J Inf Process Syst, Vol.15, No.3, pp.520~537, June 2019
https://doi.org/10.3745/JIPS.03.0116

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS

A System for Improving Data Leakage Detection based on Association Relationship between Data Leakage Patterns

Min-Ji Seo* and Myung-Ho Kim*

Abstract

This paper proposes a system that can detect the data leakage pattern using a convolutional neural network based on defining the behaviors of leaking data. In this case, the leakage detection scenario of data leakage is composed of the patterns of occurrence of security logs by administration and related patterns between the security logs that are analyzed by association relationship analysis. This proposed system then detects whether the data is leaked through the convolutional neural network using an insider malicious behavior graph. Since each graph is drawn according to the leakage detection scenario of a data leakage, the system can identify the criminal insider along with the source of malicious behavior according to the results of the convolutional neural network. The results of the performance experiment using a virtual scenario show that even if a new malicious pattern that has not been previously defined is inputted into the data leakage detection system, it is possible to determine whether the data has been leaked. In addition, as compared with other data leakage detection systems, it can be seen that the proposed system is able to detect data leakage more flexibly.

Keywords

Apriori Algorithm, Associated Abnormal Behavior List, Comprehensive Leakage Detection Scenario, Convolutional Neural Network, Data Leakage Detection

1. Introduction

Companies store important confidential information, such as intellectual property, financial information, and personal information, in the form of digital data on servers. Storing important confidential information on a server is advantageous in that the processing of information can be efficiently performed, but it is disadvantageous in that company insiders can easily leak the data through various means, such as web mail or mobile devices.

The InfoWatch Analytical Center reported that, from 2006 to 2016, the incidence of data leakage continued to rise, with more than 60% of the incidents being attributed to insiders including potential retirees [1].

In order to respond to data leakage accidents, companies are currently applying various security solutions [2,3] to monitor insider work activities or limit non-work-related activities. Each security

Manuscript received June 29, 2017, first revision March 13, 2018; second revision June 8, 2018; third revision September 20, 2018; accepted October 22, 2018.
Corresponding Author: Myung-Ho Kim (kmh@ssu.ac.kr)
* Dept. of Software Convergence, Soongsil University, Seoul, Korea ({porito2, kmh}@ssu.ac.kr)
This paper is an expanded and more detailed version of a paper entitled “Internal information leak detection system using times-series graph” in *Proceedings of the 2017 Spring Conference of the KIPS*, Jeju, Korea in April 28, 2017.

solution records the results of insider monitoring activities in the form of a security log that allows an administrator to respond to insider behavior by analyzing the log records when specific data is consumed in order to identify the path of abnormal behaviors, and a system that comprehensively analyzes the security log.

From this point of view, enterprises currently use the security information & event management (SIEM) solution [4]. This helps the administrator determine the appropriate response after collecting the system resource data and logs generated from each business activity. However, in order to understand the abnormal behavioral pattern, the SIEM solution must analyze all of the security logs generated by each insider and output the results, so it has the problem that the administrator cannot grasp the result of analyzing the behavioral pattern for data leakage of the insider in a short period of time. In addition, various studies [5-7] have been conducted to detect behavior related to data leakage, but it is difficult in practice to comprehensively analyze the behavior of insiders and detect various leakage patterns.

In the proposed paper, a leakage detection scenario for data protection, which is a criterion for detecting data leakage, is defined the trend of the behavior log indicating the behavior of the insider is graphed, and the results are analyzed by deep learning according to each scenario. A method for understanding malicious insider and data leakage behavior is suggested.

The leakage detection scenario represents a series of behavior to leak data. In the proposed system, it is possible to immediately respond to an insider who is suspected of showing leaking action in each scenario by analyzing all of the security logs happening during the day. The leakage detection scenario consists of the basic behavior scenarios already stored in the scenario database along with the associated behavior scenarios newly constructed by applying the Apriori algorithm [8] to the logs showing behaviors. The Apriori algorithm is one of the association analysis algorithms, and is mainly used for extracting a set of highly related data by analyzing the association between data. The association behavior scenario applies the Apriori algorithm to the log history of the malicious insider in order to extract and construct a set of behaviors that can occur upon data leakage. Therefore, if data leakage is detected through the list of leakage detection scenarios defined in this study, it is possible to more efficiently detect insiders who attempt to leak data.

The proposed system converts each behavior log into a type of time-series graph, then combines a set of behavior log graphs constituting each leakage detection scenario into a single image, and analyzes it using a convolutional neural network (CNN) [9]. The CNN is a deep learning algorithm that shows high performance in processing large amounts of data and image recognition. Because each image is analyzed through the CNN, it is possible to classify the graph image that shows malicious behavior, and to detect a malicious insider more accurately than the existing security system. In addition, since the graph image input to the CNN shows a set of behaviors that can appear upon data leakage, it is possible to grasp the types and order of behaviors causing the insider to leak the data.

This study is composed as follows. Section 2 shows how to detect data leakage by analyzing the association between network packets, as well as how to prevent the leakage of data that has been studied. Section 3 describes the system realized in this study. Section 4 verifies the accuracy of the proposed system using various methods, while Section 5 concludes the paper.

2. Related Research

Studies by many researchers have continued to focus on protecting data in order to prevent malicious data leakage by members of the organization, or by insiders certified by the system. Typical data protection methods include intrusion detection systems and intrusion prevention systems. An intrusion detection system is an automated system that monitors events occurring in a computer or network in order to detect intrusions. An intrusion prevention system is a security solution that prevents intrusion in real time and uses various security technologies to block harmful network traffic [10].

These systems are divided into network-based intrusion detection system (NIDS)/network intrusion prevention system (NIPS) or host-based intrusion detection system (HIDS)/host-based intrusion prevention system (HIPS), depending on whether the system is operated based on the network or host. NIDS/NIPS detects abnormal traffic appearing on the network, then blocks or responds to attacks in advance. The HIDS/HIPS monitors system logs, critical system files, and other resources that occur on the host computer itself in order to prevent anomalous processes in advance.

IDS/IPS monitors the behavior of insiders based on signature-based detection systems and abnormal-based intrusion detection systems. A signature-based detection system involves recording the characteristics of a data leak event that has already occurred in a database, and then blocking the pattern when an intrusion pattern stored in the database is detected afterwards. This method shows a high detection rate when the inputted intrusion pattern database shows good performance, but a low detection rate when the intrusion pattern is not properly inputted into the database. The abnormal-based intrusion detection system records normal activity, and when an action out of the reference value is detected, blocks the corresponding process and traffic. However, in the case of an abnormal-based intrusion detection system, there is a risk that the normal process can be handled abnormally.

Enterprises use data loss prevention (DLP) to prevent data leakage by insiders in advance. DLP is a data protection solution that manages security logs output from security devices such as firewalls and anti-virus programs, then blocks access to important information by setting certain rules. In order to protect critical information, the DLP solution applies digital rights management (DRM) technology, which is an encryption technology, to each important document. When abnormal access to the data with no access right is detected, it protects the data by blocking such access, after granting access rights according to the level of each insider. In addition, DLP defines use rules for each important item of information, and if a rule violation is detected, blocks access to data.

Event data and logs output from each security device are all collected in the integrated log analysis management system SIEM. The SIEM solution analyzes the security log history of the insiders based on the input log analysis rule list, and outputs the log list showing abnormal symptoms to the administrator, so that the administrator can analyze the log and respond appropriately. The SIEM solution mainly consists of a rule-based detection method that determines whether the insider has violated the log analysis rule list, and a method in which the value derived by a previously defined formula is judged for each abnormal behavior item, in order to analyze the log. Rule-based detection methods and score-based detection methods have disadvantages in that if they show behavior patterns that have not been input previously, they cannot respond properly to the system.

Further, research is currently being conducted into a method for detecting not only existing invasion patterns, but also related patterns through the use of an association analysis algorithm. Wuu et al. [11] carried out research to update the signature rules used by Snort, one of the NIDS tools, to monitor and analyze network traffic through the Apriori algorithm. The authors [12] applied the Apriori algorithm to normal network packets so as to create a normal behavior pattern list, and if a traffic item is detected showing a pattern different from the normal behavior pattern, such traffic is blocked.

Such research progresses in the following order. First, a transaction to define the network behavior is defined. In order to create a normal behavior pattern, a transaction consists of several packet groups. In this case, to prevent the packets corresponding to different network actions from being defined as the same transaction group, a transaction is generated by grouping packets whose difference between the time when the packet is generated, and the time when the previous packet is completed, is within some predetermined value. When the transaction has been defined, a group of packets of size 2 or more is extracted from the frequently occurring packet list using the Apriori algorithm, and this group is used to define the normal behavior pattern. Then, when a new network action is input, it is compared with the normal action pattern in order to determine whether the action is abnormal.

When abnormal behavior is determined by the normal behavior pattern defined through the association of the packets constituting the network behavior with the Apriori algorithm, it faces a problem that it can be mistaken for abnormal behavior even when the normal behavior is inputted, since the kinds of behavior patterns that can be performed by the insider are various.

As the amount of log generated during the day grows exponentially, research into detecting abnormal insiders by analyzing the log through the use of a machine learning algorithm is currently underway. Julisch [13] used the clustering algorithm for which the unsupervised model showed better results than supervised learning. The authors [14] show a new attempt to detect insider threat using graph learning. Das et al. [15] used the principal component analysis algorithm applied to the information data on the network for detecting abnormal packets in order to prevent intrusion.

Recently, the deep learning algorithm has been developed, and studies have been conducted to detect abnormal packets by learning normal network packets through the use of a deep neural network [16]. However, these studies have trained the normal behavior of insiders in the learning model, and when another pattern of behavior is detected on the trained normal activity, that behavior is regarded as abnormal. Therefore, there is a problem that the accuracy of detection can greatly change according to the learning data.

In order to overcome these problems, we propose a new algorithm that can detect malicious insiders that leak data by using associative analysis algorithms that exhibit high performance for extracting related data, and deep learning algorithms that exhibit high performance for processing a large amount of data.

In the proposed system, the association analysis algorithm is applied to the security log history shown in the data leakage security incident history, and the series of behaviors that can appear to leak the data are defined as the leakage detection scenarios. Therefore, data leakage can be detected more flexibly using the proposed system than existing intrusion detection systems. In addition, since logs are created in bar graphs according to each scenario rule and analyzed through a deep learning algorithm called a CNN, it is possible to detect abnormalities with high accuracy. It is also possible to determine the type and time of the action.

3. A System for Improving Data Leakage Detection Based on Association Relationship between Data Leakage Patterns

At present, as data leakage accidents are frequently caused by insiders, companies are applying several security solutions that monitor activity logs and restrict the use of data as appropriate. However, these systems are dependent on existing data leakage patterns; when data is leaked in a manner that has not been defined by an administrator, data leakage cannot be properly determined. This study suggests a system that can detect data leakage by an insider, even if the data is leaked through a pattern that has not been input to the system.

Fig. 1 shows that the proposed system consists of a log collection unit, a leakage detection scenario establishment unit, and a data leakage definition unit. The log collection unit collects security logs that are meaningful in determining whether data has been leaked from the security solution handled by the enterprise. The leakage detection scenario establishment unit defines the leakage detection scenario list in order to analyze the log history and judge the data leakage, and stores it in the database. The data leakage definition unit collects all log details that may appear during business activities on each security device. It judges the data leakage by classifying the collected security log as a behavior log group that constitutes each leakage detection scenario. Then, graphs are inputted into the CNN, and it searches for an insider with high similarity to the security log graph of the malicious insider.

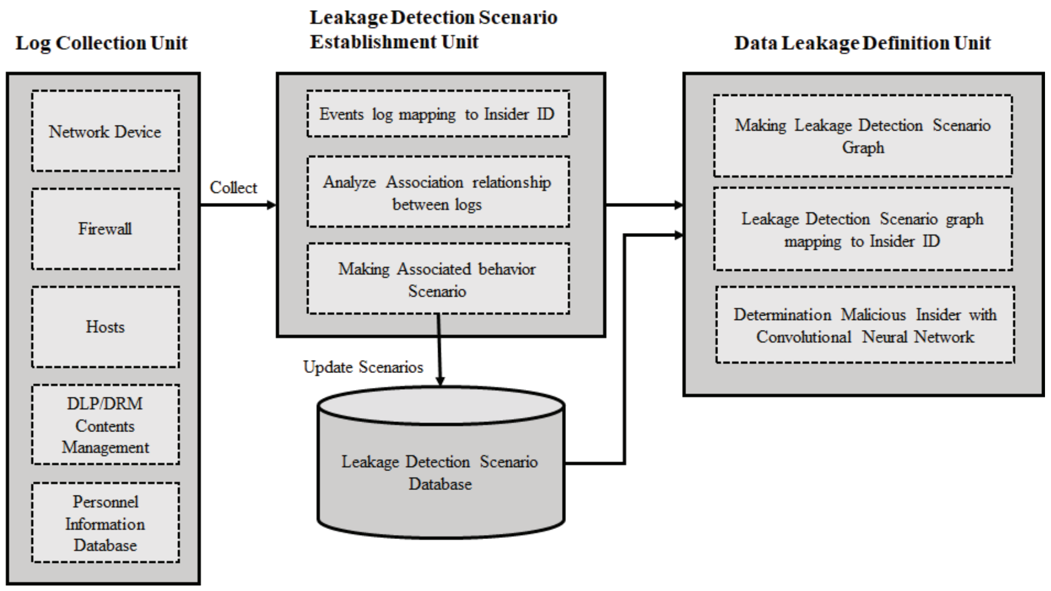
Log Collection Unit
Network Device
Firewall
Hosts
DLP/DRM Contents Management
Personnel Information Database
Collect
Leakage Detection Scenario Establishment Unit
Events log mapping to Insider ID
Analyze Association relationship between logs
Making Associated behavior Scenario
Update Scenarios
Leakage Detection Scenario Database
Data Leakage Definition Unit
Making Leakage Detection Scenario Graph
Leakage Detection Scenario graph mapping to Insider ID
Determination Malicious Insider with Convolutional Neural Network

Fig. 1. Data leakage detecting system based on association relationship.

3.1 Log Collection Unit

The behavior of each insider in the enterprise is recorded and stored in the form of a security log in the security solution, and abnormal behavior can be detected by analyzing the history of the security log collected from each insider. Security logs can be collected from the security devices applied by the enterprise, and different types of security logs are collected depending on the type of security device.

The security solution can largely be divided into a network security solution, system security solution, and contents/information leakage prevention solution according to the security device environment. Network security solutions mainly block unauthorized network ports and abnormal network traffic on the network, or attempts to break into an internal network used by an enterprise in an external network, and they save the traffic information and access time in a type of log. The system security solution stores and manages the events appearing in the host PC used by insiders. The content/information leakage prevention solution stores all of the records that are accessed to information protected by DRM, which is an encryption technology. Specifically, it stores the information of an insider who attempts access by blocking the access in advance, if an attempt is made to access data that does not fit the insider's authority, or if an attempt is made to access data through an action unrelated to the business.

The proposed data leakage detection system collects the log output from each security device, and makes it possible to judge data leakage based on the data re-processed by the graph image according to the leakage detection scenario.

3.2 Leakage Detection Scenario Establishment Unit

3.2.1 Definition of basic behavior scenario

In order to determine whether data has been leaked based on collected logs, it is necessary to define specific criteria for analyzing the logs. Therefore, we define a scenario of a series of behaviors that can occur when data is leaked, classify the collected logs according to the leakage detection scenario, and determine the data leakage based on whether or not each log shows an abnormal pattern.

A leakage detection scenario is defined as a set of behaviors that can occur for each security solution. For example, in the case of the 'DLP' scenario, all event activities, such as 'USB connection' and 'print output of protected document', that can be collected by the DLP security solution are defined, and this enables the data leakage to be judged by classifying the security log based on the leakage detection scenario, when the security log is collected from insiders afterwards.

The leakage detection scenarios consist of a basic behavior scenario and an associated behavior scenario. The basic behavior scenario defines a behavior scenario as a series of actions that may occur in cases of data leakage that have already been analyzed by the administrator. The association action scenario is defined by analyzing the security log history of those who leaked the data using the Apriori algorithm, and finding a set of behaviors that show a high correlation.

The basic behavior scenario is defined as the scenario in which behaviors for data leakage are classified into security solutions based on data leakage incidents that have already been analyzed by the administrator. For example, most insiders download or copy protected documents stored in the enterprise to leak the data, then export them via mail or portable devices. In this case, the basic behavior scenario is created as a DRM document access scenario that manages important documents, a mail management scenario, and a portable device scenario, and the behaviors corresponding to the respective scenarios are stored as components of the respective scenarios. The following are basic behavior scenarios that may occur in data leakage accidents. The system identifies the insider, who is assumed to have acted as a constituent of the basic behavior scenario through analyzing the security log, as a malicious insider, and blocks access to the important data.

- The scenario of accessing a harmful site: Even if an insider member does not intend to deliberately leak the data, the data can be leaked by connecting to an illegal site, or by downloading a file that has no relationship with the work. As a result, the scenario of accessing harmful sites should be organized by the number of times of blocking of connection to harmful sites, the number of times of blocking of malignant codes, the rate of increase of traffic, and the number of times of unauthorized file downloads.
- The scenario of access to shared folders: Corporations handle work by having staff members share folders and files with other staff members inside the corporation. However, in the case of a

staff member who has authority regarding the management of shared folders, such an insider can randomly correct or delete data and also give the right to access data to a different staff member. There is then the possibility of randomly reproducing the data of the corporation and leaking that data to the outside, deleting the data including his or her personnel information. As a result, the scenario related to the access to shared folders shall be organized by the number of times of access to shared folders, the number of times of having reproduced files in shared folders, and the number of times of deletions of files from shared folders.

- The data upload scenario: The insider leaks data by attaching the data to a messenger, or uploading the data to an online storage service or a file-sharing site. Because the data on a webmail or storage service can be leaked through the network protocol, in order to detect data leakage, it is necessary to detect the amount of usage of the protocol of each insider member. The data upload scenario is organized by the number of times of file uploads, blocking the use of unauthorized ports and services, and blocking the installations of web-hard programs.
- The DRM document storing scenario: Data from a document protected by DRM can be either downloaded according to the authority to use or leaked through printing. Each insider downloads a document protected by DRM, captures it as a screenshot, or creates a new document, then copies the contents of the document. As a result, the DRM document storing scenario is organized by the number of times of downloads of security documents, the number of times of copying DRM contents, and the number of times of capturing them as a screenshot.
- The database access scenario: As important data, like personnel information, are stored in the databases of corporations, these data can be leaked by frequently accessing the database. There are many cases in which a staff member intending to access a database transmits an abnormally high number of search queries, or transmits correction queries to a database, in order to correct his or her personnel information. Therefore, in the scenario related to the database, the number of times of requests for the right to access a database, as well as the number of times of having transmitted the search queries or the correction queries, are organized.
- The transport unit device scenario: Most of the data that have been acquired get leaked through the use of a transport unit, like a USB drive, an e-mail, or an output device, like a printer. In cases of data leakage to the outside, the transmission is made by reproducing the data in transport unit files of which the amount is more than when engaged in normal work, or by attaching them to an e-mail. In the case of an output device, like a printer or fax, the amount of documents being output is more than that at ordinary times. Hence, in order to detect data leakage, the scenario must be organized by the number of times of reproductions of files into the transport unit, the sizes of the reproductions of files, the number of times of access to the transport unit, the number of times of attachment of files to the e-mails, the size of files attached, the number of times of output from the output device, and the size of files that have been outputted.

3.2.2 Basic definition for Apriori algorithm

When the insider's behavior is monitored through the basic behavior scenarios, there is the problem that it is not possible to properly judge data leakage in cases when data is leaked in a pattern that is different from previous data leakage accidents. Therefore, it is possible to judge data leakage more flexibly when analyzing the behavior of insiders only using an existing leakage detection scenario, by analyzing the security log through the Apriori algorithm, and defining the relational behavior group as an associative behavior scenario, and updating the leakage detection scenarios.

The Apriori algorithm has labels, and is mathematically defined as follows.

Definition 1. A set $L = \{l_1, l_2, l_3 \dots l_m\}$ is a security solution list of extracted security logs, while a set $A = \{a_1, a_2, a_3 \dots a_m\}$ is a list of behaviors that aims to leak data during work hours. Given two sets L and A, each subset a and l is defined as an item set in the Apriori algorithm.

Definition 2. Let each of X, Y be a set of the item set; an Association rule is an implication of the form: $X \rightarrow Y$, where $X, Y \subset (L \cup A), X \cap Y \neq \emptyset$. It uses standard measures of the support and the confidence rules.

Definition 3. A set $d = \{l_1, l_2, l_3, a_1, a_2, a_3 \dots l_{i,} a_j\}$ is a transaction representing data leakage behavior collected by security solutions in data leakage accidents. In addition, a set $D = \{d_1, d_2, d_3 \dots d_m\}$ represents a list of all of the transactions.

Definition 4. Support is the item set occurrence rate as measured by the proportion of transactions in which an item set appears. Given item sets l, a, the support of rule $l \rightarrow a$ is defined as

$$support(l) = \frac{transaction_{\ contaitning\ l}}{Transaction\ Sets\ D} = p(l) \tag{1}$$

Definition 5. Support can be used to filter association rules by stating a lower bound for the support of a rule, minimum support. If item set $< l_i, a_j >$ has a value equal to or higher than the minimum support threshold, it is defined as a frequent item set.

Definition 6. Confidence measures the quality of association rules in item sets. Given item set l, a, the confidence of rule $l \rightarrow a$ (conditional probability that a transaction having item set l also contains item set a) is defined as

$$confidence(l, a) = \frac{p(l)}{p(l,a)} \tag{2}$$

3.2.3 Apriori algorithm discovering association rules about security log

The Apriori algorithm is an association analysis algorithm for analyzing the relationship between each set of items for a set of transactions composed of a set of items. Taking market research as an example, each purchased item is defined as an item set, and a purchased item list is defined as a transaction. By analyzing the relationship between items using the Apriori algorithm, it is possible to derive and recommend an item that is likely to be purchased together with an item x that is already purchased.

The Apriori algorithm consists of two major steps:

Step 1: Calculate the occurrence probability of each item set by calculating the support. A support value of an item set that exceeds the minimum support value is defined as a frequent item set.

Step 2: For a subset of the frequent item set derived in Step 1, if the confidence value is lower than the minimum confidence threshold, it is removed from the frequent item set. By contrast, if the confidence satisfies the minimum confidence threshold, it is determined that there is an association relationship among frequent item sets, and this is generated as one rule itemset1→itemset2.

We applied the Apriori algorithm to the log collection and security solution in order to determine the association relationship between abnormal behaviors in the data leakage incident history. Tables 1 and 2 show the notations, input, output, and procedure of the algorithm.

The information of the leakage behavior and the security equipment in the security log collected from the insider are each defined as one item set. A transaction includes all of the leaking activities and security solution types collected in an accident about data leakage. Therefore, it can be said that the types of leakage behavior and types of security equipment collected by the system are defined as the item set. The item set is used to generate a group of candidate item sets, which is composed of components of unequal item sets. In this case, the item sets not exceeding the minimum support threshold are removed from the candidate item set. By contrast, when the minimum support value is satisfied, the candidate set is defined as a frequency item set. The frequent item set generation operation continues, until the support value of all item sets is calculated. Finally, when the frequent item set is generated, a confidence value is calculated for a subset of the frequent item set, and a subset of the frequent item set is derived that has an association relation with abnormal behavior to leaks information. Based on the rules generated by the system, we grouped item sets constituting the association rule into one scenario, then defined the scenario as a data leakage detection scenario for creating a graph.

Table 1. Notations of Apriori algorithm

Notation	Description
MinimumSupport	Support threshold
MinimumConfidence	Confidence threshold
F	Set of frequent item set
C	Set of candidate item set
N	Number of transaction
T	All transaction set

Table 2. Input, output, and pseudo code of Apriori algorithm

Input	Each insider's data leaks behavior log, security solution list, and two threshold values of minimum support and confidence
Output	Rule of relationship behaviors and security solution for leaking data
Procedure	(1)$C = \{$behavior logs and security solution categories itemset$\}$
	(2)$F_1 = \{ F \mid F \in C, (F.count\ /\ N) \geq$ MinimumSupport$\}$
	(3) for($k = 2; F_{K-1} \neq \emptyset ; k++$)do
	(4) $C_K \leftarrow$ Making_candidate(F_{K-1})
	(5) for each transaction $t \in T$ do
	(6) for each candidate $c \in C_k$ do
	(7) if c is contained in t then
	(8) $c.count++$
	(9) endfor
	(10) endfor
	(11) $F_k = \{ \mid c \in C_k, (c.count\ /\ N) \geq$ MinimumSupport$\}$
	(12) endfor
	(13)$F = F \cup F_k$
	(14)end
	(15) generateRules(F, MinimumConfidence)

3.2.4 Applying Apriori algorithm to generate associated behavior scenario

In the proposed system, the Apriori algorithm is used to create leakage detection scenarios using the relationship between data leakage behaviors. Because the security administrator analyzes the correlation between malicious behaviors that can be collected by each security solution and creates the leakage detection scenario, which is a criterion for judging the data leakage, an administrator can easily understand the intention of the insider, as he or she can understand the relation between the insight and behaviors of the insider to leak the data through the leakage detection scenario. Therefore, this study collected all security log histories of malicious insiders that occurred on the date of data leakage, extracted the set of behaviors that were deemed to be highly relevant through the Apriori algorithm, and defined them as a type of behavior scenario. In this case, the item set for analyzing the association is made available for analyzing not only the relationship between the data leakage behavior, but also the relationship between the security solution and data leakage behaviors, by taking the structure of (security solution that collects the security log, a security log).

Assuming that the minimum support is 0.5 and the minimum confidence is 0.6, the application procedure of Apriori algorithm for item set 'DRM' and 'DRM document authorization request' in data such as those given in Table 3 is as follows. First, this study calculates the degree of support, then determines the share of each item set in the data leakage. In this study, when security log data are given in the form of (security solution type, security log), 'security solution type' and 'security log' data are separately defined as item sets. Therefore, the 'DRM' and 'DRM document authorization request' items in the T1 transaction in Table 3 are defined as an item set with a size of 1, the support is calculated for each item set, and it is judged whether or not the calculated support values satisfy the minimum support.

Table 3. Set of frequent-1 security logs

Transaction ID	Security solutions	Detailed scenarios
T1	DRM	DRM Document Authorization Request
		Download DRM document
	DLP	Image capture
T2	DRM	DRM Document Authorization Request
		Download DRM document
T3	DRM	Download DRM document
	DLP	Image capture
	Mail management system	Include confidential information in mail

Analyzing the correlation between the 'DRM' item set and the 'DRM document authorization request' item yields that the transaction including the 'DRM' item set and the 'DRM document authorization request' item set appears twice in the entire transaction, so we can see that the value of support is 2/3, and since the minimum support value is 0.5, it can be judged that both sets of items with the support value of 0.66 are instance of data leakage behavior satisfying the minimum support value. After that, each item set satisfying the minimum support value is defined as a frequent item set in a group, such as {'DRM', 'DRM document authorization request'}, and the size is increased. When a frequent item set is generated by calculating the support, the association analysis algorithm calculates the confidence value between the two item sets, then calculates the degree of association between them. In Table 3, a transaction including

'DRM' also includes a 'DRM document authorization request', so in this case, the confidence between two sets of items can be calculated to be 100%, and it is thus judged that they have a large correlation with each other. These frequent item sets satisfy both the minimum confidence and the minimum support, and are thus not removed in the algorithm process, but instead stored as a set of next association analysis candidate items. In the process of calculating support and confidence, all frequent item sets are generated and repeated until the support and confidence are analyzed, and finally, a frequent item set such as that shown in Table 4 is generated.

Table 4. Frequent item sets to which Apriori is completely applied

Frequency item sets
DRM Document Authorization Request, Download DRM document
DRM, Download DRM document
DRM, DRM Document Authorization Request
DRM, DRM Document Authorization Request, Download DRM document

Assuming a frequent item set is finally determined to be highly related, an attempt has been made to access the DRM document in common when the data is leaked, and it can be understood that the relevant document is downloaded after the authority of the DRM document is released. A frequent item set satisfying both the minimum reliability and support level is defined as an association behavior scenario in order to determine an insider's leaking of data, and as the scenario is composed of behaviors that can be exhibited for each security solution, in this case, 'DRM document authority request', 'DRM document download' security log information is defined as a 'DRM' association behavior scenario, so it becomes a data set for drawing a graph to be input to the combined product neural network afterwards.

3.3 Data Leakage Definition unit

3.3.1 Generating scenario graphs for leakage detection

This section introduces the method of judging data leakage by analyzing the security log collected from insiders based on the leakage detection scenarios defined in Section 3.2. The security log collected from the insiders is visualized in the form of a graph. This graph shows the frequency of occurrence of the security log indicating each data leakage behavior per hour, and a graph is collected for each leak detection scenario in order to create an image to be input to the CNN.

For example, in the case of the default behavior scenario of 'harmful site access', the rule is composed of the set of data leakage behaviors, such as 'the number of times of connection to harmful sites', 'the number of times of blocking malignant codes of traffic', and 'the number of times of unauthorized file downloads'. In the system, a graph is created based on the number of occurrences of the security log corresponding to each data leakage behavior, and another graph is created that shows the 'access to harmful site' leakage detection scenario by combining the occurrence frequency graph images of the security log corresponding to the behaviors of 'the number of times of connection to a harmful site's malicious code', 'the rate of increase of traffic', and 'the number of times of unauthorized file downloads', among the generated graphs. At this time, in order to distinguish each security log graph, the graphs are created with different background colors.

The leakage detection scenario representing each graph becomes a class number for discrimination in a CNN. In this case, the class number is differently judged for the normal and abnormal insider so as to distinguish between a normal insider and abnormal insider through the CNN. The administrator can understand the type of security solution detected with the behavior path and leakage behavior by judging whether the graph image input through the classification result output from the CNN shows a certain behavior scenario or a high degree of similarity with the graph of the abnormal insider.

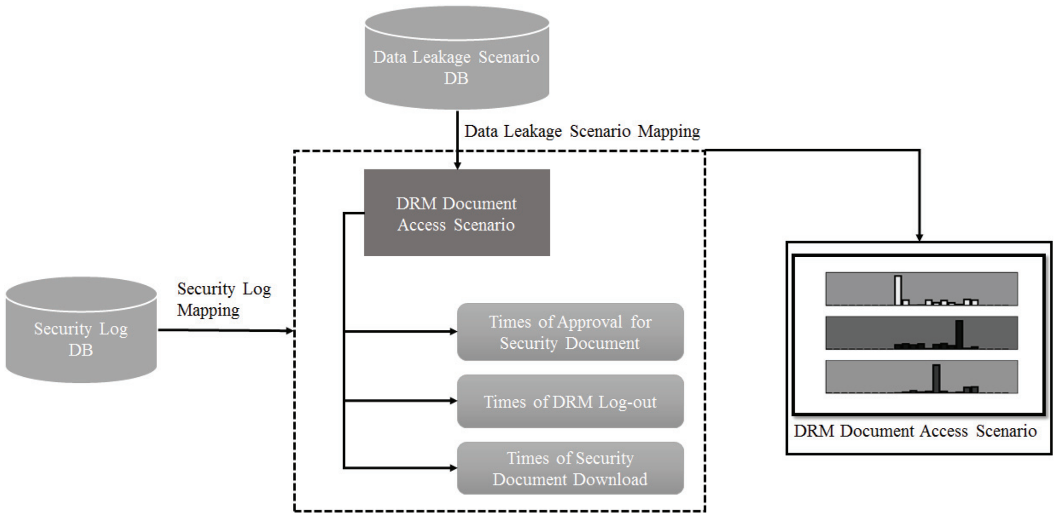
Data Leakage Scenario DB
Data Leakage Scenario Mapping
DRM Document Access Scenario
Security Log DB
Security Log Mapping
Times of Approval for Security Document
Times of DRM Log-out
Times of Security Document Download
DRM Document Access Scenario

Fig. 2. Process of drawing a graph for the leakage detection scenario.

Fig. 2 shows the process of graphically creating a DRM document access scenario. First, the DRM document scenario is extracted from the defined data leaking scenario. After extracting the list of security logs according to the DRM document scenario and creating a graph according to each scenario, the DRM document access scenario graph shown in Fig. 2 is created. Each data leakage scenario graph is entered into the CNN and used to determine whether data is being leaked by each internal employee.

The graph created based on the insider's security log is image data to be input to CNN, and the data leakage scenario representing graph is defined as a label for discriminating the data. Using the trained data and labels, it is possible to find that the graph of the insiders shows a similar trend to the graph of the abnormal employee. Using the discriminating result, we detect the type of security solution in which the action path/category leaked the data, based on the employee's information with the determined labels.

3.3.2 Convolutional neural network

In this paper, our purpose is to identify insiders suspected of leaking data by analyzing the behavior scenario graph according to insider's behaviors. Since the abnormal insider is determined by analyzing the color of the graph and the slope of the graph indicating the trend of the security log in this system, it identifies the insider who shows high similarity with the behavior log pattern of an insider who wants to leak data from the CNN, showing high performance in image analysis.

CNN is a kind of deep learning algorithm that shows excellent performance in image recognition. In the past, the feature extraction algorithm, such as sift and hog, has divided the image into pixels of a certain size, and recognized the images based on the features extracted according to the slope and brightness of each pixel. However, since the existing image feature extraction algorithm only extracts low-level features, spatial information of the three-dimensional image data can be lost. By contrast, the CNN learns the optimal feature map through the convolution layer and the pooling layer and learns the image more efficiently through the high-level feature extracted through the feature map.

CNN is a feed-forward neural network that is mainly applied to image recognition. It is composed of a convolution layer and a pooling layer for extracting features as well as a fully connected layer and softmax classifier for learning extracted features. The CNN aims to use the spatial information represented by the pixels of the image, and also learns and classifies the images through the feature extracted while moving by the pixel area with the fixed feature mask.

The convolution layer is the feature extraction process. The convolution layer applies a kernel with a certain size in order to input the image. The kernel aims to determine whether or not the feature is in the input image, and generally has a small height/width size, such as 5 or 3. Upon completion of the convolution operation, a feature map is extracted that shows the characteristics of the input image.

For the convolution operation, we denote feature maps $x^1..x^n$ of the original image. This entails x^{n-1} being a feature map derived from the previous layer, where w is a weight kernel for performing a convolution product operation and θ is a bias term. The convolution layer is operated by:

$$x^n = f(\sum w^n x^{n-1} + \theta^n) \tag{3}$$

where f is an activation function; we apply the ReLU operation to f in order to obtain a nonlinearity feature map. The convolution operation refers to a layer that performs an inner product operation by shifting a kernel for extracting a feature from a feature map derived from a previous map, in order to obtain a feature map x^n in the current layer.

The pooling layer controls the amount of computation by reducing the size of the feature map output from the convolution layer. We use the max pooling operation to select the maximum value and reduce the feature map of the image. Max pooling is formulated as follows:

$$x^n = f(max(x^{n-1})) \tag{4}$$

That is, the max pooling layer is a layer that only extracts the maximum value within a specific region, when a feature map x^{n-1} to reduce the size output from the convolution layer is input.

The feature map successfully extracted from the convolution layer and pooling layer is input to the fully connected neural network, then finally used to classify the data through the softmax classifier. The softmax classifier computes the difference between the distribution of the prediction label and the distribution of the real data label at CNN, and as this difference becomes smaller, the classifier outputs a higher value. The softmax classifier that shows the probability that the image is the *j*th label can be represented by the following equation:

$$softmax(j) = \frac{e^{xj}}{\sum e^{xi}} \tag{5}$$

As a result, an output vector having the size of the number of categories of the image to be classified is generated, and the softmax classifier outputs the data having the highest probability among them.

3.3.3 Adapting convolutional neural network to leakage detection

In order to detect an abnormal insider, we analyze the security log with the trend graph. However, instead of only analyzing the occurrence of the security log represented by the graph, we generate graphs by scenarios that express the log generation trend, the insider behavior represented by each log, and the time of abnormal behavior, so as to judge the behavior of the insider more efficiently.

The graph expressed according to the data leakage scenario is input to the CNN for training purposes. In this case, the graph about the security log trends of each insider is defined as the data to be trained, while the type of scenario that represents the category of the graph is defined by the class label. When there are *N* types of detection scenarios defined in the system, the graphs to be learned in the CNN have $G_1 \ldots G_N$ graphs for each insider. In addition, as the purpose of the system is to identify the insider who wants to leak data, if there are n number of graphs, the number of class labels is *2N*, consisting of the number of normal insider's graph *N* and the number of abnormal insider's graph *N*. We set the size of the kernel's width/height so as to find the characteristics of the image to be 5 in both cases, and set the degree of movement of the kernel in the convolution layer and degree of reducing the size of the feature map in the pooling layer to be 2 in both cases.

Table 5. Input, output, and procedure of CNN for leakage detection

Input	Data Leakage Scenario Graph Set *G*
Output	Determined Class label *L*
Procedure	Step 1. The kernel with height and width 5 is embedded and the convolution operation in the original graph *G* is operated, and then moved by 2, and the result is converted into a nonlinear value by applying the activation function ReLU. Step 2. From the pooling layer, the maximal value is extracted from the area of each feature map by the max pooling operation. Since the size of the feature map is set to a size of 2 so as to reduce the size of the feature map, it is reduced to 1/2 the size of the previous feature map. Step 3. The feature map extracted through two repetitions of the convolution layer and the pooling layer is flattened to transform into a form for inputting the form of the output data into the fully connected neural network. Step 4. When the data is delivered to the softmax classifier through a fully connected neural network, the softmax classifier outputs the data leak scenario class label with the highest probability in the overall data leakage scenario graph list.

We set the structure of the CNN so that the convolution layer and the pooling layer alternate twice to extract the feature map, and the ReLU function is used as the activation function. In addition, fully-connected layers and a softmax classifier were placed next to the last pooling layer in order to classify the images, and it was judged to which class label the input image showed the highest similarity. Let the number of scenarios of the normal and abnormal insider be 36, then the CNN algorithm can be applied as shown in Table 5, and the image feature selection and image classification layer have the structures shown in Tables 6 and 7, respectively.

According to the results of the CNN identification, it is possible for the administrator to identify the purpose of the behavioral path and action through which the data is leaked through the time at which the data is shown in the graph of the insider, which represents a graph similar to the data leakage insider.

Table 6. Image feature extraction layer structure

Layer name	Special feature extraction layer			
	Conv2D 1	MaxPool2D 1	Conv2D 2	MaxPool2D 2
#hidden units	64	64	64	64
Stride structure	1×1	2×2	1×1	2×2
Filter	5×5	-	5×5	-

Table 7. Image classification layer structure

Layer name	Image classification layer		
	FC 1	FC2	Softmax
#hidden units	384	192	36
Stride structure	-	-	-
Filter	-	-	-

4. Evaluation of Data Leakage Detection Accuracy by Experiment

We used 80,000 behavior log data points to assess the performance of the proposed system. We used five types of security solutions to collect security logs: network security, system security, content security, authentication security, and security management. In addition, we collected a security log corresponding to 28 action lists that included various actions of ‘media transmission blocking’, ‘excessive session connection’, and ‘IP access blocking’ that could be collected by each security solution.

Among the total collected security log lists, 60,000 security log sets were configured for training, and 20,000 security log sets were configured for testing. The performance test was performed for the case in which the discrimination scenario proposed in this paper is additionally constructed, and only the discrimination scenario stored in the existing system was applied, while the Apriori algorithm was not applied [17]. The performance test was conducted by plotting scenarios for each case and inputting them into a CNN, for the purpose of comparing whether CNN is properly displaying abnormal graphs, and the kinds of behavior scenarios represented by the graph. We used the accuracy, precision, recall, and f-measure as metrics to evaluate the performance. Precision represents the accuracy of the detected data and indicates whether the graph is actually an abnormal insider’s graph when our system classifies it as an abnormal insider’s graph. Recall represents a metric that measures how many outlier graphs are detected in data leakage detection scenario graphs of a real abnormal insider. F-measure is a measure for the performance of precision and recall, which can be calculated as a harmonic mean of precision and recall. Using these metrics allows us to understand how accurately the data flow path can be detected. The metrics have the following formulas:

$$accuracy = \frac{True\ detections + False\ detections}{total\ number\ of\ \text{dataset}} \tag{6}$$

$$recall = \frac{\text{detected}\ True}{total\ number\ of\ existing\ True} \tag{7}$$

$$precision = \frac{True\ detections}{whole\ detections\ of\ an\ \text{algorithm}} \tag{8}$$

$$f - measure = 2 * \frac{precision*recall}{precision+recall} \tag{9}$$

Table 8 shows the metric values of the scenario-reinforced system and the scenario-untreated system using the Apriori algorithm.

Table 8. Comparison of detection performance using accuracy, recall, precision, and f-measure

	Accuracy (%)	Recall (%)	Precision (%)	f-measure (%)
Complement scenario list with Apriori	97	97	98	97
Do not complement scenario list	95	97	95	96

The performance comparison results shown in Table 8 show that the system that has supplemented the detection scenarios with the Apriori algorithm more accurately categorized graphs with a leakage path of an abnormal insider with a higher ratio of metrics. By contrast, in the case that the scenario is not reinforced through the Apriori algorithm, if the data is leaked to a route that has not been added to the existing scenario, the system cannot detect it correctly, and the accuracy and precision ratio are lower than when the Apriori algorithm is used.

In addition, the degree to which the application of the Apriori algorithm affects the discrimination performance of data leakage can be grasped through a hypothetical scenario. Table 9 shows whether the data leakage paths of each system in the hypothetical scenario can be detected.

Table 9. Comparison of data leak detection performance by each abnormal scenario

Scenario	Behavior	The proposed system	Security solution not applying Apriori algorithms
Normal	Work normally during business hours	O	O
Abnormal 1	Access and download data that is not relevant to business during business hours	O	O
Abnormal 2	Request access to corporate data outside business hours and download	O	O
Abnormal 3	After downloading the data during business hours, change the format of the file and leaks.	O	X
Abnormal 4	In the case of non-business hours, the downloaded data is leaked out via e-mail	O	X

Judging by the results shown in Table 9, the security solution for which Apriori algorithms was not applied had a problem in that malicious access could be detected only when the history of each insider directly requested data. By contrast, the proposed system collects the history of the security log list that the insider has performed in the enterprise, and detects the data leakage, making it possible to detect the behavior, such as sending the data through e-mail or leaking to the outside via USB.

In addition, for the security solution for which Apriori algorithm was not applied, it is only possible to detect malicious access from the learned behavior in the abnormal behavior database. However, in the proposed system, by defining scenarios and detecting whether data has been leaked, it is possible to more flexibly judge whether data has been leaked.

According to the performance experiment, in the case in which the data leakage detection scenario is supplemented with Apriori, it is possible to determine whether data leakage is more accurate than when no action detection rule is stored in the system. Since the graph used to determine whether or not leakage occurs is created by aggregating actions by scenario, it is possible to identify the leakage route at once by analyzing the graph that is determined to be a problem, and it is known that the system can cope with data leakage more effectively than the existing data spill determination system.

5. Conclusion

In order to prevent data leakage by insiders, companies use DLP solutions and DRM solutions, and identify data leakage paths through SIEM, a security management system. However, in order to detect data leakage, it is necessary to analyze a large amount of security logs, and when data is leaked using a new method, it is difficult to properly determine whether data leakage has occurred.

In this study, in order to identify more patterns of data leakage, the leakage detection scenarios for detecting malicious behavior for data leakage include not only the leakage detection scenarios defined by the administrator, but also the behavior by extracting only the malicious behavior from the data leakage accident, and applying the Apriori algorithm and extracting the leakage detection scenarios. Therefore, it is possible to detect data leakage more flexibly than existing data leakage detection systems.

Further, in this study, malicious behavior is detected based on the leakage detection scenario of the data leakage graph, which shows the business behavior of insiders using CNN. Through these graphs, it is possible to grasp the outflow time and the data leak behaviors, and it is also possible to accurately and clearly analyze the large-capacity security log list, because the similarity of malicious insiders is compared through the deep learning algorithm.

In order to determine the performance of the proposed system, the leakage detection scenarios, which do not correspond to the leakage detection scenarios of data leakage, and test whether they can be detected through the associated leakage detection scenarios, is performed, and the amount of data leakage and accuracy of discrimination according to the number of behavior times done to leak the data were measured. As a result of the experiment, it was confirmed that the data can be discriminated, even if the data is leaked through a path that was not previously defined. In future research, we plan to apply association relationship analysis to small security log data by improving the performance of the Apriori algorithm.

Acknowledgement

This paper was supported by the Small and Medium Business Administration's Research Village Project (No. S2635358).

References

[1] InfoWatch, "Global data leakage report," 2017; https://infowatch.com/report2017#.

[2] T. Wuchner and A. Pretschner, "Data loss prevention based on data-driven usage control," in *Proceedings of 2012 IEEE 23rd International Symposium on Software Reliability Engineering*, Dallas, TX, 2012, pp. 151-160.

[3] W. Ku and C. H. Chi, "Survey on the technological aspects of digital rights management," in *Information Security*. Heidelberg: Springer, 2004, pp. 391-403.

[4] M. Afzaal, C. Di Sarno, L. Coppolino, S. DAntonio, and L. Romano, "A resilient architecture for forensic storage of events in critical infrastructures," in *Proceedings of 2012 IEEE 14th International Symposium on High-Assurance Systems Engineering*, Omaha, NE, 2012, pp. 48-55.
[5] M. I. Salam, W. C. Yau, J. J. Chin, S. H. Heng, H. C. Ling, R. C. Phan, G. S. Poh, S. U. Tan, and W. S. Yap, "Implementation of searchable symmetric encryption for privacy-preserving keyword search on cloud storage," *Human-centric Computing and Information Sciences*, vol. 5, article no. 19, 2015.
[6] W. Zhu and C. Lee, "A security protection framework for cloud computing," *Journal of Information Processing Systems*, vol. 12, no. 3, pp. 538-547, 2016.
[7] N. S. Houari and N. Taghezout, "A novel approach for integrating security in business rules modeling using agents and an encryption algorithm," *Journal of Information Processing Systems*, vol. 12, no. 4, pp. 688-710, 2016.
[8] C. Borgelt and R. Kruse, "Induction of association rules: apriori implementation," in *Compstat*. Heidelberg: Physica, 2002, pp. 395-400.
[9] D. C. Ciresan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber, "Flexible, high performance convolutional neural networks for image classification," in *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, Barcelona, Spain, 2011, pp. 1237-1242.
[10] A. S. Ashoor and S. Gore, "Difference between intrusion detection system (IDS) and intrusion prevention system (IPS),"in *Advances in Network Security and Applications*. Heidelberg: Springer, 2011, pp. 497-501.
[11] L. C. Wuu, C. H. Hung, and S. F. Chen, "Building intrusion pattern miner for Snort network intrusion detection system," *Journal of Systems and Software*, vol. 80, no. 10, pp. 1699-1715, 2007.
[12] S. H. Oh and W. S. Lee, "Network anomaly detection based on association among packets," *Journal of the Korea Institute of Information Security and Cryptology*, vol. 12, no. 5, pp. 63-73, 2002.
[13] K. Julisch, "Clustering intrusion detection alarms to support root cause analysis," *ACM Transactions on Information and System Security (TISSEC)*, vol. 6, no. 4, pp. 443-471, 2003.
[14] O. Brdiczka, J. Liu, B. Price, J. Shen, A. Patil, R. Chow, E. Bart, and N. Ducheneaut, "Proactive insider threat detection through graph learning and psychological context," in *Proceedings of 2012 IEEE Symposium on Security and Privacy Workshops*, San Francisco, CA, 2012, pp. 142-149.
[15] A. Das, D. Nguyen, J. Zambreno, G. Memik, and A. Choudhary, "An FPGA-based network intrusion detection architecture," *IEEE Transactions on Information Forensics and Security*, vol. 3, no. 1, pp. 118-132, 2008.
[16] J. Kim, N. Shin, S. Y. Jo, and S. H. Kim, "Method of intrusion detection using deep neural network," in *Proceedings of 2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*, Jeju, Korea, 2017, pp. 313-316.
[17] M. J. Seo, H. J. Shin, M. H. Kim, and J. H. Park, "Internal information leak detection system using times-series graph," in *Proceedings of the 2017 Spring Conference of the KIPS*, Jeju, Korea, 2017, pp. 769-770.

Min-Ji Seo https://orcid.org/0000-0002-7564-9264

She obtained her Bachelor's degree of Computer Science and Engineering from Soongsil University, Seoul, Korea, in 2015. She is a candidate for a PhD degree in the Software Convergence Department of Soongsil University, Seoul, Korea. Her research interests include System software.

Myung-Ho Kim https://orcid.org/0000-0002-1933-7987

He is a professor in the Department of Software Convergence in Soongsil University, Seoul, Korea. He obtained his Ph.D. from Pohang University of Science and Technology, Pohang, Korea, in 1995. His research interests are information security and system software.