PAPER *Special Section on Information and Communication System Security*

Novel Method to Watermark Anonymized Data for Data Publishing

Yuichi NAKAMURA[†a)], **Yoshimichi NAKATSUKA**[†], ***Nonmembers***, ***and*** **Hiroaki NISHI**[†], ***Member***

SUMMARY In this study, an anonymization infrastructure for the secondary use of data is proposed. The proposed infrastructure can publish data that includes privacy information while preserving the privacy by using anonymization techniques. The infrastructure considers a situation where ill-motivated users redistribute the data without authorization. Therefore, we propose a watermarking method for anonymized data to solve this problem. The proposed method is implemented, and the proposed method's tolerance against attacks is evaluated.

key words: *anonymization, distortion attack, turbo code, watermarking*

1. Introduction

Linked Open Data is structured data that is openly published and can be interlinked with other data as a means to purposefully use the data. For instance, LinkData [1] provides a website where users can upload data as well as applications that use the uploaded data. Another example is a website where the United States publishes statistical data regarding companies [2] where users can access the data and then create and upload applications that use the data. This data usage is called secondary use.

Although these websites deal with public data, datasets such as activity history of customers and medical data are also valuable sources of secondary use of data. For example, the visitors' activity at a shopping mall may be effectively used for marketing purposes to increase the customers' shopping activities at the mall. However, despite the usability of the data, it must not be used as secondary data without addressing privacy protections since the data may contain privacy information.

Privacy-Preserving Data Publishing (PPDP) has been proposed [3] to solve this problem. It allows for the publishing of data while protecting the personal information included in the data. PPDP methods primarily consist of perturbation, homomorphic encryption, and anonymization of the data. Perturbation techniques add noise to the data values before the data is published [4], [5]. Users can use the data by estimating the original value while the users cannot know the true value.

A PPDP technique, categorized under homomorphic encryption, is useful for surrogating calculations of untrusted entities while protecting privacy, because it encrypts not only the network communication but also the contents of the calculations, such as the input data, the output data, and the calculation procedures [6], [7]. Although homomorphic encryption is well suited for data publishing for surrogating calculations, it is not suited for other purposes such as data sharing. On the other hand, the technique of anonymization abstracts values in the data in order to eliminate records that contains unique values by deleting items that directly express privacy information, such as a user ID [8], [9]. The objective of the abstraction is to eliminate data that may be regarded as private information when combined with other data. The abstraction technique attempts to protect private information from attacks that try to identify a record regarding a specific person. Metrics exist that express the privacy protection level of anonymization. For instance, the k-anonymity metric indicates how many records contain the same combination of values in the anonymized data by focusing on the abstracted items [10], [11].

Perturbation can be regarded as another method for anonymization. However, the perturbation is easier than anonymization in watermarking and less suitable than anonymization in some cases, e.g., in cases where users need true data values even if values are abstracted. Our previous work introduces an infrastructure for PPDP using anonymization [12] that consists of four agencies: a manager of the provided data, a manager of constraints regarding data sharing, a data publisher publishing anonymized data, and a manager of the previously published anonymized data. The infrastructure allows data users to acquire anonymized data by requesting requirements of data entries and privacy protection levels. Request constraints are defined by the data provider and are published with the data property by the infrastructure in advance. The infrastructure has a special anonymization procedure that refers previously published data. Such procedure prevents attacks that spoil the privacy protection by aggregating previously published data.

However, the proposed infrastructure in [12] still has a problem when it comes to the illegal republishing of data. The infrastructure provides the same data to some data users when the users request the same requirements to the infrastructure, and some of the users may republish the anonymized data without the permission of the data providers. In this case, the illegal data user should be identified when the illegally published data is accessed. This will act as a deterrent to illegal republication. To identify illegal data republication, anonymized data should provide the information pertaining the official user of the data. Therefore,

Manuscript received September 8, 2016.
Manuscript revised February 10, 2017.
Manuscript publicized May 18, 2017.

[†]The authors are with Keio University, Yokohama-shi, 223–8522 Japan.
a) E-mail: nakamura@west.sd.keio.ac.jp

DOI: 10.1587/transinf.2016ICP0015

published data should be unique regarding its data users. Watermarking is well suited for this purpose. Watermarking adds a watermark into the data by modifying the values of the data to identify the users who published the data. In the watermarking research domain, multimedia data such as audio data, picture data, video data, and text data are popular targets for watermarking [13], [14]. Despite the availability of watermarking, the conventional watermarking methods are not appropriate for use with anonymized data because they change the anonymization level and degrades the usefulness of the data.

Moreover, the type of the anonymized data we are addressing is different from multimedia data. Multimedia data allows us to watermark information sequentially, because the data comprises ordered information, such as the elapsed time of the audio or video data. In addition, multimedia data does not negatively influence the quality of the data when modified, given that multimedia data has data redundancy. Data such as audio data has high-frequency components that humans cannot hear. In contrast to multimedia data, records in anonymized data do not have a sequence, and their values do not have suitable positions for modification by watermarking.

Another issue is the revealing of watermarks, which is a common problem faced by all watermarking techniques. Revealing may occur when several anonymized datasets are generated and published from the same original data. Watermarking for anonymized data has to be tolerant of a modification that reveals the watermarks while simultaneously preventing the degradation of the anonymization level.

The contribution of this study is the proposal of novel watermarking method for anonymized data. As the specification of the proposed method, it is tolerant against out-of-order data sets and degradation of the anonymization level. We also evaluate the proposed method to assess its robustness against attacks.

2. Related Techniques

The proposed watermarking method utilizes several error correction techniques in conjunction with cryptography. This section describes the combined techniques.

2.1 Advanced Encryption Standard

The Advanced Encryption Standard (AES) is a symmetric-key encryption method, and is a type of block cipher. Symmetric key encryption uses one secret key for both the encryption and decryption processes. The block cipher encrypts plaintext at intervals of several bits. Each interval is called a block; the general block length of AES is 128 bits. The length of a ciphertext is a multiple number of the block length.

AES outputs ciphertext that is generated from multiple blocks when the length of the plaintext is longer than the block length. The blocks should not be isolated by the encryption process in order to prevent attacks that replace parts of the ciphertext to modify its plaintext. In addition, the key for encryption should be changed with every block encrypted; otherwise attackers would be able to read the plaintext from a ciphertext without decryption. In other words, a block of ciphertext will show the same block of plaintext when the same key is used for the encryption. Several ways to connect blocks have been proposed to solve these problems. Cipher Block Chaining (CBC) mode and CounTeR (CTR) mode are recommended by Ferguson et al. [15]. One of the differences between CBC mode and CTR mode is the influence on neighboring blocks when a part of ciphertext is flipped from 0 to 1 and vice versa (bit flipping). In CBC mode, when a bit in a block gets flipped, it affects the block and its two neighboring blocks which prevents them to provide the correct decryption result. In contrast, bit flipping in CTR mode only affects a single block.

Cryptography techniques should not give attackers information that facilitates their attacks, such as information that comes from the imbalance of the bits in the ciphertexts. Therefore, bits in ciphertexts should be well balanced. According to Matsuoka et al., the number of bits that express '1' is 50% in the ciphertext of AES [16]. The study shows that bits in the ciphertext of AES are well balanced.

2.2 Turbo Code

Error-correction code enables one to recover the correct bit string, even when the bit string contains flipped bits, by adding a string of parity bits into the original bit string. Turbo code is one of the error-correction codes whose transmission efficiency is regarded to be among the most efficient in the data transmission domain. In other words, turbo code requires shorter bit lengths to encode a bit string in order to achieve the same error-correction ability as other error-correction code methods. In the encoding process, the turbo code first interleaves the input data string, generating two parity bit strings from the original data string and the interleaved string using an internal encoder. Next, the generated strings are decimated alternately in order to reduce the total length of the two strings by half (the puncturing process). Although the puncturing process is irreversible, it is effective in reducing the length of a bit string that results from the encoding. Finally, the punctured parity bit string is connected to the original bit string.

In the decoding process, the turbo code attempts to obtain the decoded result by repeatedly running an internal decoder. The Soft Output Viterbi Algorithm (SOVA) [17] is one such process algorithm. In this algorithm, a data string and a parity bit sting are used for each running process. In addition, a decoded result outputted from the previous running process is also inputted into the next running process. Prior to the decoding process, an encoded bit string is divided into a data string and a parity bit string. The two parity bit strings that were punctured during the encoding process are recovered from the parity bit string of the encoded bit string by using the mean values of '0' and '1' instead of the bits omitted during the puncturing process. For each running process of the internal decoder, non-interleaved and interleaved data strings are used alternatively. The previous decoded result is also interleaved or de-interleaved before the next running process using the data string. The recovered parity bit string is selected according to the condition whether the data string is interleaved or not. The final decoding result will be completed by the iteration of the internal decoding.

2.3 Gray Code Converter

Gray code converts data values to make all Hamming distances of neighboring values equal to 1. Generally, when the Hamming distance is small, a value that includes a flipped bit expressing a similar value to the original value. Therefore, using gray code with error-correction code enhances the error-correction ability of the code.

3. Proposed Watermarking Method

3.1 Features of the Proposed Method

In this study, we assume that the proposed watermarking method is used to identify the person who republished the anonymized data without authorization. The watermarking method should allow only authorized people to access the watermarked information (access control). Also, watermarked information should be able to be extracted even if the anonymized data containing that watermarked information is modified (robustness). Moreover, the proposed watermarking method should include protection from attacks where multiple malicious data users attempt to extract watermarked information by comparing the anonymized data received by each of the data users (a collusion attack).

The proposed watermarking method uses AES for access control. Watermarked information is encrypted by AES before being watermarked. Access control can be achieved because only the people who are authorized to access the watermarked information know the secret key of the encryption.

Use of AES is also effective to protect against collusion attacks. Attackers in collusion attacks attempt to extract watermarked information by comparing multiple instances of the published anonymized data. Even when the proposed watermarking method expresses the watermarked information in the form of binary data, when the values of the anonymized data are modified, attackers cannot identify whether the differences express ‘0’ or ‘1’ from a statistical point of view if the probability of occurrence of ‘0’ and ‘1’ in the binary data is the same. As we mentioned in Sect. 2.1, the probability is the same in AES ciphertexts. Therefore, the proposed watermarking method can protect watermarked information from collusion attacks.

The proposed watermarking method should also consider another type of attack whereby attackers try to delete or degrade the watermarked information by changing the values of published data (a distortion attack). Distortion attacks can be treated as noise addition to the data. In this paper, we use turbo code to protect watermarks from distortion attacks, since the length of the parity bit string that the code requires is shorter than other error-correction code techniques. Given that the degree of modification for watermarking is proportional to the length of the watermarked bit string, using turbo code is one of the best approaches to minimize degradation of the value of the anonymized data resulting from watermarking. The proposed watermarking method encodes information that will be watermarked into anonymized data after AES encryption. In addition, we convert the encoded information using a gray code converter in order to enhance the error-correction ability of turbo code.

3.2 Entire Flow of the Proposed Method

Figure 1 illustrates the entire flow of the proposed watermarking method. The upper side and the bottom side of the figure show the watermarking and extracting flows, respectively. In the watermarking flow, a plaintext is first encrypted to a ciphertext by AES with the block length of the encryption being 128 bits. Next, the ciphertext is encoded by the turbo code. The encoding adds a parity bit string whose length is same as the ciphertext. Finally, the encoded bit string is watermarked to the anonymized data by modifying the values of that data. The encoded bit string is converted into gray code just before the modification.

In the extracting flow, the proposed watermarking method first extracts a bit string from the anonymized data that has been republished without authorization. The extraction involves the inverse conversion of the gray code. Next, the extracted bit string is decoded by the turbo code, and finally the decoded string is decrypted by AES. In the proposed watermarking method, CTR mode is selected for use by AES since the influence of distortion attacks is small.

We implemented a systematic convolutional code as the internal encoder of the turbo code. The code holds a current state and decides the next state according to an input bit and the current state. A parity bit is output when the state changes. Figure 2 shows a trellis diagram of the implemented code. The initial state is S1. When a bit string ‘011’ is input, ‘010’ is output as a parity bit string while its state changes from S1 to S2 via S3. The implemented turbo code used SOVA for the decoding process.

Figure 3 illustrates the interleaving process that we implemented for the turbo code. The interleaver divides an input bit string into parts at 3-bit intervals. The parts of the bit string are arranged in row directions. An interleaved bit string is created by joining the arranged parts along the column direction. The reverse process of the interleaving was also implemented as the de-interleaver.

3.3 Watermarking Process of the Proposed Method

In the proposed watermarking method, the process that watermarks the information into anonymized data consists of

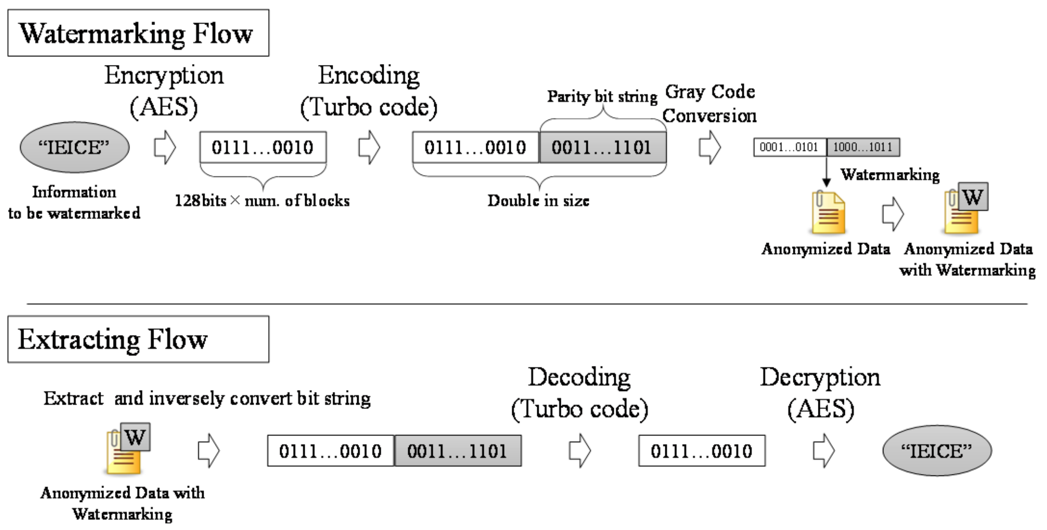

Fig. 1 Entire flow of the proposed watermarking method

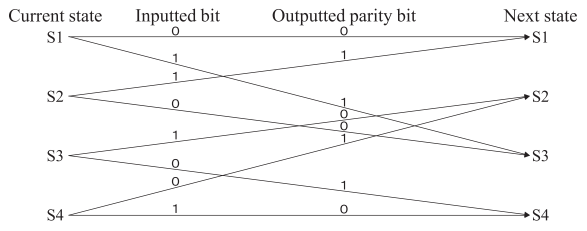

Fig. 2 Trellis diagram of systematic convolutional code

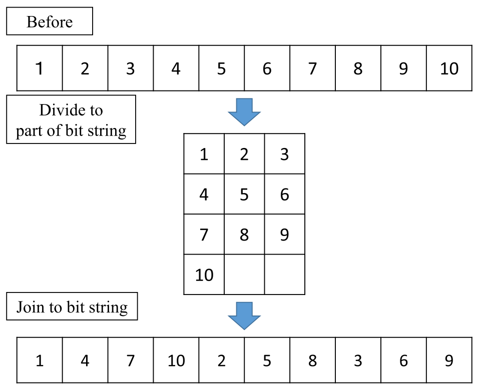

Fig. 3 Interleaving process of implemented interleaver

Table 1 Example data table after grouping process

Group	Time slot	Area of room	Congestion
G1	8:00-8:30	East	Empty
G2	8:30-9:00	South	Full
	8:30-9:00	North	Medium
	8:30-9:00	East	Full
	8:30-9:00	West	Empty
	8:30-9:00	North	Medium
	8:30-9:00	South	Medium
	8:30-9:00	East	Full

Table 2 Example data table after modification process

Group	Time slot	Area of room	Congestion
G1	8:00-8:30	South or East	Empty
G2	8:30-9:00	South	Full
	8:30-9:00	North	Medium
	8:30-9:00	East	Full
	8:30-9:00	West	Empty
	8:30-9:00	North	Medium
	8:30-9:00	South or East	Medium
	8:30-9:00	South or East	Full

two sub-processes: a grouping process and a modification process of anonymized values. The grouping process first sorts records of anonymized data and then makes record groups so that each group includes all records that have the same values of the sorted items. The grouping process was implemented in order to watermark information sequentially, otherwise the sequential order for the watermarking would be broken by changing the order of the records, since anonymized data includes records that have the same values for privacy protection.

In the modification process, values are modified to express the watermarked information. The number of modified records in each record group expresses the watermarked information. Each record group expresses a part of the information when the information is larger than the number of records in the record group because the bit length of the part is as large as the largest the record group can express.

Table 1 illustrates an example of an anonymized data table after the grouping process. The example table shows the degree of congestion of rooms classified as north, south, east, and west. In the table, records are sorted by "Time Slot" item, and then divided into two groups, G1 and G2. Table 2 is an example of an anonymized data table after the modification process when the watermarked information is '1011' in binary format. In this example, "Area" is set to a target item of the modification and the modified values are underlined. In Table 2, G1 expresses the first bit from the left side of the watermarked information since the number of records in G1 is one. The value of the left most bit is '1'. One record in G1 has been modified, because the value '1' converted by the gray code is '1'. G2 can be expressed as any value from 0 to 7 by the watermarking method, since it has 7 records. Therefore, G2 supports 3-bit bit strings. A part of the information that has not yet been watermarked is '011' where the converted value of '011' is '010'. Since the decimal format of '010' is '2', two of the records in G2 have been modified. The modification process degrades the value of the data due to the abstraction of the watermarking process. The modification process does not modify records repeatedly in order to minimize the degradation.

In this study, we implemented four abstraction methods to modify the values of the anonymized data while preventing any additional leaks of privacy information. The first method is named "masking method" which replaces a part of a value to a wildcard character '*'. The second method is named "extension method" which adds candidates of a part of a value. For instance, "Hoge city" can be modified to "Hoge or Foo city" by the extension method. The third method is named "replacing method" which replaces a part of a value to a candidate of the value. The last method is named "arranging method" which changes the order of a part of a value that has multiple candidates. Watermarked information can be extracted from watermarked anonymized data by comparing it with non-watermarked anonymized data, which is called the comparison process. In this process, the number of anonymized records used in the watermarking process are counted in each record group. The sequence of the number denotes the watermarked bit string that is encrypted by AES, coded by turbo code, and converted by gray code. The extraction flow described in Sect. 3.2 and Fig. 1 shows how the plaintext is extracted from the watermarked bit string.

The non-watermarked anonymized data is not published to the data users, making it impossible for the attackers to compare watermarked and non-watermarked anonymized data for watermark extraction. Therefore, the only way for the attackers to extract watermarked information is to compare multiple sets of published anonymized data. This is called collusion attack as mentioned in Sect. 3.1. The proposed watermarking method protects the watermarked information from collusion attacks by using the characteristic of the AES encryption as described in Sect. 3.1.

4. Specification of the Implemented Turbo Code

4.1 Success Rate of Error Correction

Before evaluating the proposed watermarking method, we measured the error correction ability of the implemented turbo code while shifting the length of a bit string input to the turbo code from 128 bits to 384 bits at 128-bit intervals. The input bit strings were the ciphertext of AES using the CTR mode, which were created by randomly generated plaintexts, secret keys, and nonces. Figure 4 illustrates the success rate of error corrections when some bits of input bit strings are flipped before the decoding process of the turbo code. The flipping rate against the input bit string was set from 1/256 to 1 while the decoding process was executed twenty times for each flipping rate. The internal decoder ran up to one thousand times for the each decoding process. The iteration of the internal decoding process was a bottleneck in terms of the decoding processing time. To reduce the influence of the bottleneck, the iteration was aborted when the internal decoder output the same result for four times in a row.

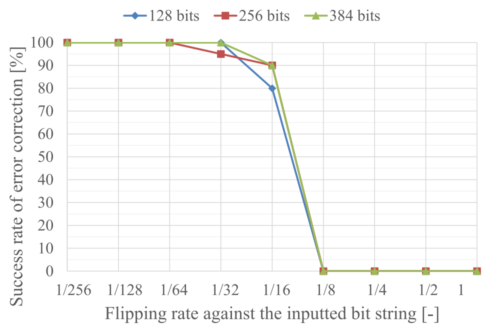

Fig. 4 Success rate of error correction (1)

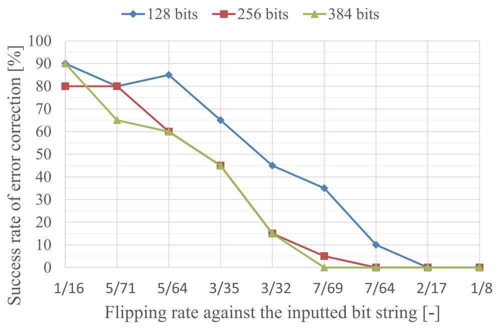

Fig. 5 Success rate of error correction (2)

Figure 4 shows that the success rate is greater than 80% when the flipping rate was 1/16 or smaller, and 0% when the flipping rate was 1/8 or larger. Figure 5 shows another result of the measurement with the same conditions as Fig. 4 except for the flipping rate of the input bit string. The flipping rate was set from 1/16 to 1/8 in order to clarify the fluctuation of the success rate using these flipping rates. In Fig. 5, the success rate fluctuated linearly. In addition, for most of the flipping rates, the success rate was small compared to others where the input bit strings were longer than in this case. This is because SOVA calculates the most feasible path of state transition of the internal encoder at every 1 bit of the input bit string when decimating the paths in order to reduce the calculation cost of the decoding process. Therefore, the total number of the decimated paths is comparatively small when the length of the input bit string is

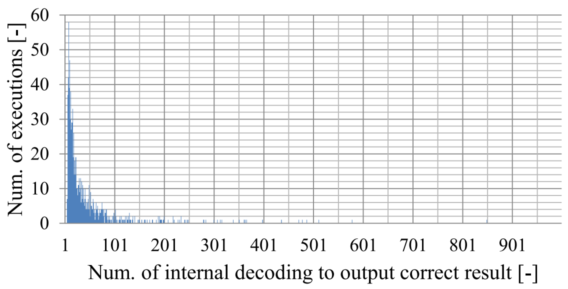

Fig. 6 Distribution of number of decoding (128 bits)

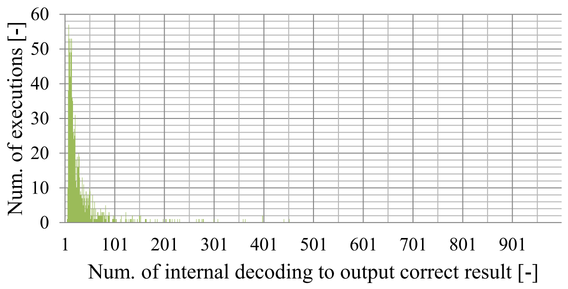

Fig. 7 Distribution of number of decoding (256 bits)

short.

4.2 Validity of the Maximum Limit of Iteration

We measured the required number of iterations of the internal decoding to output the correct decoding result when the length of the input bit string is 128, 256, and 384 bits in order to evaluate the validity of the maximum limit of iteration of the internal decoding process. The results are shown in Figs. 6, 7, and 8, respectively. The number of flipped bits was set to 23, 42, and 64 bits, respectively, such that the success rate of the decoding process is 50% for each of the conditions. The numbers of flipped bits measured are displayed in Fig. 5. For each of the three figures, the horizontal axis indicates the number of iterations of the internal decoding executed until the result of the internal decoding became stable while the vertical axis indicates the number of the decoding processes. The decoding process was executed two thousand times for each of the three measurements. Executions that failed to decode were excluded since the evaluations focused on the influence of the maximum limit to the success rate. Executions that were unstable at the maximum limit of the iteration were also excluded as outliers. The averages of the number of the iterations required were 44.41, 33.88, and 61.31, respectively.

We determined from the three measurement results that most of the decoding processes became stable within four hundred iterations, and the required number of iterations to output the correct decoding result decreased exponentially when increasing the values of the vertical axis. Therefore, the maximum limit of the number of iterations is reasonable when the limit is one thousand.

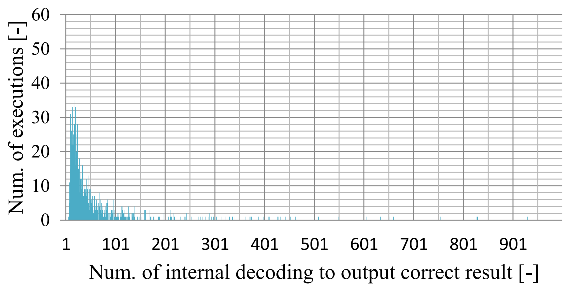

Fig. 8 Distribution of number of decoding (384 bits)

Table 3 Rates of incorrect aborting

Aborting condition	Flipping rate of bits			
	1/16 +5/128	1/16 +6/128	1/16 +7/128	1/8
Four times	5%	0%	15%	0%
Six times	0%	0%	10%	0%
Eight times	0%	0%	5%	0%

4.3 Validity of Aborting Iterations of Internal Decoding

The implemented decoding process aborts the iteration of the internal decoding process when the internal decoder outputs the same result four times in a row. We measured the rate where the decoded result of the aborted decoding process is different compared to the case when the internal decoding process is iterated one thousand times. The rates were measured while shifting the flipping rate of bits of the input bit string from 1/16 to 1/8. Table 3 shows a part of the result of the measurement when the aborting condition is four, six, and eight times. The rates of all the aborting conditions were 0% when the flipping rate of bits was 1/16 + 4/128 or lower. According to the Table 3, the rate is small when the aborting condition is large. When the decoded result at abortion was different from the result after thousand iterations, both results were incorrect. Therefore, we determined that the aborting condition of the implemented decoding process does not degrade the decoding process.

5. Evaluation

In this study, we implemented the proposed watermarking method, executed three distortion attacks using Python 3.5, and evaluated the proposed watermarking method tolerance to the distortion attacks. The distortion attacks are a deleting attack, an adding attack, and a replacing attack that respectively delete, add, or replace the records of the data. These are common types of distortion attacks in the watermarking domain.

5.1 Tolerance against Distortion Attacks

We evaluated the tolerance to the distortion attacks of

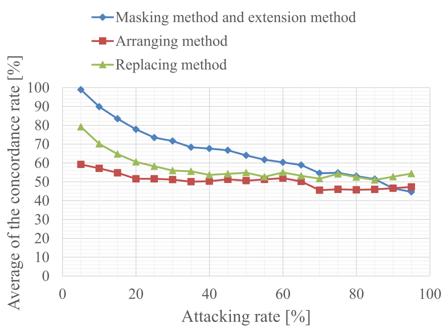

Fig. 9 Tolerance against deleting attack

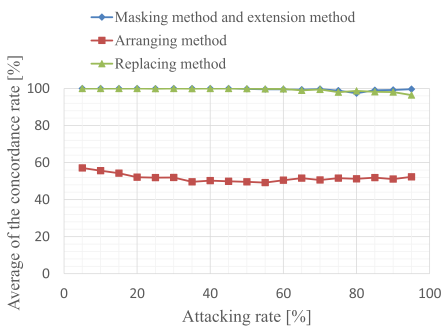

Fig. 10 Tolerance against adding attack

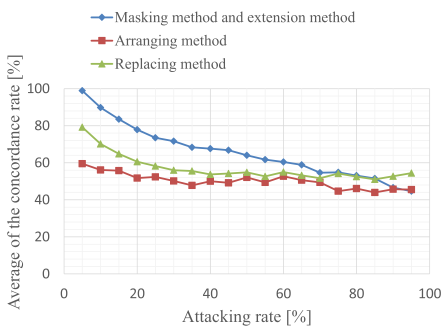

Fig. 11 Tolerance against replacing attack

the proposed watermarking method by attempting to extract correct watermarked information from the anonymized data modified by the distortion attacks. Measurements for the evaluation were executed for each abstraction method. Anonymized data for the evaluations was generated from the location data of bus stops published at linkdata.org [1]. The location data of a bus stop itself does not contain privacy information. However, the data has to be anonymized if the data also indicates where passengers embarked and disembarked. Therefore, although the location data with information of passengers is valuable for secondary use such as for the marketing of shops along the bus line, attackers may be able to identify personal information of a specific passenger from the data. We set the number of records in the anonymized data to 3,000, which is the average number of passengers that ride on one bus in one month [18].

We measured the concordance rate of a watermarked bit string extracted from attacked anonymized data when the bit string is compared with the correct bit string. Figures 9, 10, and 11 display the measurement results of the tolerance to the deleting, adding, and replacing attacks, respectively. The vertical axes indicate the average of the concordance rate while the measurement was executed twenty times for each attacking rate shown in the horizontal axes. The attacking rate expresses the rate of the number of records that are deleted, added, or replaced by the distortion attacks compared with the original number of records in the data. The attacking rate was set from 5% to 95% at 5% intervals. Results of the masking method and the extension method were the same since their comparison algorithms at the extraction process of the watermarked information are the same. The watermarked information was 256 bits long.

Figure 9 shows that the most tolerant methods against the deleting attack are the masking method and the extension method when the attacking rate is 85% or smaller. Figure 10 shows that all abstraction methods except the arranging method keep the concordance rate to 96% or larger against the adding attack. Figure 11 shows similar measurement results as in Fig. 9 because their algorithms of attack are similar. We can regard the replacing attack as a combination of the deleting attack and the adding attack with the same attack rates, and the influence of the adding attack against this similarity is comparatively small since the proposed watermarking method is tolerant against the addition attack except when using the arranging method.

When the attacking rate is high, the extracted bit string consists of a large number of ‘0’s since the proposed watermarking method expresses watermarked information by the number of modified records. In addition, half of the ciphertext of AES consists of ‘0’s. Therefore, the concordance rates in the three measurement results come close to 50%. The three measurement results show that the arranging method is not tolerant to distortion attacks since its concordance rate is 50%. However, this weakness can be an advantage when the arranging method is used to detect data modification.

5.2 Availability of the Proposed Watermarking Method in the Proposed Infrastructure

We can assume similar situations can exist where attackers use the same kind of distortion attacks in the infrastructure for PPDP that we have proposed in [12]. The adding attack can be regarded as a situation where multiple anonymized data tables are gathered into one data table. According to the measurement results, the proposed watermarking method can extract at least 96% of the watermarked information even if the anonymized data is combined with other data whose size is 95% of the anonymized data. The deleting attack can be regarded as a part of the anonymized data being republished. In the deleting attack, the concordance rates of the masking method and the extension attack are 50% when the attacking rate is 70% or higher. Therefore, the proposed watermarking method is effective in identifying whether the watermarking information is embedded or not if the republished data includes at least 30% of the original anonymized data. This tolerance is sufficient because anonymized data has no value when 70% of its records are lost.

6. Conclusion

We proposed and implemented a watermarking method for anonymized data while assuming the proposed method is used to identify the user that republished the anonymized data without authorization within an infrastructure to publish anonymized data. The proposal facilitates secondary use of data that may include privacy information by providing an approach to generate identifiable anonymized data. The proposed watermarking method consists of AES, turbo code, and a gray code converter to protect watermarking information from collusion attacks and distortion attacks. Experimental results showed that the proposed watermarking method can extract more than 95% of the watermarked information from anonymized data even if the data is combined with other records whose size is 95% of the data. In addition, the embedding of watermarked information can be detected from data that is 30% of the anonymized data.

Acknowledgments

This work was partially supported by MEXT/JSPS KAKENHI Grant (B) Number JP16H04455, through the funding received from SECOM Science and Technology Foundation, from MLIT Grant for development of advanced technology in housing and buildings, by R&D project "Design of Information and Communication Platform for Future Smart Community Services" by the Ministry of Internal Affairs and Communications of Japan, from Keio Univ. Global Smart Society Creation Project Research, and from Keio Univ. Doctorate Student Grant-in-Aid Program.

References

[1] LinkData, "Link and Publish your data | Open data sharing," http://linkdata.org, accessed Aug. 2016.
[2] The U.S. General Services Administration, "Data.gov," http://www.data.gov/, accessed Aug. 2016.
[3] B.C.M. Fung, K. Wang, R. Chen, and P.S. Yu, "Privacy-preserving data publishing: A survey of recent developments," ACM Computing Surveys (CSUR), vol.42, no.4, pp.1–53, 2010.
[4] M.B. Malik, M.A. Ghazi, and R. Ali, "Privacy Preserving Data Mining Techniques: Current Scenario and Future Prospects," Third International Conference on Computer and Communication Technology (ICCCT), pp.26–32, 2012.
[5] T. Jahan, G. Narasimha, and C.V. Guru Rao, "A Com-parative Study of Data Perturbation Using Fuzzy Logic to Preserve Privacy," Networks and Communications (Net-Com2013), the series Lecture Notes in Elec-trical Engineering, Springer International Publishing, vol.284, pp.161–170, 2014.
[6] C. Gentry, "Fully homomorphic encryption using ideal lattices," In STOC, pp.169–178, 2009.
[7] W. Wang, Y. Hu, L. Chen, X. Huang, and B. Sunar, "Exploring the Feasibility of Fully Homomorphic Encryption," IEEE Trans. Comput., vol.64, no.3, pp.698–706, 2015.
[8] A. Asayesh, M.A. Hadavi, and R. Jalili, "(t,k)-Hypergraph anonymization: an approach for secure data publishing," Journal of Security and Communication Networks, vol.8, no.7, pp.1306–1317, 2015.
[9] K. Shared and G. Danezis, "An Automated Social Graph Deanonymization Technique," Proceedings of the 13th Workshop on Privacy in the Electronic Society (WPES), pp.47–58, 2014.
[10] L. Sweeney, "k-anonymity: a model for protecting privacy," International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, vol.10, no.5, pp.557–570, 2002.
[11] L. Sweeney, "Achieving k-anonymity privacy protection using generalization and suppression," International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, vol.10, no.5, pp.571–588, 2002.
[12] Y. Nakamura, K. Matsui, and H. Nishi, "Anonymization Infrastructure for Secondary Use of Data," Proceedings of the International Conference on Internet Computing (ICOMP), SESSION 9-ICOMP, pp.65–71, 2014.
[13] C.I. Podilchuk and E.J. Delp, "Digital watermarking: algorithms and applications," IEEE Signal Process. Mag., vol.18, no.4, pp.33–46, 2001.
[14] P. Singh and R.S. Chadha, "A survey of digital watermarking techniques, applications and attacks," International Journal of Engineering and Innovative Technology (IJEIT), vol.2, no.9, pp.165–175, 2013.
[15] N. Ferguson and B. Schneier, "Practical Cryptography," Wiley, 2003.
[16] H. Matsuoka, K. Inoue, and H. Nishi, "Perfect Classified Channel retaining DC balance," IEICE technical report, vol.108, no.15, DC2008-1, pp.1–6, 2008 (in Japanese).
[17] J. Hagenauer and P. Hoeher, "A Viterbi algorithm with soft-decision outputs and its applications," Global Telecommunications Conference and Exhibition 'Communications Technology for the 1990s and Beyond' (GLOBECOM), vol.3, pp.1680–1686, 1989.
[18] Ministry of Land, Infrastructure, Transport and Tourism, "Statistical Survey of Automobile Transportation," http://www.mlit.go.jp/k-toukei/06/annual/index.pdf, 2015, accessed Aug. 2016 (in Japanese).

Yuichi Nakamura received his M.S. degree from Keio University, Japan, in 2016. He has been a research associate at Keio University, Japan, from 2016 to the present. He is currently a doctoral course student in the Center of Information and Computer Science, Keio University, Japan.

Yoshimichi Nakatsuka is a bachelor degree student at the Department of System Design, Faculty of Science and Technology, Keio University, Japan, from 2013 to the present. The main theme of his research is network security in information centric networking.

Hiroaki Nishi received his Ph.D. degree from Keio University, Japan, in 1999. He is a professor at Keio University, and has been a visiting professor at the National Institute of Informatics (NII), Japan, since 2014. The main theme of his research is in building total network systems, including the development of hardware and software in the field of next-generation IP router architecture.