Data Reliability Enhancement Method through Data Validation in Crowdsensing System

Mihui Kim*, Junehyeok Yun
Department of Computer Science & Engineering
Hankyong National University
Anseong, Republic of Korea
{mhkim, junhyeok2723}@hknu.ac.kr

***Abstract*— Crowdsensing is a new information system providing with sensing data collected by widespread mobile devices. Crowdsensing service provider could provide sensing data providers incentive to motivate the participation of data provision. However, some users could generate and provide fake data that different from real values to get incentives or maliciously. In this paper, we propose data reliability enhancement method through data schema based data validation and sensing client validation. Finally, we show the feasibility of proposing method with performance evaluation.**

Keywords—Crowdsensing; Data Validation; Reliability; Data Schema based

I. INTRODUCTION

Crowdsensing is an information system that collects data from various places using mobile devices such as smart phones and wearable devices used by the public, analyzes and processes the collected data, and provides it to the user again or directly[1]. Crowdsensing allows the public to participate in data collection tasks that are difficult for one person or a group to collect, making it easy to collect data without building a separate sensor system. Such a crowdsensing system is composed of a service provider and a user, and the user is again divided into a data requestor and a data provider. Crowdsensing cannot operate normally without the participation of data providers. Therefore, it is important to provide an incentive mechanism to provide reasonable compensation when providing effective sensing data in order to induce active involvement of data providers in crowdsensing system [2].

If the compensation type of the incentive mechanism is point or money that can be used as goods inside or outside the system, an attacker who generates false data that is different from the actual one for compensation purposes may appear. If the collected data is different from the actual data, the information generated by processing the data may also be inaccurate or ineffective. Therefore, data reliability in crowdsensing is very important for ensuring the trust of user in the system and operating the system normally. The crowdsensing service provider must be able to verify that the data contains valid content to ensure data reliability. However, since the data collected through the crowdsensing system has a wide collection range, it is difficult for the service provider to directly confirm the authenticity of the data. Therefore, there is a need for a method that can validate data without directly checking by the service provider.

In this paper, we propose a blocking method for false data generation and provisioning attacks using data rules and client validation. The data requestor establishes a data rule, which is a formalized format in which the data provided by the data provider must be satisfied. A data rule consists of one or more data fields. Each field has only one piece of information that cannot be separated and has constraints, including data types. By setting the data rules, you can block data whose values cannot be logically in each field. Service providers can quickly process data rules for large amounts of data by separating data rules and data into a MapReduce method [3].

Providing false data the attacker can manipulate the sensing client to automatically generate and provide false data in order to obtain maximum compensation in a short time. Client validation can be used to block attacks through these sensing client operations. The service provider stores the validation key in the sensing client code and distributes it. At this time, anti-reversing technology [4, 5] is utilized to prevent a third party from reversing the program code and taking information. When providing the sensing data, the client calculates the hash value together with the validation key on the data, and transmits the hash value together to the service provider. The service provider verifies the hash value in the same manner as the client and confirms that the data is generated in the valid sensing client.

In Chapter 2, we introduce the existing data validation technique, anti-reversing technology, and MapReduce technology as related studies. In Section 3, we introduce the crowd sensing system model and the false data attack model based on this paper. In Section 4, we propose a data validation scheme that can cope with the attack. In Section 5, we analyze the performance of the proposed technique and conclude in Section 6.

II. RELATED WORK

A. Existing Data Validation Techniques

Data validation mechanisms are divided into data-based verification and non-data-based verification. In data-based verification, a data clustering method [6] that classifies sensing data into groups based on the distribution of collected sensing data and assigns high validity scores to the group to which a large number of data belongs, and data types and constraints. A data schema-based verification method [7] in which data that

do not satisfy them are judged to be invalid data are included maintaining the integrity of the specifications. The data-based verification method can be expected to have higher accuracy than the data-based verification by validating the collected data itself. Data clustering techniques, however, are vulnerable when an attacker is deceived by multiple users (Sybil Attack [8]) or when many users collide to provide false data. In addition, an attacker can perform an effective attack by manipulating the sensing client to automatically provide false data.

The non-data element based verification includes the verification method of TPM (Trust Platform Module) based client validation [9], the verification based on user trust [10], the sensing time and the location information verification [11]. Non-data element based verification can estimate data validity, but it is difficult to confirm the validity of actual data. A user with high reliability can provide false data, and can manipulate middleware for inserting sensing time and position information, so that it can be deceived as valid data.

In this paper, we propose a method that can collect desired data with high reliability by using data validation schema validation method and client validation method based on non-data element. Schema validation techniques allow data requestors to accurately collect desired data by formatting data and setting constraints. Because each structure of data is used as a verification element, it is strong against Sybil attack or attack by many users. An attacker can manipulate the client to interpret the published schema and automatically generate and provide false data. To solve this problem, client validation is performed.

B. *Anti-reversing*

Reversing is an attack method in which program binaries are decompiled to obtain source code and duplicate information or operating mechanisms within the program through source code. Anti-reversing is a technique that prevents third parties from exploiting the source code of a program by using techniques such as code obfuscation [4] and anti-debugging [5]. If the program contains information that should not be disclosed to a third party, such as a cryptographic key or a unique algorithm that the program has, it may be possible for a third party to reverse the program code to take information Can be prevented. Code obfuscation is a way to make it difficult for a reversing attacker to interpret the source code by changing variable names and function names to arbitrary strings and arbitrarily modifying the placement of code blocks. Anti-debugging is a way to prevent a reversing attacker from obtaining source code by executing code that interrupts a program or interrupts debugging when the execution of the debugger is detected.

In this paper, we propose a method to prevent the attacker from taking the validation key and to perform the client validation securely by applying anti-reversing to the client validation module.

C. *System Model*

MapReduce is a framework developed for parallel distributed processing of big data [3]. The MapReduce framework consists of a mapper and a reducer. The mapper binds the input data into key-value form among the relevant data. The reducer removes the key-value data with the duplicated key from the key-value data generated by the mapper and extracts the desired data. The reducer may add or subtract the value of data having the same key value to remove duplicate key-value data. The number of data having the same key value may be utilized.

In this paper, we aim to improve the performance of the parallel rules by applying the mapping rule to the data rule satisfaction test. The mapper separates the data fields included in the sensing data and generates key-value data having the data as a key and the data rule satisfying each field as a value. If all the key-value data having the same data as the key is true, the reducer judges that the data satisfies the data rule.

III. PROPOSING MECHANISM

In this section, we propose a data reliability enhancement mechanism through data schema based data validation and sensing client validation.

A. *System Model*

Figure 1 shows the structure and processing flow of the crowdsensing system. The crowdsensing system consists of a service provider server and a user. The user is further divided into a data requestor (customer) and a data provider (provider). The service provider server acts as an intermediary between the data requester and the data provider. The service provider server broadcasts the request of the data requester to another user. (2) The data provided validates by the data provider, processes the data, and transmits the data to the data requester. (3) The service provider also performs the appropriate compensation management necessary for this process.

Crowdsensing can be used to collect data that is difficult to obtain due to initial sensor installation costs, such as health data from multiple people or environmental information (e.g., parking saturation, micro dust, humidity). However, since most of the general public are not experts in data collection, data reliability can be lower than direct sensor installation method, and false data can be generated and provided for compensation purposes. Therefore, the service provider must select valid data from the provided data, process it, and provide reliable information to the data requester.

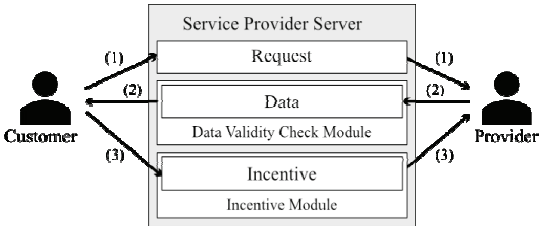

Fig. 1. Crowdsensing system structure.

In this paper, we propose a data validation method which combines data rule and client validation to increase data reliability in crowdsensing.

B. Attack Model

In [12] and [13], malicious users who provide false data for compensation purposes may appear in the crowdsensing system. These malicious users arbitrarily generate facts and other data to provide and compensate. At this time, it is possible to write a program that generates and provides random number data as soon as a data request is generated by monitoring the occurrence of a data request in order to obtain maximum compensation with minimum effort.

Figure 2 shows a flow diagram of an attack that uses a program to provide false data. (1) The attacker analyzes the communication contents between the service provider server and the sensing client, and confirms the request broadcasting format and the data providing format. (2) The attacker create a program that can detect data request occurrence and provide data based on the confirmed communication format. (3) The attack program monitors the service provider server and continuously checks whether a data request occurs. (4) If a data request occurs, the attacker generates false data by using random number, etc. and (5) he transmits it to the service provider server.

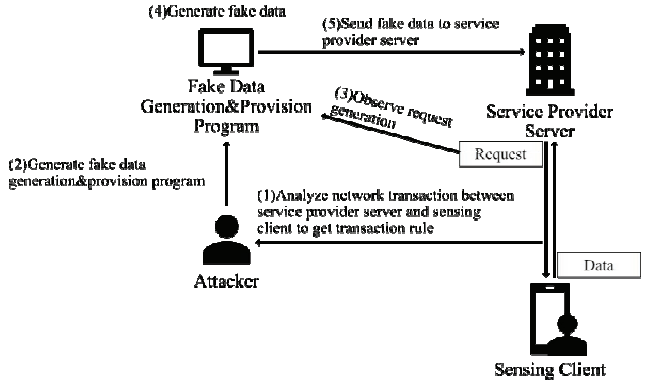

Fig. 2. False data attack using program

C. Proposing Mechanism

1) Data rule checking

Figure 3 shows a data validation scheme using data rules in a crowdsensing system. (1) The data requester (Customer) sets the data rule. Data rules are the types of data that you want to collect. (2) The data provider that receives the data request containing the data rule from the server collects the data, and (3) it processes it in the form conforming to the data rule set by the data requestor. (4) The service provider server checks whether the data satisfies the data rule. The data rule shown in Figure 4 consists of three fields: integer field between 1 and 100, character field with M or F value, and real field between 0.0 and 2.0. The data provided by the data provider satisfies the data type and requirements of each field and is therefore determined to be valid data. Data determined to be valid data is delivered to the data requester after appropriate processing, and rewards the data provider.

A data rule consists of a header and one or more data fields. The header includes the number of data fields and whether to allow duplication. Data fields include data types, availability, and constraints. Constraints have different types and meanings depending on the data type. Table 1 shows the format and semantics of constraints for each data type. By not disclosing data types and constraints, such as mine, blood type, and time, when they are clear, an attacker can use the program to make it difficult to generate false data.

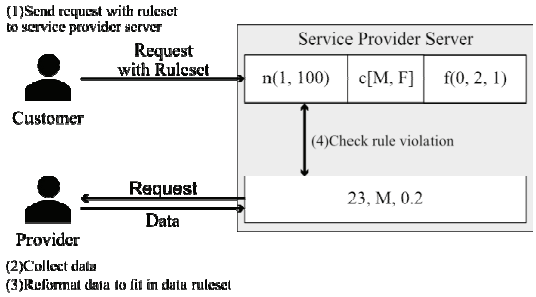

Fig. 3. Data rule-based data validation techniques

Figure 4 is a schematic diagram of a process flow for checking whether the data satisfies the data rule by applying the MapReduce method. (1) Input the data rule and data into the mapper. (2) Mapping and outputting each field of data rule and each field of data. (2-1) When the number of data fields and the number of data fields are different, it is judged as a rule violation and removed. (3) The mapping data output by the mapper is transmitted to the rule checking module. (4) Check whether the data satisfies the data types and constraints specified in the data rule. (4-1) If the data does not satisfy the data rule, it is determined that the rule violation is removed. (5) Pass the verified data to the data requester and (6) provide the data provider with compensation. By separating data rules and data from each other and processing them in a MapReduce manner, it is possible to quickly process data rules for a large amount of data.

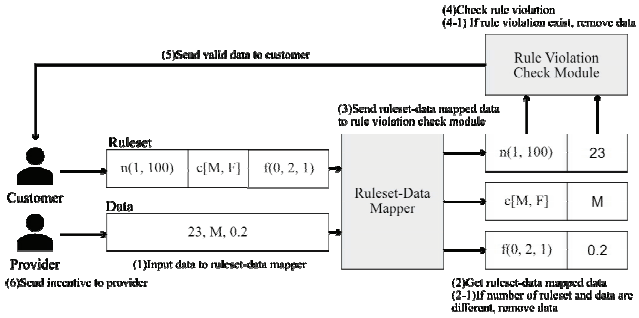

Fig. 4. Flowchart of data rule violation check.

TABLE I. CONSTRAINTS BY DATA TYPE

Data type	Restrictions
n(min,max)	Minimum, maximum values, empty if no limit
f(min,max,dig)	Minimum, maximum values, digits, empty if no limits
c(min,max)[characters]	(min,max): minimum, maximum lengths, empty if no limit, [characters]: values restrictions

2) Client checking

The attacker can automate the attack by manipulating the sensing client to interpret the data rules and automatically generate and provide false data. This automated attack can provide high attack efficiency because it can provide large amounts of false data in a short period of time. The service provider can check the operation of the sensing client and block the data not generated through the valid sensing client to block the automated attack.

Figure 5 shows the structure of the client validation module included in the sensing client. The client validation module consists of a validation key insertion submodule and a hashing submodule. The validation key is pre-inserted into the code by the service provider during the development of the sensing client. The validation key is stored as anti-reversing technology, so that even if the attacker takes the validation key through reversing, it cannot be used continuously. The service provider can determine that the corresponding data is generated by the valid client if the validation key value included in the data is the same as the key value inserted in advance. The validation key insertion submodule joins the data provided by the user with the validation key and transfers them to the hashing submodule. The hashing submodule hashes the data to which the validation key has been added and returns the hash value. The client validation module contains important information that should not be exploited by an attacker, such as a validation key or a hash function. Therefore, all code in the client validation module uses anti-reversing technology to make it difficult for the attacker to steal.

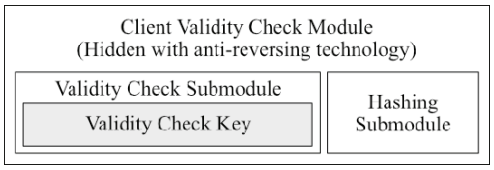

Fig. 5. Client validity cehck module

IV. Performance Evalutation

We perform performance evaluation simulation to show the performance of the proposed data validation technique. This simulation confirms the false data attack delay performance of the data rule checking method, the false data blocking performance of the data rule checking method, and the time required to validate the data rule based data.

In order to evaluate the performance of the proposed data validation method, we compared the health information of 1000 Koreans including height, weight, visual acuity, hearing, blood glucose level, and diabetes diagnosis [14] and data including temperature, humidity, The simulation was conducted using the meteorological observation information of Seoul in December [15]. Data rules applied to each data are shown in Tables 2 and 3. We used the Python built-in random number module to generate false data for the attacking model. All simulation results are the average of 10 simulations of the same conditions.

TABLE II. 1,000 Korean Health Information Data Rule

Fields	Comments
n(100,250)	Height integer between 100 and 200
n(30,150)	Weight integer between 30 and 150
f(0,2,1)	Vision value between 0.0 and 2.0, one significant digit
n(0,100)	Hearing ability integer between 0 and 100
n(0,400)	Blood sugar integer between 0 and 400
c(1)[t,f]	Charater (t or f) for diabetes diagnosis, true(t) or false(f)

TABLE III. Seoul Metropolitan City December weather information data rule

Fields	Comments
n(-20,20)	Temperature value between -20 and 20
n(0,100)	Humidity value between 0 and 100
c(1,2)[n,e,w,s]	Wind direction with characters n, e, w, or s of 1 or 2 length
n(0,80000)	Irradiation amount between 0 and 80000

A. Dealy Performance for Data Rule Checking

Figure 6 shows the false data attack delay performance of the data rule scheme. It is a graph showing the false data generation time according to the data rule application and the number of data rule fields. False data generation time is the time taken to generate 1000 false data. The false data generation time when the data rule is not applied is 0.001 second and the false data generation time when the data rule is applied is 0.018 seconds and 0.024 seconds respectively in the weather information and health information data. Data generation time is 10 times higher than false data generation time when data rule is not applied. These simulation results show that the application of data rules is effective in delaying false data providing attacks. If some of the data rule constraints are set to private, the attack delay effect can be expected to be higher because the attacker must write the false data code himself.

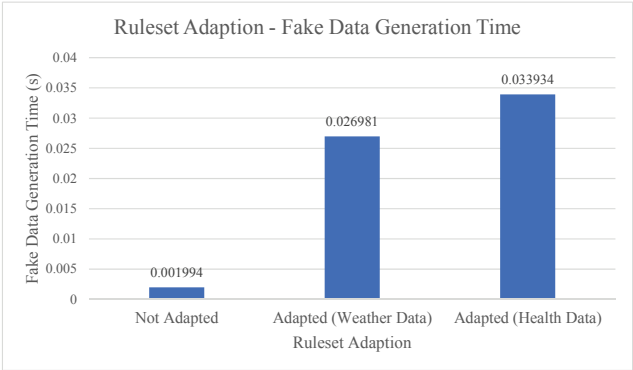

Fig. 6. Delay performance against false data attack with/without ruleset adaption

B. *False Data Blocking Performance for Data Rule Checking*

Figures 7 and 8 are graphs showing false data rate and effective data error rate when the data rule checking method is applied. The false data blocking rate is (the number of blocked data / the total number of false data), and the effective data false detection rate is (the number of blocked data / the total number of effective data). The simulation proceeds in a data pool that contains 1000 valid data and 1000 false data. The constraints of each field except the string field are assumed to be private. In both simulation groups, false data rates of more than 90% and false positives of less than 0.01% showed the possibility of data rule-based validation. When the number of data fields was large, higher false data rate and lower validity data were detected.

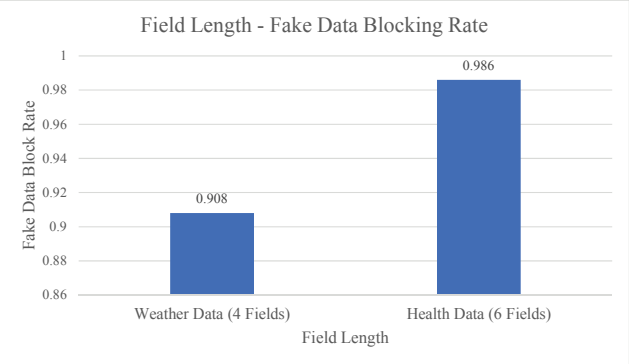

Fig. 7. Number of data fields vs. false data blocking rate

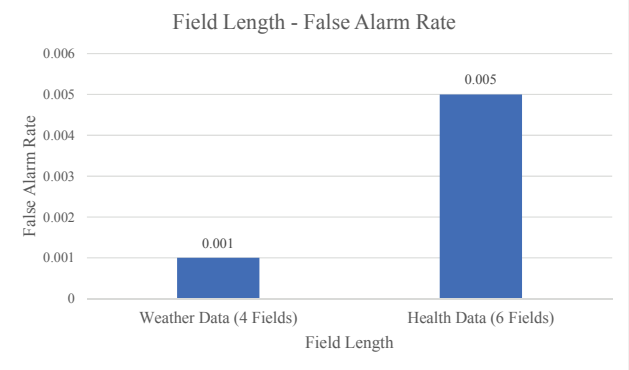

Fig. 8. Number of data fields vs. false positive alarm rate

C. *Validation Performance with MapReduce*

Figure 9 is a graph showing the time required for data rule checking depending on whether the MapReduce method is applied or not. When the number of data is small, overhead is generated in the process of dividing the data into fields for distributed processing and distributing the data to the distributed processing module for the distributed processing when applying the MapReduce, which takes more time than when the MapReduce is not applied. However, as the number of data increases, the data validation time of the system using MapReduce is lower than the data validation time of the system without MapReduce. The experimental results show that the data rule - based data validation using the MapReduce method in the crowdsensing system handling big data has high performance efficiency.

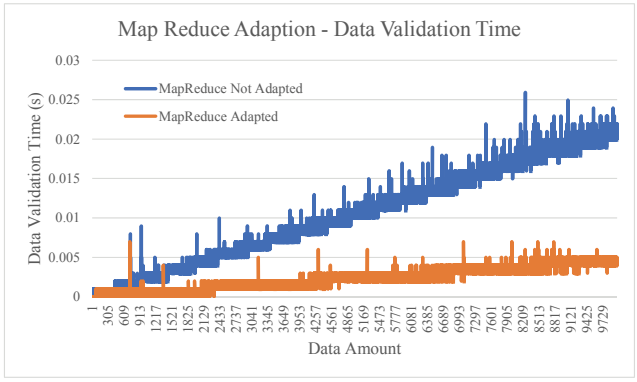

Fig. 9. Applying map redeuce vs. data validation time

V. Conclusions

In this paper, we proposed a method to improve the reliability of data in crowdsensing by performing data validation and client validation based on data rules. By solving the data validation based on the data rules desired by the data requestor, the validation accuracy problem of the existing non-data element based validation is solved. The performance evaluation of the proposed method shows that it has the effect of delaying and blocking automated false data attacks using the program.

Acknowleagement

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(grant number NRF-2018R1A2B6009620). Mihui Kim is a corresponding author.

References

[1] J. A. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Sasank and B. Mani, "Participatory sensing," May. 2006.

[2] X. Zhang, Z. Yang, W. Sun, Y. Liu, S. Tang, K. Xing and X. Mao, "Incentives for Mobile Crowdsensing: A Survey," IEEE Commun. Surv. Tutor., Vol.18, No.1, pp. 54–67, 2016.

[3] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," Commun. ACM, Vol.51, No.1, p.107, Jan. 2008.

[4] J. M. Borello and L. Mé, "Code obfuscation techniques for metamorphic viruses," J. Comput. Virol., Vol.4, No.3, pp. 211–220, Aug. 2008.

[5] M. N. Gagnon, S. Taylor, and A. K. Ghosh, "Software Protection through Anti-Debugging," IEEE Secur. Priv. Mag., Vol.5, No.3, pp. 82–84, May. 2007.

[6] T. Zhou, Z. Cai, K. Wu, Y. Chen, and M. Xu, "FIDC: A framework for improving data credibility in mobile crowdsensing," Comput. Netw., Vol.120, pp. 157–169, Jun. 2016.

[7] S. Xu, D. Su, and X. Wang, "The data validity evaluation in land change survey based on remote sensing," in Proceedings of 18th International Conference on Geoinformatics, Beijng, China, pp. 1–5, 2010.

[8] B. N. Levine, C. Shields, and N. B. Margolin, "A Survey of Solutions to the Sybil Attack," Technical Report of Univ. of Massachussets Amherst, 2006.

[9] P. Gilbert, J. Jung, K. Lee, H. Qin, D. Sharkey, A. Sheth and L. P. Cox, "YouProve: authenticity and fidelity in mobile sensing," in Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems - SenSys '11, Seattle, Washington, p. 176, 2011.

[10] X. Oscar Wang, W. Cheng, P. Mohapatra, and T. Abdelzaher, "ARTSense: Anonymous reputation and trust in participatory sensing," in Proceedings IEEE INFOCOM, Turin, Italy, pp. 2517–2525, 2013.

[11] K. N and A. Vs, "Sensor Data Modeling for Data Trustworthiness," in Proceedings of IEEE Trustcom/BigDataSE/ICESS, Sydney, Australia, pp. 909–916, 2017.

[12] R. Khorshidi and F. Shabaninia, "A new method for detection of fake data in measurements at smart grids state estimation," IET Sci. Meas. Technol., Vol.9, No.6, pp. 765–773, Sep. 2015.

[13] B. Wang, L. Kong, L. He, F. Wu, J. Yu, and G. Chen, "I(TS, CS): Detecting Faulty Location Data in Mobile Crowdsensing," in Proceedings of IEEE 38th International Conference on Distributed Computing Systems (ICDCS), Vienna, pp. 808–817, 2018.

[14] https://www.data.go.kr/dataset/15007122/fileDa ta. do, 2019.1.29

[15] http://data.seoul.go.kr/dataList/datasetView.do ?infId=OA-2225&srvType=S&serviceKind=1&currentPageNo=1, 2019.1.29