

Article

Two-Phase Industrial Control System Anomaly Detection Using Communication Patterns and Deep Learning

Sungjin Kim ¹, Wooyeon Jo ², Hyunjin Kim ³, Seokmin Choi ⁴, Da-I Jung ⁴, Hyeonho Choi ⁴ and Taeshik Shon ^{3,5,*}

¹ OT Security Group, Samsung SDS, Seoul 03922, Republic of Korea; ksjskyblue@gmail.com

² SAFE Lab, Virginia Commonwealth University, ERB 3339 401 W Main St, Richmond, VA 23284, USA; jow@vcu.edu

³ Department of Computer Engineering, Ajou University, Suwon 16499, Republic of Korea; hyunjin.infosec@gmail.com

⁴ Information Security Team, Korea Power Exchange, Naju 58322, Republic of Korea; mini0827@kpx.or.kr (S.C.); jung@kpx.or.kr (D.-I.J.); hhchoi@kpx.or.kr (H.C.)

⁵ Department of Cybersecurity, Ajou University, Suwon 16499, Republic of Korea

* Correspondence: tsshon@ajou.ac.kr

Abstract: Several cases of Industrial Internet of Things (IIoT) attacks with zero-day vulnerabilities have been reported. To prevent these attacks, it is necessary to apply an abnormal behavior detection method; however, there are three main problems that make it hard. First, there are various industrial communication protocols. Instead of IT environments, many unstandardized protocols, which are usually defined by vendors, are used. Second, legacy devices are commonly used, not only EOS (End-of-service), but also EoL (End-of-Life). And last, the analysis of collected data is necessary for defining normal behavior. This behavior should be separately defined in each IIoT. Therefore, it is difficult to apply abnormal behavior detection in environments where economic and human investment is difficult. To solve these problems, we propose a deep learning based abnormal behavior detection technique that utilizes IIoT communication patterns. The proposed method uses a deep learning technique to train periodic data acquisition sequences, which is one of the common characteristics of IIoT. The trained model determined the sequence of packet is normal. The proposed technique can be applied without an additional analysis. The proposed method is expected to prevent security threats by proactively detecting cyberattacks. To verify the proposed method, a dataset was collected from the Korea Electric Power Control System. The model that defines normal behavior based on the application layer exhibits an accuracy of 79.6%. The other model, defining normal behavior based on the transport layer, has an accuracy of 80.9%. In these two models, most false positives and false negatives only occur when the abnormal packet is in a sequence.

Keywords: industrial IoT; industrial 4.0; anomaly detection; industrial control system; security with deep learning



Citation: Kim, S.; Jo, W.; Kim, H.; Choi, S.; Jung, D.-I.; Choi, H.; Shon, T. Two-Phase Industrial Control System Anomaly Detection Using Communication Patterns and Deep Learning. *Electronics* **2024**, *13*, 1520. <https://doi.org/10.3390/electronics13081520>

Academic Editors: Nadia Kanwal, Mohammad Samar Ansari, Yuhang Ye and Brian Lee

Received: 19 February 2024

Revised: 7 April 2024

Accepted: 9 April 2024

Published: 17 April 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The manufacturing industry has changed significantly. In Information Technology, The Internet of things (IoT) era was an advent. With the pace of it, the industrial control system (ICS) has also been developing into industrial IoT (IIoT) from factory automation. Standardization of the ICS, which represents the fourth industrial revolution, such as RAMI 4.0, is actively progressing simultaneously. With these changes, the ICS has been changed to an open structure which increases the possibility of unconsidered cyberattacks.

Most cyberattacks are low-level and indiscriminate attacks in a normal IT environment. These can be easily prevented using signature-based countermeasures; however, in an ICS environment, complex cyberattacks are performed over a longer duration by specialized hacker groups. The aim of these hacker groups is serious physical and financial damage

to the target system even if it takes a lot of cost and effort. Therefore, it is hard to prevent. Since the Stuxnet attack against Iran's nuclear power plants in 2010, several APT attacks using zero-day vulnerabilities have been reported [1]. The threat of targeted ransomware attacks on an ICS is increasing [2], including Colonial Pipeline ransomware attacks.

To respond to such advanced attacks, research on abnormal behavior detection is essential. To find abnormal behavior in traffic, normal behavior should be defined. Anomaly detection has a lower detection rate than signature-based detection methods; however, it can detect a zero-day attack which has no signature. Anomaly detection is essential for preventing zero-day attacks, a important issue in ICS. However, huge effort is required to analyze the specific characteristics of the system to define normal behavior for anomaly detection. It is possible to develop abnormal behavior detection equipment through sufficient economic investment in a large-scale ICS, even if it takes a long time to analyze the characteristics of each system. However, it is almost impossible to apply it to a small-scale factory because of an economic problem.

To solve this problem, we propose an ICS network anomaly detection method based on acquired data in [3]. This method focuses on detecting an anomaly in the data area. However, this method has a shortcoming in that it is difficult to detect false data injection attacks. It also requires a long preprocessing time to parse entire protocol fields. Therefore, in this paper, we propose a novel anomaly detection method based on communication patterns to overcome the shortcomings of our previous study. The proposed technique automatically calculates the data acquisition cycle of the target ICS and uses it to define normal behavior. Because this method uses general ICS characteristics, it is applicable to most ICSs. An RNN based long short-term memory (LSTM) deep learning algorithm is trained with the automatically analyzed data acquisition sequence to detect anomalies. The main contributions of this study are as follows:

- As the proposed method defines normal behavior based on the general characteristic of ICS communication being periodic, it is not dependent on communication protocols and can be applied to various ICS environments.
- Defining normal behavior is performed automatically; it can be applied without any knowledge of the domain, thereby reducing the cost of analysis and preventing a wrong normal behavior definition because of human error.
- The proposed technique is verified using an attack dataset based on the analysis of the ICS Cyber Kill Chain.

The proposed scheme uses the periodicity of communication, which is a general characteristic of ICS communication, to define normal behavior. Therefore, it can be applied to all ICSs where periodic communication is performed, regardless of the communication protocol. This is discussed in detail in Section 3. It is possible to apply the proposed method even with insufficient domain knowledge because it can automatically analyze patterns in communication traffic. An advantage is the prevention of human errors in defining normal behavior. The communication pattern analysis is presented in detail in Section 4. To verify the proposed method, we used to collect traffic from an operating Korea Electric Power Control System as our dataset. The attack dataset was based on an analysis of each step in the ICS Cyber Kill Chain. The network traffic observed in each attack step was derived and added to the dataset to create an attack dataset. This is discussed in detail in Section 5.

The remainder of this paper is structured as follows. Studies on intrusion detection methods in ICSs are discussed in Section 2. The characteristics of an ICS network are mentioned in Section 3. Based on these characteristics, the proposed method is explained in Section 4. The proposed method is evaluated in Section 5, and the results are discussed in Section 6. Finally, we present the conclusions and future work directions in Section 7.

2. Related Works

To prevent cyberattacks on ICSs, many researchers have attempted to develop various countermeasures for ICSs, such as encryption and authentication in field networks [4–7]. However, due to commercial problems, it is difficult to replace legacy equipment which

does not support secure communication using encryption and authentication [8]. Most legacy systems operate with non-secure ICS communication protocols that do not have security measures such as encryption and authentication. Therefore, an intrusion detection system for ICSs is required. Research on anomaly detection methods is actively conducted to prevent zero-day attacks without frequently updating the attack signature database in ICSs [9].

Studies on anomaly detection for the ICS are classified with two concepts. These two concepts are classified based on defining normal behavior. One uses data patterns to detect false data injection (FDI), such as the injection of control commands or the fabrication of measured data. This attack causes unpredictable device operations. The other focuses on detecting abnormal network behavior, such as the appearance of a hacked node using characteristics of network traffic.

Current studies to prevent FDI can be divided into two methods, machine learning-based and invariant rule mining-based approaches. The machine learning-based studies in this field are [3,10–12]. In [10], an anomaly detection method that analyzes the noise pattern of sensors was proposed. This method is highly accurate because it relies on the hardware characteristics of the sensors. In [11], an autoencoder based anomaly detection method was proposed. It groups the related network features and learns a normal pattern for each group. Autoencoders are suitable for anomaly detection because they can be learned without a labeled dataset, and they can reduce a high-dimensional feature space to a lower dimension. In [12], a deep learning-based anomaly detection method with association rule mining was proposed. In a previous study, the author of [3] demonstrated that an autoencoder can learn the data changing patterns and the relationships between data by training data in the same operation and presented an autoencoder based anomaly detection method using these characteristics. Some studies have defined rules based on data relationships [13–15]. The method proposed in [13] detects anomalies by analyzing data relationships. It has a low false positive because it relies on the physical characteristics of the measured data. However, it cannot detect data that has no analyzed invariant rule. These studies have only focused on transmitted data. Therefore, although they are effective in detecting false measured data and control commands, they cannot detect other attacks.

Other methods for anomaly detection in ICSs use network traffic characteristics. These detection methods are based on learning the major features of a packet, the sequential communication pattern with data mining, and so on. References [16,17] tried to train learning models with the major fields of a packet. In [16], an anomaly detection method that utilizes the relationships between the major fields of a packet was proposed. The proposed method extracts the major fields of each protocol layer and learns the relation using a convolutional neural network (CNN). A method that converts packets into two-dimensional images and classifies attacks and the normal using a CNN algorithm is proposed in [17]. This method can be applied to several domains; however, an attack dataset is required because CNNs involve a supervised learning algorithm. In [18], the ICS communication patterns were analyzed, and a detection method based on sequential communication patterns was proposed. In [19], an ensemble method to detect anomaly in CPS was proposed. The model is composed of 5 different supervised learning models, Logistic Regression, Naïve Bayes, SVM (Support Vector Machine), KNN (K-nearest Neighbor), and MLP (Multi-Level Perceptron). To apply these studies, it is necessary to select proper features by analyzing the network traffic and training the learning models because it has high dependency on the target system. Owing to these shortcomings, a long-time analysis of each system is required for applying these studies to several systems. This requires a considerable amount of time and effort.

To overcome these problems, a novel method that can be applied without any additional analysis of each network is needed. To develop such a method, we analyze the network communication pattern of an ICS. This is described in Section 3.

3. Classification of ICS Network Communication

An ICS is constructed with several field and control devices. The measured values are transmitted to the control devices from the field devices, and the control devices change the operation of the field devices using control commands. Most of the traffic is data acquired from the field devices for monitoring. The traffic for control is comparatively rare. Therefore, it is possible to predict incoming network traffic because ICS network communication is static, which is impossible in a general IT network. Network communication in an ICS can be classified as shown in Figure 1.

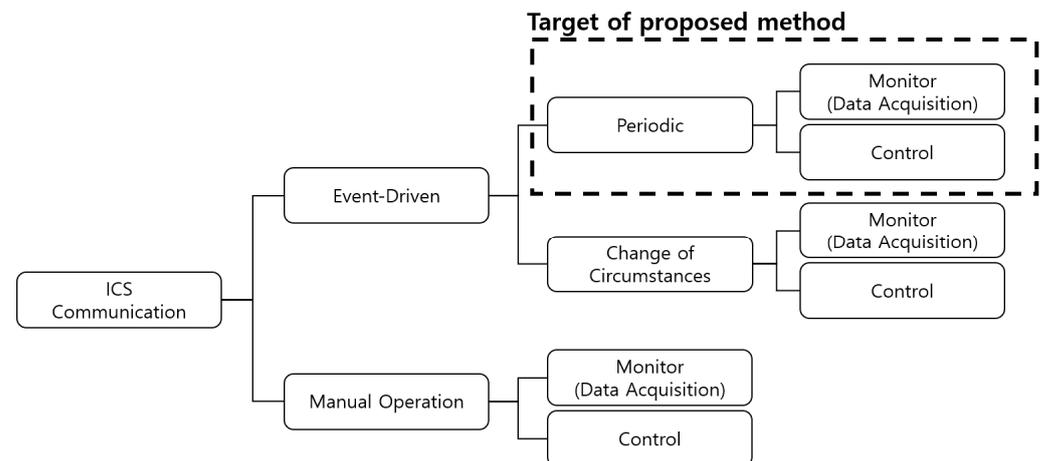


Figure 1. Target of the proposed anomaly detection method.

There are two main types of ICS network communications: that caused by a manual operation and that driven by an event. Event-driven communication is automatically performed under certain conditions without any administrator intervention. For supervision and control, two types of event-driven communications can be used together. Basically, the ICS is designed for automation with minimal administrator intervention. Manual operation occurs in unusual situations, such as faulty devices. In general, most of the communications in an ICS are periodic transmissions, whereas the others are the administrator's manual operations or value fluctuations [20,21]. ICS communication traffic has a static pattern because it is designed to perform only supervision and control. As periodic data acquisition is a general ICS characteristic, it can be used to define the normal behavior of most systems.

Some studies have used periodic communication to define normal behavior [22,23]. In [2], a communication profile was generated by using the difference in arrival times and by training a support vector machine for anomaly detection. This method can be used in various domains because it utilizes limited features (packet length and time interval); however, an SVM, one of the supervised learning methods, requires attack data for training. In [23], an automatic communication period analysis was proposed; however, it does not consider errors such as network delays. To overcome these shortcomings, we propose an anomaly detection method based on communication patterns that considers network delays.

4. Proposed Method

An ICS is composed of low-power devices. It is difficult to apply techniques that require high computing power. Deep learning technique requires high computing power for training, and the proposed technique is designed in two phases: a training phase that trains the deep learning model and an execution phase that detects anomalous behavior. Due to the two-phase design, training can be performed in devices with a high computing power, and the execution can be performed in low-power devices. In addition, it can prevent adversarial attacks that cause the deep learning model to be abnormal by contaminating the training data [24]. Details of the two phases are presented in Figure 2. Deep learning technique has several known problems, e.g., it requires a large amount of data for

training model, fails to distinguish causality from correlations, and always assumes a stable environment. Despite these problems, deep learning to ICS abnormal behavior detection can be used. At first, the training data is easy to acquire in ICS networks in which a lot of communication occurs. The ICS network traffic is usually stable if there is no attack. The data transmitted in ICS has a relation with each other. Therefore, deep learning technique is possible to apply ICS abnormal behavior detection.

In the training phase, data preprocessing and communication pattern analysis are performed before the training model. During preprocessing, traffic is separated based on the IP, packet parsing, grouping with function code, and target data object. After that, the analysis for periodic communication patterns is performed in Communication Pattern Analyzer. At the last of the training phase, the deep learning model is trained using only the packets that have performed periodic communication.

In the execution phase, preprocessed packet data are injected into the learned deep learning model, and abnormal behavior is detected based on the prediction of the deep learning model. The preprocessing is also conducted in the execution phase; however, the step for grouping packets is not performed. Instead, it generates a sequence of packets to be used as an input for the deep learning model. Subsequently, a packet sequence is injected into the deep learning model, and its result is compared with the actual data to determine whether the incoming packet is normal.

As shown in Figure 2, if the training phase and execution phase are distinguished, deep learning models can be trained in a high-performance environment. It is good for high performance where availability is important. The proposed method can be used in various domains owing to this advantage.

Because the proposed method trains only periodic data transmission, the learned deep learning model cannot predict manual operation and non-periodic control command packets. These packets are usually associated with an abnormal status in a manufacturing process. In other words, a false positive rarely occurs because manual operation and control command packets are scarcely observed in a stable state. The details of each phase are described below.

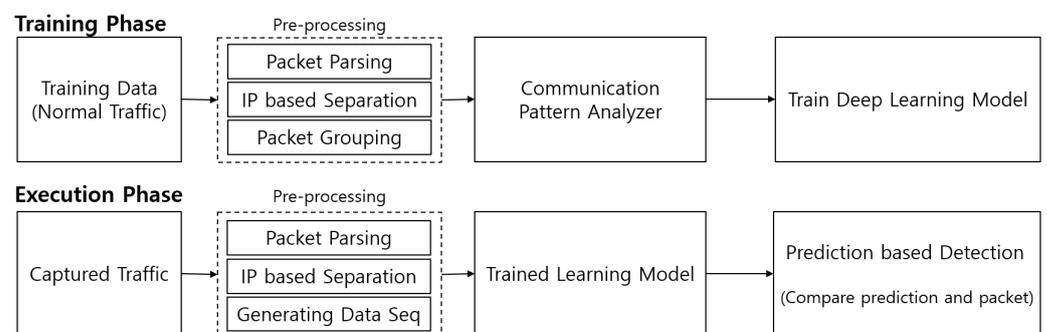


Figure 2. Two phases of the proposed ICS anomaly detection method.

In the training phase, the deep learning model is trained for the detection model. In this phase, data preprocessing and communication pattern analysis are performed to learn the periodic data. In the case of a single communication section, the packet sequence is not affected by the network delay. However, in the case of several communication sections, the packet sequence can be twisted owing to a network delay. To prevent this problem, packets are separated for each communication section, and the packet sequence of each communication section is used for training. Details of the training phase are presented in Figure 3.

First, data preprocessing is performed in the training phase for communication pattern analysis before training the deep learning model. In the preprocessing stage, the time, source IP, destination IP, function code, and target data object are parsed from the packet, and the communication section is separated based on the source IP and destination IP.

Then, grouping is performed based on the function code and data object for communication pattern analysis.

Communication pattern analysis is performed for each group after preprocessing. Then, a periodic data communication sequence of the communication section is generated by combining the periodic transmission sequence of each group. Two observations from the periodic data communication used in the periodicity analysis are as follows:

- **Observation 1.** If the dataset has a periodic communication, then none of the periods is smaller than the largest delta time.
- **Proof.** If not, there is a P , where P is a period, and $P < \text{largest delta time}$. Let the delta time, DT , be the largest delta time, composed of $T1$ and $T2$. If $T1$ follows period P , the packet should arrive at $T1 + P < T2$. If not, the packet that follows period P should arrive before $T1 + P$. These two cases are contradictory because there is no packet in $T1$ and $T2$. Therefore, the existence of a period P that is smaller than the largest delta time is impossible.
- **Observation 2.** The data acquisition sequence is not switched in the same communication section.
- **Proof.** Network delay is composed of processing delay, queuing delay, transmission delay, and propagation delay. These four delays have a similar effect on each packet because the ICS network is static. The propagation delay, processing delay, and transmission delay are always the same for all packets. The changes in queuing delays are very small because the target network is very static. Therefore, a jumbled data acquisition sequence does not exist in any single communication section.

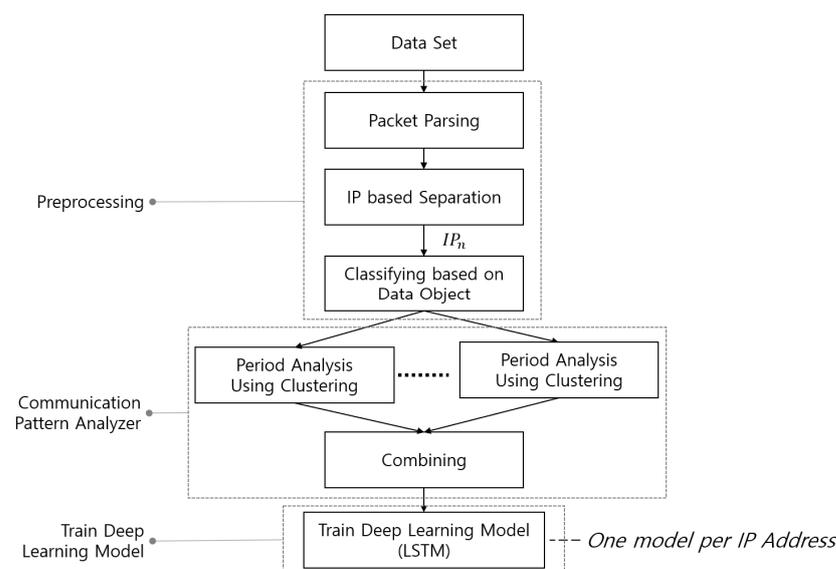


Figure 3. Procedure of the proposed ICS anomaly detection method—training phase.

The communication pattern analyzer uses the above two features for periodicity analysis. The detailed operation of the communication pattern analyzer includes four steps, as shown in Figure 4.

Figure 4 depicts the traffic acquiring a and b data objects over time. ① is the data acquisition details over time in a single communication section, generated via packet separation during preprocessing. Periodicity analysis is difficult because of the data coming in simultaneously. Therefore, grouping is performed based on the function code and data object in step 0. This clearly reveals the periodicity of each datum and helps in the periodicity analysis. As the example deals with only two data acquisitions, ②, which is divided into two parts, is derived. In step 1, from ② to ③, the periodicity of each data object is checked, and aperiodic data are removed. In step 2, a one-dimensional graph (④)

that displays only periodic data acquisition is derived by combining all the data objects of the 2-dimensional graph (③) from which the aperiodic transmission has been removed.

In step 1, from ② to ③, we assume the expected value to be the period of data acquisition and check whether the period of data acquisition is the expected value. In the case of data object b in Figure 4, d_{b1} is set as the expected period to check the periodicity because d_{b1} is longer than d_{b2} . Here, because d_{b1} , which is the maximum value of the simple delta time, includes delays from the network condition, we use the centroid of the cluster that contains d_{b1} . Therefore, the delta times are clustered, and the centroid of the cluster that contains the maximum delta time is inferred as the period of data acquisition. If data communication is periodic, this centroid should be one of the periods.

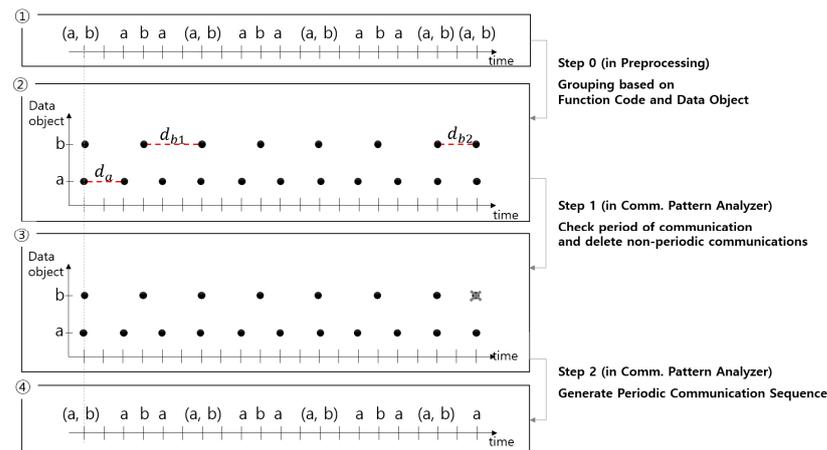


Figure 4. Four steps of communication pattern analyzer.

When it is determined that the traffic follows the expected period, the previous process is repeated to check whether another period exists. To prevent that a packet is checked with duplicate, the checked packet is skipped. After this process is complete, the remaining packets are excluded from the training dataset because periodicity is not observed. As this algorithm assumes that packets are captured for a sufficient period, a long-time packet capture is necessary if the data acquisition cycle is long.

Due to this analysis, the training dataset has only periodic data. Therefore, it is possible to train the data pattern using LSTM, a type of RNN based deep learning algorithm specialized in learning time-series data. It can be used for long sequence learning by solving the vanishing gradient problem of the Vanilla-RNN. Among the various structures of the LSTM model, we use a many-to-one LSTM model here. This model learns the relationship with the previous and next data and returns one output. In the case of a unidirectional LSTM, in which training is performed in one direction, there is a problem, namely that training is difficult when there is a data pattern that rarely exists. Therefore, for the learning of the proposed technique, in this study, the model is constructed with a bidirectional LSTM rather than a unidirectional LSTM. Figure 5 shows the many-to-one bidirectional LSTM used in the proposed method.

The many-to-one bidirectional LSTM performs a prediction by considering both data arriving before and after, as shown in Figure 5. This is advantageous when learning the pattern of a communication section in which communication with a short period and communication with a long period exist together. When the input sequence $(x_{t-2}, x_{t-1}, x_t, x_{t+1})$ is sequentially input to the model, y_t is output through the forward and backward layers. Because the model is trained such that y_t and x_t have the same value, an abnormality detection is possible by understanding whether y_t and x_t coincide in the execution phase.

In the execution phase, the trained LSTM model is used to determine whether the packet is normal. Packet parsing and packet separation are performed for each communication section. Subsequently, the sequence packets are generated for use as an input in the LSTM model. Unlike the training phase, this sequence includes all aperiodic commu-

nication. The LSTM model’s prediction and the actual packet are matched to distinguish between normal and abnormal packets. The operation of the execution phase is presented in Figure 6.

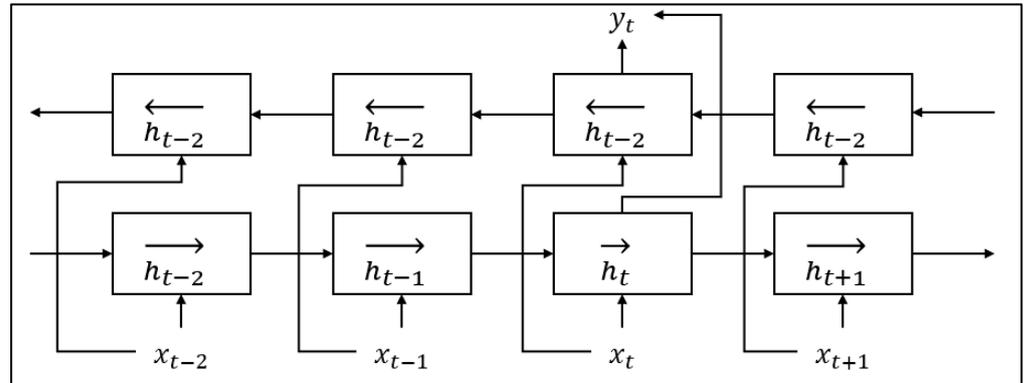


Figure 5. Many-to-one bidirectional LSTM.

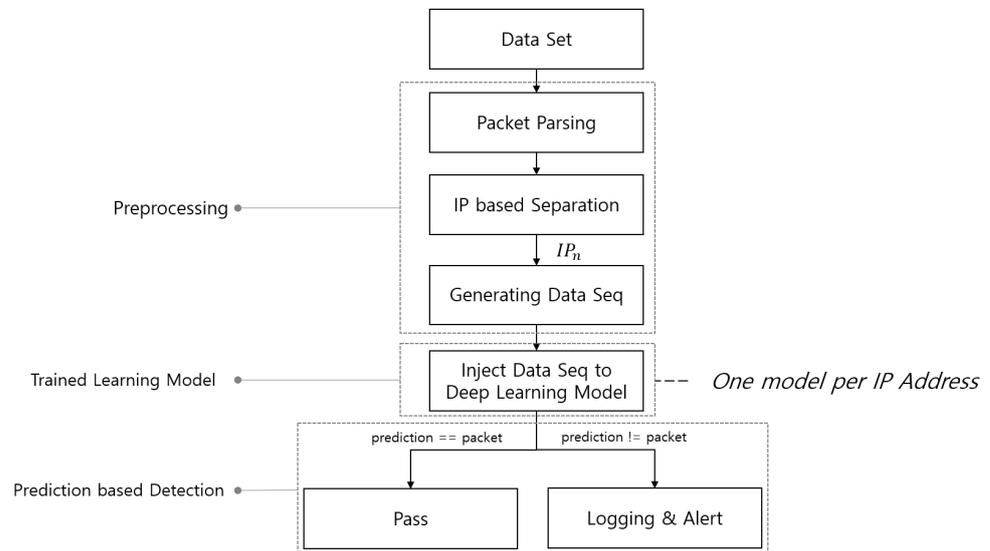


Figure 6. Procedure of proposed ICS anomaly detection method—execution phase.

The execution involves preprocessing for generating the input prediction through the deep learning model and the determination of abnormal behavior. When a normal data sequence is input into the deep learning model, the prediction and the actual packet is matched. Therefore, the abnormality criterion of the proposed method is whether the output of the deep learning model matches the actual data. If the outputs of the deep learning model and the packet are the same, then it is determined as normal. If not, it is considered abnormal.

Because the learning model trains only periodic transmissions, false positives are possible when the normal sequence involves a non-periodic transmission. However, as the proposed method targets the ICS where periodic transmission includes most of the network traffic, a false positive is rarely observed. If the network is highly periodic, a lower false positive rate can be achieved through retraining. Details of the retraining are discussed in Section 6.

5. Evaluation

5.1. Dataset

To verify the proposed anomaly detection method, a dataset was developed by collecting network traffic from the Korean Electric Power System. A total of 70% of the collected

data were used as a normal training dataset to train the detection model, and the remaining 30% were used to build a normal test dataset for testing. The attack dataset, which includes abnormal behaviors for verification of the model, is generated by changing the normal test dataset. In the case of the normal training dataset, the collected data were used without any changes because there was no abnormality during the network traffic collection period. The normal test dataset is used to measure the accuracy of the pattern analysis and check the false positive rate of the detection model. As there was no abnormality during the collection period, it was built using the remaining 30% of the traffic, as it is similar to the normal training dataset. The attack dataset is used to evaluate the abnormality detection capability of the proposed detection model. It was constructed by changing the normal test dataset. The details of the datasets are as follows:

- Normal training dataset: Used for LSTM learning of the proposed method (70% of collected traffic/42 GB)
- Normal test dataset: Used for pattern analysis accuracy and false positive analysis of the proposed technique (30% of collected traffic/18 GB)
- Attack dataset: Used to evaluate the anomaly detection ability of the proposed technique (Add attack to normal dataset/20 GB)

Network traffic collected over a week in the Korean Electric Power System was used to construct the normal training dataset and the normal test dataset. Approximately 60 GB of network traffic was collected over a week, which was confirmed as normal network traffic without any issues during the actual operation. The collected network traffic is composed of several periodic data exchanges between 15 sub-systems and one upper-level system, using the inter control center communications protocol (ICCP) as the communication protocol. Manufacturing message specification (MMS), which has multiple lower layers because of the use of OSI 7 layers, was used for the ICCP. Therefore, the ICCP packet can be fragmented in multiple layers. In the collected traffic, several TCP and COTP fragmentations were observed. Because of a number of similar communication sections, an experiment was conducted by selecting one representative communication section. The transmission details of the representative communication section are described below.

As shown in Table 1, there are seven communication patterns in the traffic in total. The PDU and service of the MMS are command sets provided by the protocol. It is similar to the function code in other protocols such as DNP3 and Modbus. Most of the collected traffic consists of periodic data acquisition using the Information Report service of the Unconfirmed PDU. The remaining collected traffic consists of the Identify service of the Confirmed Request PDU and the Confirmed Response PDU. While using the Information Report service of the Unconfirmed PDU, a considerable amount of fragmentation is observed because most of them send a large amount of data at once.

Table 1. Details of the data transmission of the representative communication section.

PDU	Service	Data Object	Period	Fragmentation
Unconfirmed	Information Report	Data #1	4	TCP, COTP
Unconfirmed	Information Report	Data #2	4	TCP, COTP
Unconfirmed	Information Report	Data #3	4	TCP, COTP
Unconfirmed	Information Report	Data #4	4	TCP, COTP
Unconfirmed	Information Report	Data #5	4	TCP, COTP
Unconfirmed	Information Report	Data #6	4	TCP, COTP
Confirmed Request	Identify	-	60	-

The attack dataset was constructed by changing the normal test dataset. An attack dataset is classified into four types according to the added abnormal behavior to the normal test dataset. The four types are abnormal command transmission, abnormal command transmission after dropping normal packets, an attack against network equipment (for example, denial of service (DoS)), and abnormal data access.

To generate actual abnormal behaviors, we analyzed the ICS Cyber Kill Chain, which is shown in Figure 7 [25]. As ICS operates as a closed network that is disconnected from the outside, direct penetration is difficult. According to the ICS Cyber Kill Chain, the attack targeting the ICS occurs in two stages. In Stage 1, the attacker attacks the vulnerable corporate network connected to the ICS network and gains access rights to the internal network of the ICS. Then, in Stage 2, the attacker directly attacks the ICS using these access rights. Because the proposed scheme in this study aims to detect attacks on the ICS internal network, the attack traffic is generated by focusing on Stage 2.

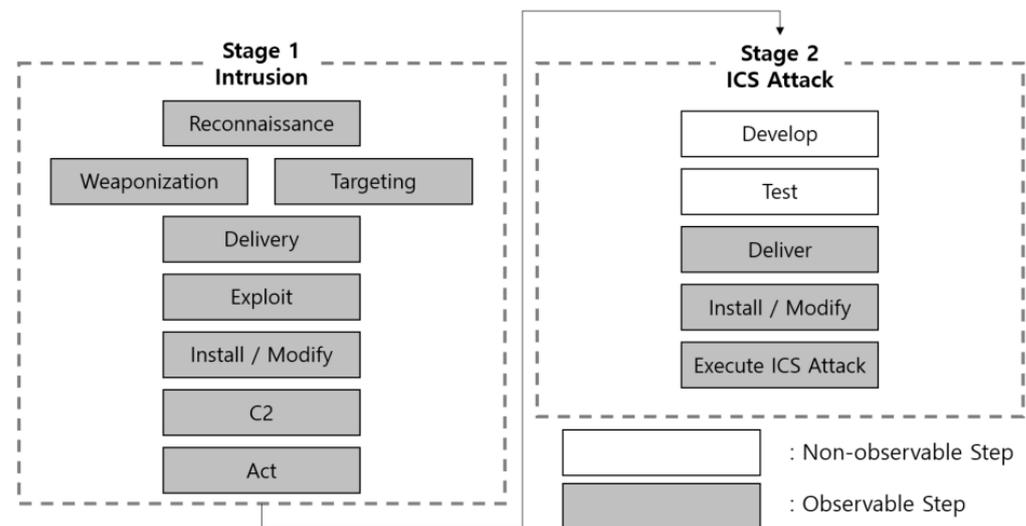


Figure 7. Lockheed Martin's ICS Cyber Kill Chain.

In Stage 2, attacks targeting the ICS internal network can be observed during the deliver, install/modify, and execute ICS attack steps. In the deliver step, an attack is performed through the attack path in Stage 1. In this step, the file transfer command of the communication protocol used by the target ICS can be utilized. The install/modify step is the process of installing the transferred file and changing the operation of the target system. In this process, commands, such as those for the control of the target device and execution of a specific process of the target device, can be utilized. The execute ICS attack step is the process of executing the already installed malicious code. The same install/modify commands can be used in this step. In the MMS Confirmed Request/Response PDU, services are used for the aforementioned attacks. The Obtain File service is related to file transfer. The Take Control and Relinquish Control services can control the device. The Start, Stop Resume, Rest, and Kill services are used to execute and terminate processes. When an attacker uses these services, detection is impossible with security solutions in general IT environments that do not support ICS protocols. Therefore, in this study, we focus on finding abnormal communication using an ICS communication protocol.

If a compromised node exists, an attack can be caused by transmitting abnormal control commands using the aforementioned services to damage the physical manufacturing process. In this case, other MMS commands, such as Read and Write, can be used. Therefore, we generate an attack dataset, which includes transmitting an abnormal file attack, executing an abnormal process attack, and accessing unauthorized data attacks using read and write operations. The attack traffic is composed of four types, as shown below:

- Attack Data #1: Inject unused function code
- Attack Data #2: Swap unused function code
- Attack Data #3: Jumble up the packet sequence
- Attack Data #4: Unauthorized data access

Attack Data #1 is an anomaly in which an unused function code is found in the training phase. It corresponds to an attack where the attacker attempts to install a malicious code

using the aforementioned services related to files and processes. Attack Data #2 is an anomaly in which a normal packet is dropped, and a packet containing a function code that is not used in the training phase is transmitted. This attack differs from Attack Data #1 in that an attacker intercepts a packet in the middle; in other words, the normal packet is dropped, and the malicious packet is sent. Attack Data #3 is an anomaly where the sequence is mixed differently from the training sequence. This is a situation in which a normal packet block or drop occurs. It can occur because of a service failure in network devices caused by attacks such as a distributed DoS attack. The final type, Attack Data #4, is the case in which unauthorized data are accessed, or attempts are made to access data not observed in the training data. It is similar to performing a reconnaissance attack to obtain information on the target system.

5.2. Preprocessing and Training

To verify the proposed method, the LSTM model was implemented using TensorFlow V2.16.1, and the experiment environment consisted of an Intel i7-8700 CPU, 16 GB memory, and GTX 1060 6 GB GPU. TShark 2.6.8 was used to parse the packet data. Subsequently, the other steps, from preprocessing to training and executing the model, were developed using Python 3.9. If the LSTM model is trained in high-performance devices, the trained model can be executed in low-performance devices such as field devices.

In the preprocessing of the proposed method, the fields of each packet were parsed using TShark. Then, the packets were classified for each communication section, and packets that had the same function code and data object were grouped. As the dataset used in the experiment consists of MMS traffic, both PDUs and services were used instead of the function code. If the packet parsing in preprocessing is changed, the proposed method can be used in other protocols. Therefore, the proposed method is not dependent on the protocol.

Subsequently, communication pattern analysis was performed for each group. In the pattern analysis, the one-dimensional data are clustered; therefore, the K-means-based algorithm uses X-means [26]. X-means is similar to K-means; however, it is suitable for automation because it can run without a parameter that represents the number of clusters. After the delta times are clustered using X-means, the centroid value of the cluster that includes the maximum delta time is inferred as a period. Even if the centroid value is the same as the period set in the system, the centroid value and the time between packets are different because network delays may occur during the traffic transmission process. Therefore, it is necessary to set an error range to determine the traffic periodicity. If this error range is too small, the proposed method may incorrectly analyze the period because of delayed packets. However, if it is too wide, aperiodic communication may be periodically misrecognized. If the communication pattern analysis between target systems is not performed well, it can be solved by adjusting the error range.

The LSTM model is trained with a sequence of periodic communications and saved. TensorFlow is used to implement the LSTM model. To learn smoothly the patterns that appear in a small number of sequences, a bidirectional LSTM was used. The following hyperparameters were used:

- Learning Rate: 0.01
- Optimizer: Adam Optimizer
- Epoch: 50
- Sequence Length: five

As the data used for learning have periodicity, it is repeated continuously after a certain period. Therefore, the epoch was set to 50 because it was predicted that sufficient learning would be achieved even with a smaller number of training epochs. The length of the sequence was arbitrarily set to five. A longer sequence can lead to higher accuracy. However, it is not suitable for field devices because their memory occupancy is high. The results of the experiment confirmed a good performance, even with an arbitrary sequence length. It is possible to set the sequence length by using an open source to tune

the hyperparameter such that a high accuracy would be achieved. In the execution phase, the trained model is loaded after the preprocessing. The prediction and real packets are compared. When the target packet and prediction of the model are different, it is determined to be abnormal.

5.3. General Result

The data used in this experiment are composed of periodic data, as shown in Table 1. In total, seven periodic communications are determined using the communication pattern analysis algorithm. In the case of MMS packets, all Information Report packets of the Unconfirmed PDU, and the Identify packet of the Confirmed Request PDU were determined as periodic. Based on the result of the analysis based on TCP, most periodic communications were well analyzed; however, it was incorrectly confirmed that there is no period in some traffic. Table 2 presents the analysis of the communication pattern based on the application layer (MMS) and the transport layer (TCP).

Table 2. Result of Communication Pattern Analysis.

Target	Application Layer	Transport Layer
Repo 1	Periodic	Periodic
Repo 2	Periodic	Periodic
Repo 3	Periodic	Periodic
Repo 4	Periodic	Periodic
Repo 5	Periodic	Periodic
Repo 6	Periodic	Periodic
Identify	Periodic	Periodic
TCP (length = 0)	-	Non-Periodic
TCP (length = max)	-	Partially Periodic

In Table 2, Repo 1 to 6 refer to data transmitted to the Information Report of the Unconfirmed PDU and Identify refers to the Identify service packet of the Confirmed Request PDU. TCP (length = 0) is a TCP packet with a length of 0, and TCP (length = max) is a TCP packet whose length is set to the maximum segment size. Packets of the maximum size are a part of the fragmented Information Report packet. As the last packet of TCP fragmentation is reassembled along with the previous packets, it is interpreted as an MMS packet. Therefore, the above analysis table deals with only TCP packets that have the maximum length or a length of zero.

In the result of the communication pattern analysis based on the application layer, all periodicities are identified. However, the communication analysis for each data based on the transport layer had a problem in that it is not possible to identify accurately the period of the TCP packets with a length of zero or the maximum length. First, a TCP packet with a length of zero was confirmed as an ACK via a detailed analysis. An ACK, a zero-length TCP packet, is sent in response to any type of data transmission. While analyzing only a unidirectional communication section, it is difficult to distinguish the ACKs that correspond to a specific packet. Hence, the period of the zero-length TCP packets was not analyzed well. In the case of TCP (length = max), it is a part of a fragmented packet. When the entire packet arrives, it is reassembled and identified as one MMS packet. However, it is impossible to distinguish each datum with a divided TCP packet. Therefore, not all the periods were recognized. Tables 3 and 4 present the test results in Model A, which analyzes the communication pattern based on the application layer, and Model B, which analyzes the communication pattern based on the transport layer.

To measure the performance of the proposed method, we used the accuracy, precision, recall, and number of false positive cases. Each indicator consists of the calculations of the true positive (TP), true negative (TN), false positive (FP), and false negative (FN). The details are as follows:

- $Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$
- $Precision = \frac{TP}{TP + FP}$
- $Recall = \frac{TP}{TP + FN}$
- # of FPCases = Number of False Positive Cases

Table 3. Test result of proposed anomaly method (Application layer).

Dataset	Accuracy	Precision	Recall	# of FP Cases
Normal Data	100%	-	-	-
Attack Data #1	72.5%	57.1%	93.2%	171 Cases
Attack Data #2	85.8%	71.8%	90.9%	48 Cases
Attack Data #3	78.7%	60.4%	88.4%	65 Cases
Attack Data #4	81.6%	65.5%	100%	121 Cases

Table 4. Test result of proposed anomaly method (Transport layer).

Dataset	Accuracy	Precision	Recall	# of FP Cases
Normal Data	90.5%	-	-	10 Cases
Attack Data #1	83.1%	72.4%	82.9%	254 Cases
Attack Data #2	83.1%	71.8%	82.4%	173 Cases
Attack Data #4	91.1%	78.8%	100%	202 Cases

In the case of Model A, which analyzed and trained the communication pattern based on the application layer, all data were well identified as normal. However, the accuracy was lower than that of Model B in the test that used attack data. By contrast, in the case of Model B, which analyzed and trained the communication pattern based on the transport layer, some of the normal data lead to incorrect alerts. However, its accuracy was higher than that of Model A in most experiments. Model A was observed to have a lower FP rate than Model B.

In the experiment that used normal data, Model A had no FPs. However, in the experiment that used the test data, the accuracy was lower than expected. To find the reason for it, we analyzed the FP and FN cases. It was confirmed that most of the FPs and FNs occurred when attack data were included in the sequence. The detection model outputs the expected value of the fourth packet out of a sequence consisting of five packets. If the expected value does not match the actual value, the model decides that it is abnormal. Therefore, because all five packets forming a sequence affect the expected value, the probability of FPs and FNs is very high when there are abnormal packets in the sequence. The FPs and FNs of the tests are shown in Table 5.

In both models, we confirmed that most of the FPs and FNs are related to attack data in the sequence. In the case of Model A, all the FPs are related to attacks in the sequence. Because the model is well trained, there were no false alarms in the test when the normal test dataset was used. By contrast, in the case of Model B, there were ten FPs in the test that used only normal data, unlike Model A. Some of these FPs were not related to the attack.

Table 5. False positives and false negatives related with attack.

Dataset	Model A (Application Layer)				Model B (Transport Layer)			
	FP Cases	FN Cases	Related FP Cases	Related FN Cases	FP Cases	FN Cases	Related FP Cases	Related FN Cases
Attack Data #1	171	17	171	14	254	150	252	139
Attack Data #2	48	13	48	11	173	103	167	92
Attack Data #3	65	13	65	11	-	-	-	-
Attack Data #4	121	0	121	0	202	0	193	0

6. Discussion

An FP case analysis was performed to improve the performance of Model B. 10 FP cases were observed. All of them included zero length TCP packets excluded in the training phase. In a traffic where periodic and aperiodic data coexist, we performed aperiodic data removal through X-means clustering. In this method, distinguishing whether a TCP ACK corresponds to a periodic or an aperiodic request is difficult, thus TCP ACKs were omitted from the training data.

However, since most of our data set was composed of periodic data transmission, we determine that it was possible for the learning of the sequence with entire data. If the model is trained with a large amount of normal data, it is anticipated that small amounts of aperiodic ACKs will have no significant impact on learning from normal data. Therefore, we experimented with two methods: a method to fine-tune the model using the sequences that caused false positives, and another method to training the entire data, including aperiodic data. The relearning results are shown in Figure 8.

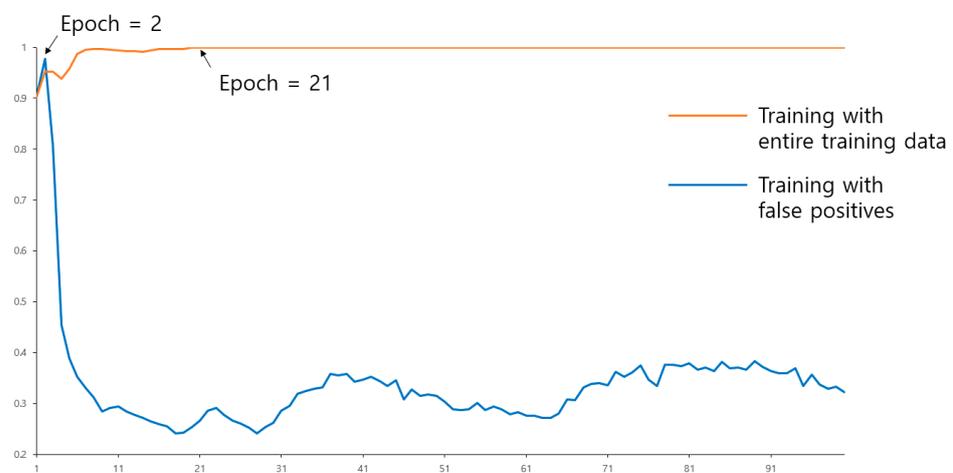


Figure 8. The change in model accuracy across epochs.

As a result of retraining with FP cases, the model was overtrained to them. After 2nd epochs, previous learning was almost initialized as an untrained model. After ten epochs, we observed all the trained FP cases to be correct, but the model confused a periodic packet transmission as abnormal. To solve this problem, we trained the entire dataset that contained aperiodic communication. Consequently, the detection model classified all normal data as normal after 21 iterative trainings. Tables 6 and 7 show the evaluation result of two models using attack data.

Table 6. Test result of the proposed anomaly method using the retrained model.

Dataset	Retraining with False Positives			Retraining with Entire Normal Data		
	Accuracy	Precision	Recall	Accuracy	Precision	Recall
Attack Data #1	84.2%	75.1%	82.9%	87.7%	80.1%	83.6%
Attack Data #2	85.7%	76.5%	82.4%	88.4%	81.0%	86.6%
Attack Data #4	96.5%	90.5%	100%	96.7%	91.2%	100%

Table 7. False positives and false negatives related with attack in the retrained model.

Dataset	Retraining with False Positives				Retraining with Entire Normal Data			
	FP Cases	FN Cases	Related FP Cases	Related FN Cases	FP Cases	FN Cases	Related FP Cases	Related FN Cases
Attack Data #1	246	166	243	143	173	136	173	120
Attack Data #2	141	95	140	80	119	81	119	75
Attack Data #4	87	0	85	0	86	0	85	0

As we anticipated the re-trained model is improved in the accuracies for all the attacks. Retraining using all data improved the detection rate by approximately 5.15% compared to the previous one. It also had higher accuracy than the fine-tuned model using only FP cases.

In the case of a model fine-tuned to FP cases, the model identified the FP cases to be normal. However, it falsely detected some of the patterns that were previously determined as normal. New FP cases were observed in the fine-tuned model. In some of the new FP cases, the attack packet was not included in the sequence. In the case of the model that trained the entire dataset, the result confirmed that all of FP cases were related with attack packets. In other words, all input of FP cases included the attack packet. Therefore, it is confirmed that the performance of the model can be improved by re-training the entire data.

The comparison with other methods is very limited. The common anomaly detection studies focus on detecting complicated attacks which cannot detect the proposed method, such as FDI. However, the proposed method tries to detect replay attacks. Most other approaches are not designed to detect replay attacks. It does not mean the proposed method is outperformed by other methods. They have different focuses and detect different attacks.

7. Conclusions and Future Works

In this study, to detect an attack against an ICS, we proposed a novel anomaly detection method that defines normal behavior using a periodic communication pattern. The proposed method can be used in most ICSs, independent of a communication protocol, because it automatically regulates normal behavior based on the communication pattern's periodicity. Because the proposed method automatically defines normal behavior, it can reduce the cost of analysis and incorrect normal behavior owing to human error. In addition, as the training phase and the execution phase of the detection model are separated, it is relatively robust against the adversarial attacks that induce a false learning by contaminating the training data.

To verify the proposed technique, a test was performed using the network traffic of the Korea Electric Power Control System. Consequently, the model that defines the normal behavior of the application layer found all the communication patterns and exhibited excellent performance. In the other model, which defines the normal behavior of the transport layer, where fragmentation occurred, the average detection rate was 80%. Most FPs and FNs occurred before and after abnormal packets. This was confirmed to be because of the characteristics of the time-series prediction model, that is, LSTM. In future studies, we will consider methods for preventing the propagation of FPs and FNs because of such abnormal packets. To reduce the effect of abnormal packets, a technique for removing packets that are determined to be abnormal from the sequence can be used. However, because a bidirectional LSTM deep learning model is used in this study, both previous and subsequent packets affect the prediction. In other words, the prediction is affected by a packet that has not yet been inspected. This problem cannot be solved by simply removing the packet detected by an attack. Therefore, further research is necessary to learn the patterns that are rarely observed in a unidirectional LSTM, which considers only previous data for prediction.

The proposed method can be improved with domain knowledge. The improved model is expected to have high accuracy.

Author Contributions: Conceptualization, H.K. and D.-I.J.; Methodology, S.K., H.K. and H.C.; Validation, W.J. and H.C.; Investigation, D.-I.J.; Resources, S.C.; Writing—original draft, S.K. and W.J.; Writing—review & editing, S.K., H.K., S.C. and T.S.; Project administration, T.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (NRF-2018R1D1A1B07043349).

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: This research was conducted before Author Sungjin Kim was employed by Samsung SDS. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Kaspersky. ICS Threat Predictions for 2021. Available online: <https://ics-cert.kaspersky.com/media/Kaspersky-ICS-CERT-Threat-Predictions-2021-EN.pdf> (accessed on 8 December 2020).
2. Al-Hawawreh, M.; Hartog, F.D.; Sitnikova, E. Targeted ransomware: A new cyber threat to edge system of brownfield industrial Internet of Things. *IEEE Internet Things J.* **2019**, *6*, 7137–7151. [[CrossRef](#)]
3. Kim, S.; Jo, W.; Shon, T. APAD: Autoencoder-based Payload Anomaly Detection for industrial IoE. *Appl. Soft Comput.* **2020**, *88*, 106017. [[CrossRef](#)]
4. Guo, S.; Wang, J.; Chen, Z.; Li, Y.; Lu, Z. Securing IoT Space via Hardware Trojan Detection. *IEEE Internet Things J.* **2020**, *7*, 11115–11122. [[CrossRef](#)]
5. Chaudhry, S.A.; Shon, T.; Al-Turjman, F.; Alsharif, M.H. Correcting design flaws: An improved and cloud assisted key agreement scheme in cyber physical systems. *Comput. Commun.* **2020**, *153*, 527–537. [[CrossRef](#)]
6. Mahmood, K.; Chaudhry, S.A.; Naqvi, H.; Shon, T.; Ahmad, H.F. A lightweight message authentication scheme for Smart Grid communications in power sector. *Comput. Electr. Eng.* **2016**, *52*, 114–124. [[CrossRef](#)]
7. Yoo, H.; Shon, T. Challenges and research directions for heterogeneous cyber–physical system based on IEC 61850: Vulnerabilities, security requirements, and security architecture. *Future Gener. Comput. Syst.* **2016**, *61*, 128–136. [[CrossRef](#)]
8. McLaughlin, S.; Konstantinou, C.; Wang, X.; Davi, L.; Sadeghi, A.-R.; Maniatakos, M.; Karri, R. The cybersecurity landscape in industrial control systems. *Proc. IEEE* **2016**, *104*, 1039–1057. [[CrossRef](#)]
9. Khraisat, A.; Gondal, I.; Vamplew, P.; Kamruzzaman, J. Survey of intrusion detection systems: Techniques, datasets and challenges. *Cybersecurity* **2019**, *2*, 20. [[CrossRef](#)]
10. Ahmed, C.M.; Ochoa, M.; Zhou, J.; Mathur, A.P.; Qadeer, R.; Murguia, C.; Ruths, J. Noiseprint: Attack detection using sensor and process noise fingerprint in cyber physical systems. In Proceedings of the 2018 on Asia Conference on Computer and Communications Security, Incheon, Republic of Korea, 4 June 2018.
11. Mirsky, Y.; Doitshman, T.; Elovici, Y.; Shabtai, A. Kitsune: An ensemble of autoencoders for online network intrusion detection. *arXiv* **2018**, arXiv:1802.09089.
12. Kwon, S.; Yoo, H.; Shon, T. IEEE 1815.1-Based Power System Security with Bidirectional RNN-Based Network Anomalous Attack Detection for Cyber-Physical System. *IEEE Access* **2020**, *8*, 77572–77586. [[CrossRef](#)]
13. Feng, C.; Palleti, V.R.; Mathur, A.; Chana, D. A Systematic Framework to Generate Invariants for Anomaly Detection in Industrial Control Systems. In Proceedings of the Network and Distributed Systems Security (NDSS) Symposium 2019, San Diego, CA, USA, 24–27 February 2019.
14. Choi, H.; Lee, W.-C.; Aafer, Y.; Fei, F.; Tu, Z.; Zhang, X.; Xu, D.; Deng, X. Detecting attacks against robotic vehicles: A control invariant approach. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, Toronto, ON, Canada, 15–19 October 2018.
15. Jo, W.; Kim, S.; Kim, H.; Shin, Y.; Shon, T. Automatic whitelist generation system for ethernet based in-vehicle network. *Comput. Ind.* **2022**, *142*, 103735. [[CrossRef](#)]
16. Yang, H.; Cheng, L.; Chuah, M.C. Deep-learning-based network intrusion detection for SCADA systems. In Proceedings of the 2019 IEEE Conference on Communications and Network Security (CNS), Washington, DC, USA, 10–12 June 2019.
17. Jo, W.; Kim, S.; Lee, C.; Shon, T. Packet Preprocessing in CNN-Based Network Intrusion Detection System. *Electronics* **2020**, *9*, 1151. [[CrossRef](#)]
18. Pan, S.; Morris, T.; Adhikari, U. Developing a hybrid intrusion detection system using data mining for power systems. *IEEE Trans. Smart Grid* **2015**, *6*, 3104–3113. [[CrossRef](#)]
19. Jeffrey, N.; Tan, Q.; Villar, J.R. Using Ensemble Learning for Anomaly Detection in Cyber–Physical Systems. *Electronics* **2024**, *13*, 1391. [[CrossRef](#)]
20. Galloway, B.; Hancke, G.P. Introduction to industrial control networks. *IEEE Commun. Surv. Tutor.* **2012**, *15*, 860–880. [[CrossRef](#)]
21. Barbosa, R.R.R.; Sadre, R.; Pras, A. A first look into SCADA network traffic. In Proceedings of the 2012 IEEE Network Operations and Management Symposium, Maui, HI, USA, 16–20 April 2012.
22. Terai, A.; Abe, S.; Kojima, S.; Takano, Y.; Koshijima, I. Cyber-attack detection for industrial control system monitoring with support vector machine based on communication profile. In Proceedings of the 2017 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), Paris, France, 26–28 April 2017.
23. Ni, J.; Yin, W.; Jiang, Y.; Zhao, J.; Hu, Y. Periodic Mining of Traffic Information in Industrial Control Networks. In *Advanced Information Networking and Applications, Proceedings of the 34th International Conference on Advanced Information Networking and Applications (AINA-2020)*, Caserta, Italy, 15–17 April 2020; Springer: Cham, Switzerland, 2020.
24. Sun, L.; Dou, Y.; Yang, C.; Zhang, K.; Wang, J.; Yu, P.S.; He, L.; Li, B. Adversarial attack and defense on graph data: A survey. *arXiv* **2018**, arXiv:1812.10528. [[CrossRef](#)]

25. Lee, R.M. The Industrial Control System Cyber Kill Chain. Available online: <https://www.sans.org/white-papers/36297/> (accessed on 7 April 2024).
26. Pelleg, D.; Moore, A.W. X-means: Extending k-means with efficient estimation of the number of clusters. In Proceedings of the 17th International Conference on Machine Learning, San Francisco, CA, USA, 29 June–2 July 2000; Volume 1.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.