Journal of Ambient Intelligence and Humanized Computing (2024) 15:1379–1386
https://doi.org/10.1007/s12652-018-0871-7

ORIGINAL RESEARCH

Guaranteeing the integrity and reliability of distributed personal information access records

ChaeHo Cho[1] · Manki Baek[1] · Yoojae Won[1]

Received: 18 September 2017 / Accepted: 21 May 2018 / Published online: 1 June 2018

Abstract
Attackers try to forge or delete personal information access records to hide traces of their attacks. As personal information access records can be used to analyze infringement accidents or as legal evidence in the event of malicious attacks, maintaining their integrity is very important. This article presents measures to efficiently prove the integrity of distributed personal information access records. To construct a reliable log system, diversified security requirements are established, and mechanisms such as a hash chain, message authentication code, and Merkle tree are incorporated. Moreover, as integrity is proved through a third-party verification institution, attacks by external as well as internal attackers can be detected. During the validation process, existing log record protection methods fail to detect forgery or deletion of certain data or have difficulty identifying the time of attack, but such drawbacks are addressed by the proposed integrity verification process, with only a minor increase in computational load.

Keywords Integrity · Hash chain · Message authentication code · Third-party verification institution · Merkle tree

1 Introduction

Personal information refers to any information that can be or has been used to identify individuals (OECD Council 1980). If personal information is leaked, individuals may be vulnerable to personal and financial damage and various crimes such as identity theft, and companies and countries may suffer negative publicity, reduced reliability, brand value loss, and liabilities arising from potential lawsuits. In an information technology (IT) environment, information systems that collect, process, and store personal information must be stringently protected to prevent the leakage and exposure of personal information and the infringement of personal privacy caused by internal and external attacks. Policies and systems for quick analysis and response measures in the event of accidents must be designed. The National Institute of Standards Technology advises organizations to establish a system audit trail environment and specify procedures and responsibilities to maintain personal information securely within all systems and processes that create and use personal information (Swanson and Guttman 1996). Such security control measures provide the foundation for ensuring the security of the critical information assets and personal information of organizations. In Korea, the Personal Information Protection Act and related laws require information systems to collect and manage the history of access and use of personal information (Ministry of the Interior 2017; Korea Communications Commission 2012).

The generation and management of personal information access records are enforced by specific laws so that audit trails can be analyzed to determine the reasons for infringement accidents such as illegal viewing, misuse, and abuse of personal information, and measures can be developed to deal with such incidents in the future. Personal information access records can not only be generated in forms required by laws but also included in logs generated automatically by various interconnected information systems. These include security systems such as firewalls, operating systems, and database management systems. These logs are stored separately in different forms from the personal information access records specified by legal regulations. Consequently,

✉ Yoojae Won
yjwon@cnu.ac.kr

ChaeHo Cho
greatopen@cnu.ac.kr

Manki Baek
bmg8551@cnu.ac.kr

1 Department of Computer Science and Engineering, Chungnam National University, Daejeon, South Korea

when infringement accidents occur in the information systems of organizations, it is difficult to extract, integrate, and analyze personal information access records from the separate logs of the various information systems and the personal information access records specified by legal regulations. This environment makes it difficult to quickly analyze and respond to infringement accidents (Andersson and Nilsson 2014; Patrascu and Patriciu 2014; Roratto and Dias 2014).

In September 2016, Yahoo disclosed that information related to approximately 500 million user accounts was leaked via hacking in 2014. The items leaked at that time were various personal information items such as name, e-mail, phone number, and date of birth. Many lawsuits were filed in connection with this case. In October of the same year, the personal information of 50 million Uber users and 7 million Uber drivers was leaked via hacking, and Uber paid $100,000 to hackers. Personal information leaks are costly, as they lead to corporate business interruptions, productivity drops, and lost revenue. In addition, such incidents result in the loss of time and money, owing to the necessity for post-attack responses such as restructuring the corporate security infrastructure or hiring additional personnel to prevent recurrence and to increase security.

Therefore, there is a need to collect logs generated in different information systems in order to extract and manage only personal information access records for the quick and accurate analysis of infringement accidents. Personal information access records can be used not only for analyzing infringement accidents but also as legal evidence. However, they have no value if their integrity cannot be guaranteed, i.e. if the information can potentially be deleted or forged. Therefore, it is critical to detect attempts to delete or modify personal information access records and thus prove their integrity (Eye et al. 2014).

This article presents a method to prevent the forgery and deletion of personal information access records and prove their integrity. Also, in the existing authentication method, it was not possible to identify which part of the log data was attacked. However, this study suggested a method of detecting the point of attack. A method to manage the integrity verification of personal information access logs generated from various information systems through a third-party verification institution and thus guarantee their reliability for infringement incident analyses and forensic investigation processes is also described.

2 Related works

2.1 MAC-based authentication method

The logs generated by information systems record events such as user activity, use of system resources, data access, and change of data. In the event of an infringement accident, attackers try to access and delete logs to conceal their intrusion path and remove traces of their malicious activities. To prevent this, the integrity of logs generated and recorded by a system must be safeguarded. Schneier and Kelsey (1999) suggested a method to guarantee the integrity of each log generated in information systems by creating a message authentication code (MAC) that uses different secret keys for the verification process. This method can guarantee log integrity even if the secret key is exposed by an attacker, because a verification key is also generated when the secret key of each log is generated. However, a drawback of this method is that even though a forgery can be detected with the verification value of each log, it is impossible to detect if the data of the last log have been deleted by attacker. To overcome this drawback, Ma and Tsudik (2009) used the forward-secure sequential aggregate (FssAgg) MAC authentication method. They guaranteed integrity by generating a verification value when logs are transferred to the data collection point between wireless sensors. The FssAgg authentication method has the advantage of maintaining the integrity of the individual verification values of each previously generated log through forward security, even after the key is exposed. In FssAgg authentication, different secret keys are generated each time through the hash chain method, and the MAC of each log is calculated with the secret key. In addition, a value is generated to verify the logs in an integrated manner; a new value is generated in the verification step, which is compared with the previously generated value to prove log integrity. FssAgg MAC authentication progresses through four steps to prove data integrity, as presented in Table 1. However, one drawback of this method is that the integrity is proven through a single verification value that is

Table 1 FssAgg MAC authentication steps

Step	Main tasks
FssAgg.Kg	An initial secret key s of a fixed length is generated using the symmetric key generation algorithm
FssAgg.Asig	MACs, which are individual verification values for the data $M_1, \ldots, M_i$, are generated with different secret keys, and a verification value is derived
FssAgg.Upd	The secret keys generated in the FssAgg.Kg step are updated
FssAgg.Aver	The integrity of data $M_1, \ldots, M_i$ is verified

generated for all logs; we cannot know when the forgery or deletion attack occurred.

2.2 Other authentication methods

Zhu and Lee (2016) and Keegan et al. (2016) studied the detection of network intrusions and a means to guarantee integrity in a cloud environment. Im et al. (2015) proposed a method of reducing the cost and vulnerabilities of certification by using a public key infrastructure instead of the conventional certification method. Holt (2006) proposed a method for generating the verification value of each log by using multiple public keys. This method offers the advantage of higher security because the key used in the signature step is different from the key used in the verification step. However, this method is vulnerable to the log deletion attack because only the integrity of individual log data is proven with no integrated verification. Another drawback is that the processing time is longer during the generation and verification process of the verification value owing to the nature of the public key-based structure, because the data of each log data are signed separately. Therefore, it is inappropriate for proving the integrity of a large amount of log data. Choi and Kim (2013) suggested a method to verify software integrity by using a third-party verification institution. In this method, the integrity verification value of a normal software application is registered at a third-party verification institution before the application is distributed. Then, before a user uses the software application, a verification value is requested from the third-party verification institution and compared to check for forgery of the software application. This method has the advantage of guaranteeing integrity from both internal and external attacks, because neither malicious attackers nor internal employees such as software developers can forge the verification value. However, it has the limitation that it takes a long time to register and verify the integrity verification value in an environment with frequent software updates.

3 Security requirements for the log system

3.1 Forward security and stream security

A log system sequentially records data such as the behavior of users accessing an information system, events that occur in the system itself, and the data processing history. Thus, log records can be used to examine and respond to incidents that occur in a system, and logs can be analyzed and investigated in the event of an infringement accident. Secure storage and management of logs is critical because attackers commit forgery or deletion attacks on generated and stored logs to destroy the traces of their intrusion. Measures are needed to detect and prevent such attacks (Veeken 2018; Khan et al. 2017; Rajalakshmi et al. 2014).

If an attacker forges the verification value for proving the integrity of access records and the personal information access records to delete traces of an intrusion, it becomes difficult to detect and analyze the forgery. Therefore, an integrity protection measure is required to prevent the forgery of log verification values before an intrusion. Such a protection measure is referred to as forward security. Forward security generates a verification value by using a different key for each log, and the unidirectional property of the hash function is used. A crucial characteristic of this method is that each succeeding key can be derived from the key used to generate the previous individual verification value, but it is impossible to infer the previous key from the derived key (Bellare and Yee 1997).

Even if a log system meets the requirements of forward security, it is difficult to detect if a log item is selectively deleted or the log sequence is changed by an attacker. Therefore, integrity verification is required for the sequential flow of logs; this is referred to as stream security. In stream security, when an integrated verification value is generated for all logs, the previously generated integrated verification value is combined with the newly generated individual verification value of the unit log. This combined value is converted to a new integrated verification value through a hash function. This assigns the dependency of each log with the previous log (Yavuz and Ning 2012). If the attacker rearranges the log sequence of an information system, the rearrangement attack on the log data can be detected because the verification value at the time when the log was created is different from the newly created verification value (Veeken 2018; Wouters 2012; Yavuz and Ning 2009).

3.2 Detecting log forgery and deletion attacks

To allow a log forgery or deletion attack to be detected, a verification value is generated with a secret or private key each time a log is generated. In addition, an integrated verification value is generated with a hash function for integrated verification of all logs, as shown in Fig. 1.

Individual verification values can be used to detect forgery attacks on each log (L_i). When logs are created, individual verification is generated with a secret or private key. When the integrity of a log is verified, the verification value generated at the time of data creation is compared with the newly generated verification value. If the new verification value differs from the previous verification value, it indicates that the corresponding log was attacked. However, individual verification values can only prove the integrity of individual logs. If the logs and their individual verification values are deleted together, detection and analysis are difficult.

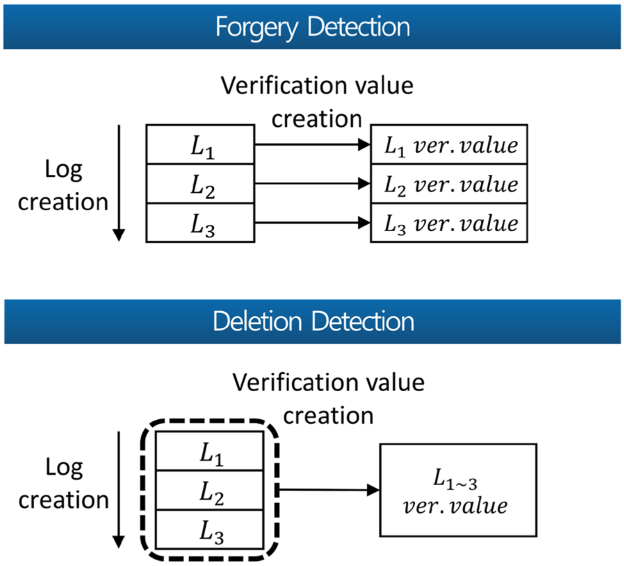

Fig. 1 Detection method for forgery and deletion attacks

Integrated verification values can be used to detect forgery and deletion attacks on each log. When a log (L_i) is created, it is combined with the previous logs into a single log, and an integrated verification value is generated by deriving a hash value from the combined log. With this method, even if some logs are deleted, the verification value generated from the integrated log changes. Thus, a deletion attack can be detected by comparing the integrated verification value generated at the time of log creation with the integrated verification value generated after log deletion.

3.3 Characteristics of log data

Logs are continuously generated during system operation, and the amount of log data becomes very large with increasing system operation time. Owing to the large amount of log data, if all verification values for all unit logs are managed, the system load can increase during the data integrity verification step. Furthermore, the digital signature method of using a private key when generating an individual verification value is more secure than the method of creating a MAC with a secret key, but it involves higher computation time. Therefore, it is important to select a method while considering the log system environment when large volumes of log data are generated. When a log system is attacked, it is critical to quickly determine the time of attack. Although the logs generated by systems and applications have different components and formats, most logs contain the log generation times. The time information of logs is used to analyze infringement accidents and resolve system troubles. Therefore, efficiently managing the verification values of log data

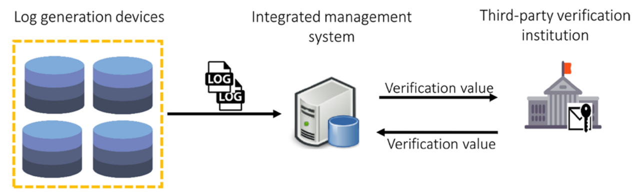

Fig. 2 Proposed system structure

using the time information is important. There are four main requirements for the protection and management of logs:

1. Protection methods that suit the log characteristics should be prepared.
2. Forgery and deletion attacks should be detectable.
3. Forward security and stream security should be followed.
4. The point where the log is attacked should be identifiable.

4 Proposed measures

In an environment where personal information access records are managed in an integrated manner, logs created in various log systems are collected and stored. Personal information access records are identified in the collected logs and separately extracted, stored, and managed in a new database. As the personal information access records are managed in an integrated manner, protection measures are required, and a way to prove integrity against forgery and deletion attacks is crucial.

In the existing environment, personal information access records are stored separately. In this study, however, personal information access records are proposed to be stored in an integrated system and verified through a third-party verification institution. As shown in Fig. 2, the personal information access records from various applications and security systems are integrated, and verification values are created and sent to a third-party verification institution.

The integrated management system can guarantee the reliability of stored logs against internal and external attacks because the verification values are managed by a third-party verification institution. When the integrated management system requests log integrity verification, the third-party verification institution sends the verification value. In this study, the process of generating, integrating, storing, and verifying the personal information access logs was assumed to be repeated periodically. In other words, the personal information access records are assumed to be stored periodically in the integrated management system. In this section, the integrity verification of personal information access records is explained in three steps: an early stage involving information exchange, a generation stage of integrity verification values, and a verification stage for integrity. A third-party verification institution is employed to verify the integrity of personal information access records against internal and external attacks is proposed.

4.1 Early stage of information exchange

In the early stage of information exchange, the verification value for personal information access records is generated, and the procedure in Fig. 3 is carried out for the safe transfer of the verification value between the integrated management system and third-party verification institution. First, the integrated management system and third-party verification institution create a pair comprising of a public key and private key, respectively. Next, the integrated management system requests the public key from the third-party verification institution. After confirming the public key request message, the third-party verification institution sends its public key to the integrated management system. Through this process, the integrated management system and third-party verification institution can encrypt the data transferred between them through the exchanged public key. In this process, the integrated management system and third-party verification institution are assumed to be mutually certified.

After the public key is exchanged, the integrated management system generates a secret key to create the first MAC as a seed and encrypts it with its own identifier and the public key of the third-party verification institution. Next, the integrated management system makes a digital signature with its private key and sends it to the third-party verification institution. The digital signature cannot be forged because only the signer can create the signature value, and the receiver of this value can verify the signature value with the public key of the signer. This method can prove that the transferred message has not been forged during transfer and offers the advantage of a non-repudiation feature. Finally, the third-party verification institution decrypts the secret key received from the integrated management system and saves it.

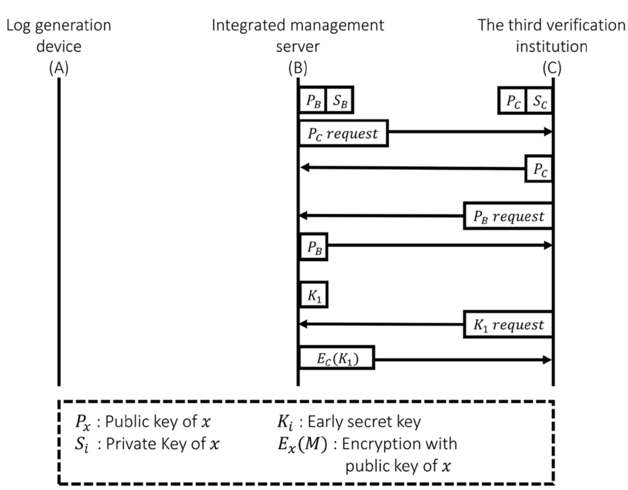

Fig. 3 Early information exchange

4.2 Generation stage of integrity verification values

In the generation stage of integrity verification values, a MAC is generated whenever a set of personal information access records is stored in the integrated management system. Even though digital signatures can be used to create a verification value for each personal information access record, the creation of a verification value with a digital signature is slower than using a MAC and less efficient in an environment where a large amount of data is generated. Therefore, similar to the FssAgg MAC authentication method, the verification value of each personal information access record is created with a MAC. For the secret key to create the MAC, the seed generated in the information exchange step is used. When the next MAC is generated, a different secret key is created by applying the hash chain mechanism to the seed. The used secret key is deleted immediately after the secret key for the next MAC is derived. This is a forward security method to protect the data generated before the secret key is exposed by an attack. The forgery of personal information access records can be detected by creating a MAC for each personal information access record. Figure 4 shows the procedure for deriving individual verification values with the MAC and secret key to which the hash chain mechanism that satisfies the forward security has been applied.

For each MAC generated in this manner, the root value is created with the Merkle tree mechanism. This is used as an integrated verification value for one cycle of collecting the personal information access records. Figure 5 shows the total process of creating the integrity verification value. This integrated verification value can be used to detect a deletion attack on personal information access records collected in one collection cycle.

The third-party verification institution only manages the integrated verification values instead of all the individual

K_i : ith secret key, K_1 : Seed

h(x) : Unidirectional hash function

R_i : ith personal information access records

$MAC_{ki(x)}$: MAC function

Z_i : ith individual verification value

$K_i = h^{i-1}(Seed)$, $Z_i = MAC_{ki}(R_i)$

Fig. 4 Procedure for deriving individual verification values

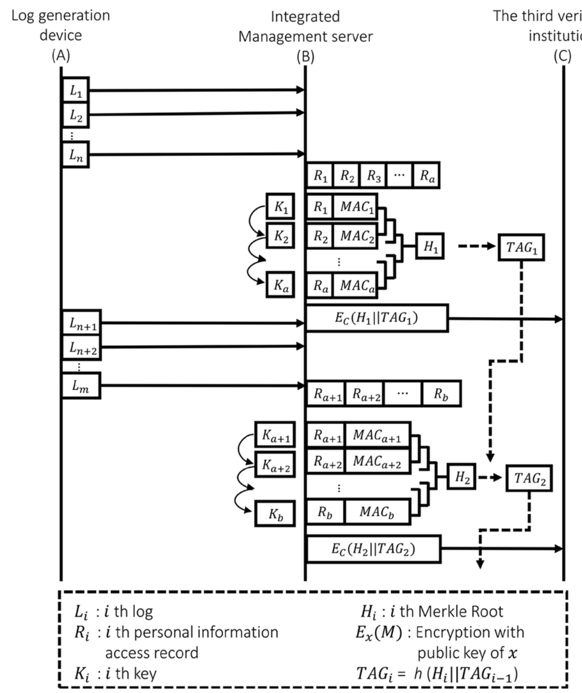

Fig. 5 Generation of verification values

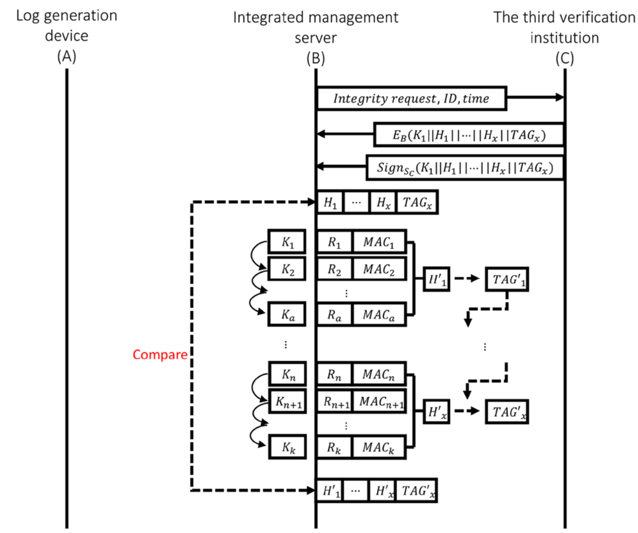

Fig. 6 Verification stage for integrity

verification values. This requires less storage space and involves a smaller communication load.

As this integrated verification value is generated per collection period as time elapses, the number of integrated verification values gradually increases. In such a case, suitable verification of the integrity cannot be realized if the attacker rearranges the sequence of integrated verification values. For this reason, security for the integrated verification values (i.e., stream security) should be verifiable. Thus, TAG generates values to be used to verify the integrity of a sequence of integrated verification values upon their generation. When an integrated verification value is generated, it is derived in a form that generates a hash value by combination with the previously generated TAG. As the value is derived by configuring the integrated verification value in chain form, TAG can be used to verify the integrity of the sequence.

The generated integrated verification value and TAG are transmitted to the third-party verification institution. The integrated verification value and TAG are encoded for transmission by using the public key. Given that one TAG is generated per collection period, data transmission to the third-party verification institution is performed per TAG generation period. When the third-party verification institution is transmitted a verification value of the Merkle tree and TAG from the integrated management server, this is decoded by using its own individual key. A new TAG is derived from the previously received TAG and used with the current integrated verification value to confirm its integrity by comparison with the current TAG.

4.3 Verification stage for integrity

The verification stage for integrity proves that the personal information access records stored in the integrated management system have not been forged or deleted. This step is started as needed by the integrated management system. When the integrated management server requests an integrity verification value from the third-party verification institution, the latter transmits the initial secret key and integrated verification value saved by itself along with the TAG that was saved last to the integrated management server. At this point, the third-party verification institution electronically signs the data with its own individual key and transmits the data after decoding it with the public key of the integrated management server. After receiving this value, the integrated management server decodes it by using its own individual key, affirms the electronic signature by using the public key of the third-party verification institution, and checks whether the third-party verification institution is the correct transmission point. Next, the task performed in the generation stage of integrity verification values is performed again on the personal information access records saved by itself. At this time, the integrity is verified by comparing the new integrated verification value and TAG with the values received from the third-party verification institution for agreement, as shown in Fig. 6.

If the newly generated verification value differs from the value received from the third-party verification institution, it indicates that the personal information access records were attacked. The point in time when the attack occurred can be identified through comparison with the integrated verification values.

5 Comparison of previous methods and proposed method

In order to guarantee the integrity of the personal information access record, this study defined the requirements for the log management system in Sect. 3.3. In this chapter, we compare the existing methods to meet the requirements and the methods presented in this study. The FssAgg authentication method verifies the integrity of all log data with one integrated verification value. Therefore, it is possible to detect whether the data has been tampered with or deleted, but the point of attack cannot be determined. The method using MAC and the method using digital signature generate a verification value corresponding to each log data and compare the values in the integrity verification step to verify the attack time of the log data. MAC and FssAgg authentication methods guarantee the integrity of data flow by assigning dependencies between data during the step of generating verification values. However, the method using the digital signature does not satisfy the flow security because there is no such process. Also, there is a problem that the method using MAC does not detect attacks that delete some sets at the end of log data. The method using MAC and the method using digital signature generates a verification value for each log data and speed reduction occurs because the verification value newly created in the integrity verification step with the verification value that has been generated previously. Therefore, these are not suitable for use in an environment where a large amount of personal information access logs are generated. By contrast, the FssAgg authentication method performs integrity verification of all log data with a single integrated value, so that the system is not burdened. Table 2 compares existing security log authentication methods with the method proposed in this study.

6 Conclusions

Personal information access records are used to analyze the cause of infringement accidents and search for solution measures. They can also be used as legal evidence in the event of infringement accidents. Therefore, such logs are a target for attacks, and must be protected.

The present article proposes a method of proving the integrity of personal information access records in an environment of integrated management. As verification is complex, and there are multiple methods to prove integrity, a safer and more efficient log system can be constructed. The reliability of the personal information protection can be guaranteed because the integrity is proven by a third-party verification institution.

This study is meaningful from three perspectives. First, we proposed a superior integrity verification method to match the characteristics of the personal information access log. The existing public key and private key method for verifying the integrity of mass and continuous personal information log data has a drawback in that it can slow down the operation and increase load on the system. However, in this study, the MAC scheme is used. Second, the integrity of personal information access records was ensured by using a third-party trust authority. If an attack on the personal information access record occurs within the organization, it may be difficult to verify the objective integrity by managing the verification value in the system. Therefore, we propose a method to reliably verify the integrity of the personal information access log against an attack that occurs internally by using the third-party trust authority. Third, we used a more efficient method to detect when an attack on the personal information access log occurred. In the existing FssAgg authentication method, it was not possible to identify which part of the log data was attacked. However, this study suggested a method of detecting the point of attack.

The proposed method enables more secure management of personal information access records because it can detect both internal and external attacks. Furthermore, faster and

Table 2 Comparison of existing methods with the proposed method

	MAC authentication method	FssAgg MAC authentication method	Digital signature method	Proposed method in this study
Consider the log characteristics	○	X	○	○
Detection of forgery attacks	○	○	X	○
Detection of deletion attacks	○	○	X	○
Forward security	○	○	○	○
Stream security	X	○	○	○
Identify the point of attack	X	○	X	○

more accurate analysis and response to infringement accidents can be expected.

Acknowledgements This work was supported by Institute for Information and communications Technology Promotion (IITP) Grant funded by the Korea government (MSIT) (No.2016-0-00193, IoT Security Vulnerabilities Search, Sharing and Testing Technology Development).

Compliance with ethical standards

Conflict of interest The authors declare that there is no conflict of interest regarding the publication of this paper.

References

Andersson M, Nilsson A (2014) Improving integrity assurances of log entries from the perspective of intermittently disconnected devices. MECS-2014-10

Bellare M, Yee B (1997) Forward integrity for secure audit logs. Tech Rep. University of California Press, San Diego

Choi Y-S, Kim I-K (2013) Software integrity authentication with trusted third-party party. Res Notes Inf Sci. https://doi.org/10.4156/rnis.vol14.18

Holt JE (2006) Logcrypt: forward security and public verification for secure audit logs. In: Proceedings of the 2006 Australasian workshops on grid computing and E-research, vol 54. Australian Computer Society, Inc. Darlinghurst, Australia, pp 203–211

Im H, Kang J, Park JH (2015) Certificateless based public key infrastructure using a DNSSEC. J Converg 6(3):26–33

Keegan N, Ji S-Y, Chaudhary A, Concolato C, Yu B, Jeong DH (2016) A survey of cloud-based network intrusion detection analysis. Hum Cent Comput Inf Sci. https://doi.org/10.1186/s13673-016-0076-z

Khan A et al (2017) Secure logging as a service using reversible watermarking. Procedia Comput Sci 110:336–343

Korea Communications Commission (2012) Standard for technical and administrative protection of personal information. Guide of notification, Korea Communications Commission, Seoul

Ma D, Tsudik G (2009) A new approach to secure logging. ACM Trans Storage 5(1):2

Ministry of the Interior (2017) Standard for securing personal information. Guide of notification, Ministry of the Interior, Seoul

OECD Council (1980) OECD guidelines on the protection of privacy and transborder flows of personal data. Organization for Economic Cooperation and Development, Paris

Patrascu A, Patriciu V-V (2014) Logging system for cloud computing forensic environments. J Control Eng Appl Inf 16.1:80–88

Rajalakshmi JR, Rathinraj M, Braveen M (2014) Anonymizing log management process for secure logging in the cloud. Circuit, Power and Computing Technologies (ICCPCT), 2014 International Conference on. IEEE

Roratto R, Dias ED (2014) Security information in production and operations: a study on audit trails in database systems. JISTEM J Inf Syst Technol Manag 11.3, 717–734

Schneier B, Kelsey J (1999) Secure audit logs to support computer forensics. ACM Trans Inf Syst Secur 1(3):159–176

Swanson M, Guttman B (1996) Generally accepted principles and practices for securing information technology systems. National Institute of Standards and Technology. Gaithersburg 800–14

Van der Veeken P (2018) Constructing a confidential, authenticated, forward secure and offline logging scheme. Dissertation. Delft University of Technology

Von Eye F, Schmitz D, Hommel W (2014) A framework for secure logging with privacy protection and integrity. Conference on Internet Monitoring and Protection (ICIMP), pp 14–19

Wouters K (2012) Hash-chain based protocols for time-stamping and secure logging: formats, analysis and design. Diss. Dissertation report. Arenberg Doctoral School of Science, Engineering and Technology, Katholieke Universiteit Leuven, Belgium, pp 1–220

Yavuz AA, Ning P (2009) Baf: an efficient publicly verifiable secure audit logging scheme for distributed systems. Computer Security Applications Conference, 2009. ACSAC'09. Annual. IEEE, pp 219–228

Yavuz A, Ning P (2012) BAF and FI-BAF: efficient and publicly verifiable cryptographic schemes for secure logging in resource-constrained systems. ACM Trans Inf Syst Secur 15:9

Zhu W, Lee C (2016) A security protection framework for cloud computing. J Inf Process Syst 12(3):538–547