56–66, 2020.

[20] N. Kolokotronis, K. Limniotis, S. Shiaeles and R. Griffiths, "Secured by Blockchain: Safeguarding Internet of Things Devices," *IEEE Consumer Electronics Magazine*, vol. 8, no. 3, pp. 28-34, 2019.

Xin Su is a professor with the College of Internet of Things (IoT) Engineering, Hohai University, Changzhou Campus, 213022, China. His research interests include blockchain applications, 5/6G systems, edge/fog computing, and mobile ad hoc networks. Contact him at leosu8622@163.com.

Inam Ullah is currently a PhD student at the College of Internet of Things (IoT) Engineering, Hohai University, Changzhou Campus, China. His research interests include wireless sensor networks (WSNs), underwater communication, underwater localization, and robot localization. Contact him at inam.fragrance@gmail.com.

Meiling Wang is with the College of Internet of Things (IoT) Engineering, Hohai University, Changzhou Campus, China. Her research interests include blockchain applications, 5G systems, and edge/fog computing. Contact her at 2960661135@qq.com.

Chang Choi is currently an assistant professor at Gachon University, Republic of Korea. His research interests include intelligent information processing, semantic webs, smart IoT systems, and intelligent system security. He is the corresponding author of this article. Contact him at changchoi@gachon.ac.kr