applied sciences

MDPI

Article

Anomaly Detection Method for Unknown Protocols in a Power Plant ICS Network with Decision Tree

Kyoung-Mun Lee [1,2], Min-Yang Cho [3,4], Jung-Gu Kim [5] and Kyung-Ho Lee [2,*]

[1] Korea Southern Power Co., Ltd., Andog-si 36618, Republic of Korea
[2] Department of Information Security, Korea University, Seoul 02841, Republic of Korea
[3] Department of Computer Science and Engineering, Korea University, Seoul 02841, Republic of Korea
[4] Department of Computer Software, Dong Seoul University, Seongnam-si 13117, Republic of Korea
[5] Department of AI Application Software, Dong Seoul University, Seongnam-si 13117, Republic of Korea
* Correspondence: kevinlee@korea.ac.kr; Tel.: +82-2-3290-4885

Abstract: This study aimed to enhance the stability and security of power plant control network systems by developing detectable models using artificial intelligence machine learning techniques. Due to the closed system operation policy of facility manufacturers, it is challenging to detect and respond to security threats using standard security systems. With the increasing digitization of control systems, the risk of external malware penetration is also on the rise. To address this, machine learning techniques were applied to extract patterns from network traffic data produced at an average of 6.5 TB per month, and fingerprinting was used to detect unregistered terminals accessing the control network. By setting a threshold between transmission amounts and attempts using one month of data, an anomaly judgment model was learned to define patterns of data communication between the origin and destination. The hypothesis was tested using machine learning techniques if a new pattern occurred and no traffic occurred. The study confirmed that this method can be applied to not only plant control systems but also closed-structured control networks, where availability is critical, and other industries that use large amounts of traffic data. Experimental results showed that the proposed model outperformed existing models in terms of detection efficiency and processing time.

Keywords: ICS; unknown protocol; fingerprint; anomaly detection; AI

Citation: Lee, K.-M.; Cho, M.-Y.; Kim, J.-G.; Lee, K.-H. Anomaly Detection Method for Unknown Protocols in a Power Plant ICS Network with Decision Tree. *Appl. Sci.* **2023**, *13*, 4203. https://doi.org/10.3390/app13074203

Academic Editors: Chin-Shiuh Shieh, Shu-Chuan Chu and Tarek Gaber

Received: 16 February 2023
Revised: 23 March 2023
Accepted: 24 March 2023
Published: 26 March 2023

1. Introduction

The control network used in power generation is an important backbone network that is physically isolated from outside networks. In case of any abnormality within the network, it can directly affect the competitiveness of the power plant, making it crucial to preventing and detecting issues proactively rather than just responding to them after they have occurred [1,2].

The ICS (Industrial Control System) network configuration comprises two sections: the controller communication section and the control server/client communication section. The ICS network uses a one-way data transmission device to transfer data to a business network. This enables the operation data generated from the control system to be used offsite while ensuring the data transmission direction remains one-way and cannot be reversed from the business network to the control network. The one-way data transmission device is responsible for the secure transmission of data from the control network to the business network [3,4].

Since the control network is a closed network as shown in Figure 1, the current ICS network requires a device identification method based on passive methods. Moreover, the physical risks to the network are not evaluated, and the status of devices on the control network cannot be determined using any method other than that provided by the control system manufacturer [5,6].

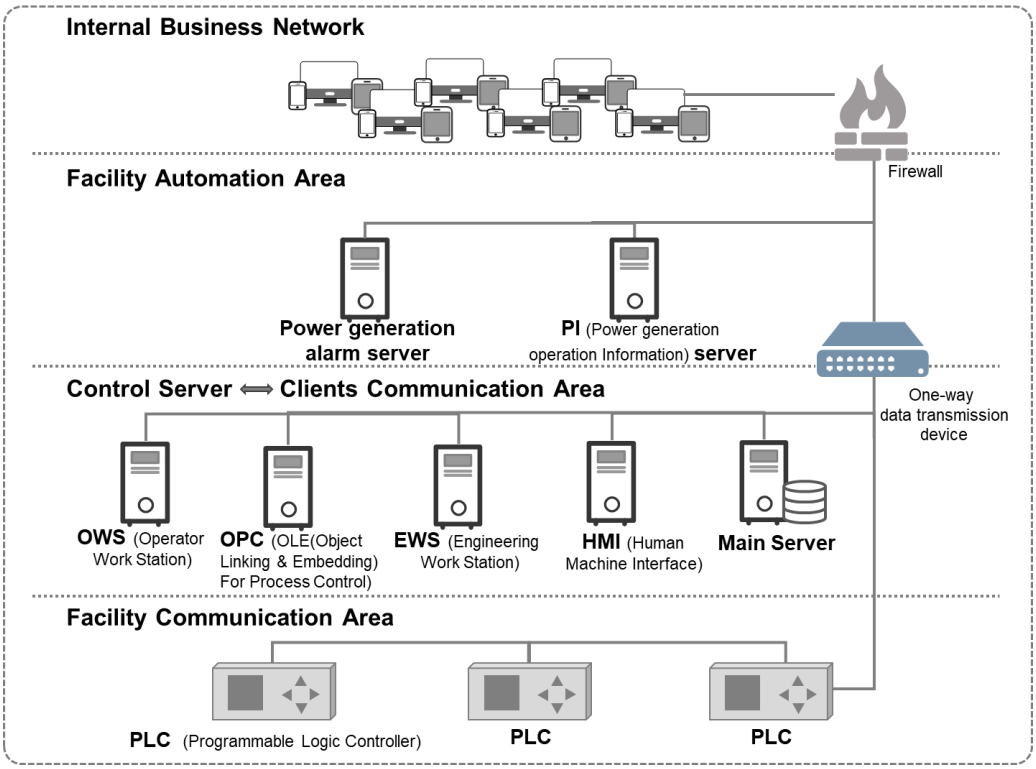

Figure 1. Network configuration of the power generation control system.

The control system in Korean power plants operates as a closed network that is directly disconnected from external networks, and remote security installation is not possible. In addition, it is difficult to apply the security system in the field because it is limited to installing a separate security solution in the control system due to the manufacturer's policy. Therefore, there is a lack of real-time identification of assets in the control network and methods for obtaining network visibility, which makes it impossible to verify the stable operating state of the control network. When a problem occurs in the control network, there is no information to identify the cause, making it difficult to analyze the cause and hindering the establishment of a plan for stable operations [7,8].

Thus, it is necessary to collect network traffic data without affecting the control system's availability to detect and record any anomalies in the control system. In order to solve these problems, this study aimed to use a machine learning-based approach that extracts communication patterns from network traffic data and a fingerprinting technique that detects unauthorized terminals attempting to access the control network.

In this study, white fingerprinting is used as the primary method for TCP (Transmission Control Protocol), which increases the probability of identifying human communication. On the other hand, UDP (User Datagram Protocol) is primarily used for automated processing, and pattern extraction can more easily distinguish what is not a pattern. We propose a technique for detecting anomalous traffic that involves comparing detection rates by applying machine learning to traffic data accumulated in the control network based on big data analytics. Machine learning techniques can be classified into supervised learning, unsupervised learning, and reinforcement learning [9,10].

Our approach can accurately identify new patterns of data communication between the source and destination by setting a threshold between transmission amounts and attempts and using one month's worth of data to learn an anomaly detection model. Our technique can help protect industrial assets from potential threats caused by changes in the industrial environment and meet operator requirements for a secure industrial control environment.

In this paper, we propose a detection technique for anomalous traffic that compares detection rates by applying machine learning to traffic data accumulated in the control

network based on big data analytics. We present an anomaly detection model that matches the characteristics of the stable operation of the control network. Our main contributions are as follows: (1) we use machine learning techniques to automatically pattern and classify private protocols in the closed control network, providing a basis for securing the network's health by checking the types of transmitted traffic data, and (2) we propose a whitelist (allowlist) that enables control system operators to intuitively recognize abnormal patterns that may occur in the operation stage and notify the information security officer for further analysis.

The remainder of this paper is organized as follows. Section 2.1 reviews existing machine learning and traffic detection methods. Section 2.2 presents our anomaly detection model. Section 3 presents the test and verification results of the proposed model and the existing model for the ICS network of the power plant. Section 4 provides a performance comparison and discussion. Section 5 is the conclusion of this study.

2. Materials and Methods

2.1. Related Work

This section examines existing ML technologies and traffic detection technologies in the developed ICS network and compares them in terms of availability and safety.

2.1.1. AI Machine Learning

Artificial intelligence technology refers to the ability of machines to perform human cognitive processes such as perception, information processing, and decision-making. AI can be divided into general AI and narrow AI. General AI refers to human-level intelligence, which means performing cognitive activities regardless of the level, type, or content. However, there is currently no AI technology that satisfies the definition of general AI, and it is expected to take several decades of further development. Narrow AI, on the other hand, refers to an algorithm that specifically solves a defined problem, and all currently used AI technologies are narrow AI [11–14]. As shown in Table 1, the types of machine learning algorithms can be classified.

Table 1. Machine learning types.

Types	Description	
Supervised Learning	Classification	
	- K-Nearest Neighbors [15]	- Support Vector Machine [16]
	- Decision Tree [17]	- Random Forest [18,19]
	Regression	
	- Linear Regression [20,21]	- Logistic Regression [22]
Unsupervised Learning	- Clustering [23]	- Visualization [24]
	- Dimensionality Reduction [25]	- Association Rule Learning [26]
Reinforcement Learning	- DQN (Deep Q-Network) [27]	
	- A3C (Asynchronous Advantage Actor–Critic) [28]	

The AI algorithm used in this study is determined by categorical prediction targets based on the number of data cases, and a decision tree is used during classification with the correct labels. The algorithm selection flow is illustrated in Figures 2 and 3, which show the steps for selecting the most appropriate algorithm for the dataset.

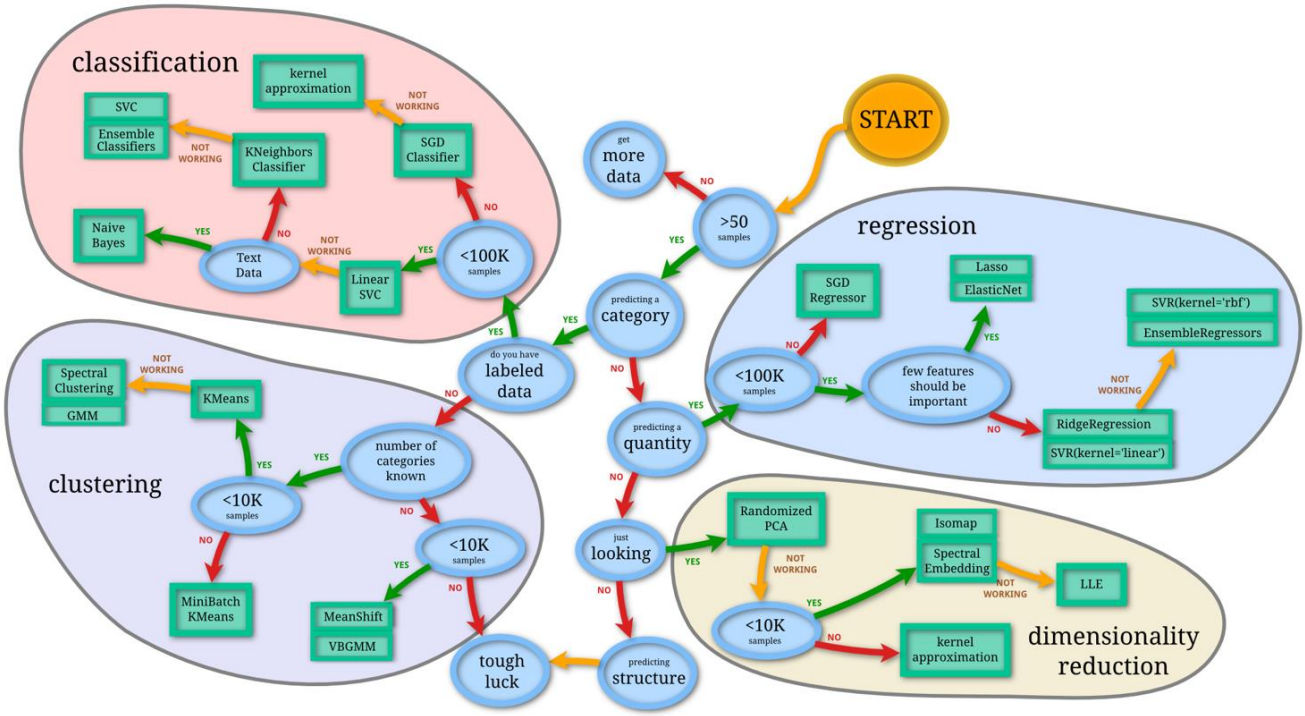

Figure 2. Selecting the right machine learning technique [29].

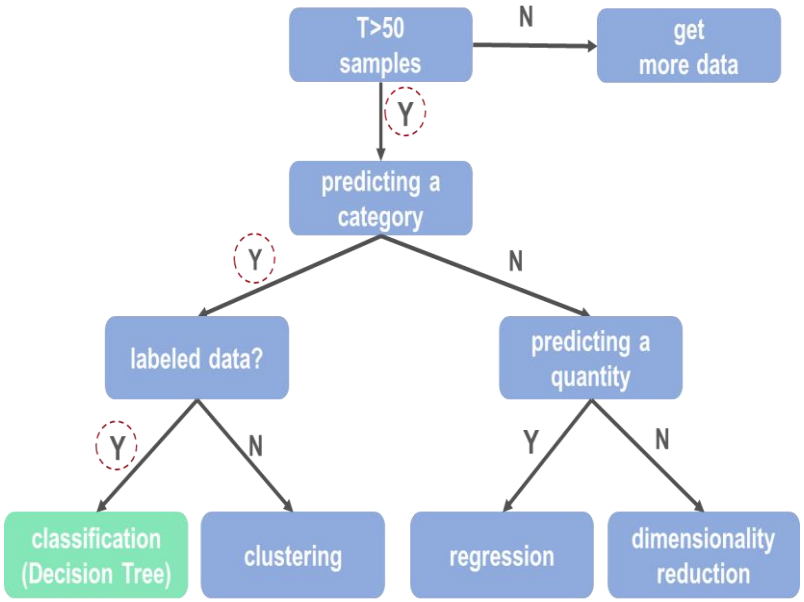

Figure 3. Flowchart for selecting machine learning techniques.

In this study, we chose the decision tree as our machine learning algorithm because it is intuitive and well-suited for analyzing closed networks of private protocols. We also minimized the data preprocessing process to prevent overfitting and ensure that the importance of variables is fully understood. By doing so, we aimed to achieve accurate and reliable anomaly detection results that can help improve the security of industrial control systems.

2.1.2. Closed Network Asset Fingerprinting

Asset fingerprinting involves gathering information such as MAC (Media Access Control) and IP (Internet Protocol) addresses, host names (Client Identifier option), manufacturer information (Vendor class identifier), and the parameter request list option from network packets. These details can be used as a type of fingerprint, and asset fingerprinting is a technique that aims to find these fingerprints. The technique is implemented passively by observing captured network data and can be used to identify network IT assets to determine if any unauthorized IT asset is attempting to access the network. This can be completed by checking a specially crafted packet that uses an unusual set of options [30–33].

2.1.3. Closed Network Manual Whitelisting

Manual whitelisting is a technique that only allows explicitly white-listed network communications, which offers the advantage of protecting a network from a range of known and unknown malicious events and behaviors [34]. While high accuracy increases the chances of blocking known and unknown attacks, it comes with the disadvantage of requiring an expensive configuration to achieve high accuracy and efficiency [35–37].

2.1.4. Anomaly Detection in an Industrial Control System

Recently, research has been conducted to assess the performance of anomaly detection models using techniques such as preprocessing, filtering, and feature extraction of data, which leverages machine learning and deep learning algorithms [38,39]. Moreover, methods for detecting anomalies on control systems are being studied to enhance the performance of anomaly detection models for network traffic. However, directly implementing anomaly detection solutions on control systems may not be feasible for systems that prioritize availability.

For modern industrial control systems, the configuration system changes frequently. Therefore, deep learning neural network techniques are being utilized to detect and monitor rapid asset changes. However, industrial facilities, such as power plants, undergo minimal changes [40]. Generally, autoencoder neural networks individually reconstruct predicted data by learning from normal data. However, methods that simultaneously perform prediction and reconstruction on input data, utilizing a single-mode approach, are also being studied to achieve more effective attack detection [41].

In the case of industrial control systems connected to the internet, research is also being conducted to enhance the detection rate of abnormal symptoms by identifying data imbalance to address undefined attacks on the network [42].

2.2. Proposal

This study presents a monitoring technique to prevent intrusion accidents in a closed control network environment by detecting anomalous traffic. Anomalous traffic in a closed control network can broadly be defined into two patterns. The first pattern is a case where it is necessary to verify whether a normal operation is performed, as the expected traffic does not occur. The second pattern is when random traffic that should not occur is generated.

Pattern 1 results from mechanical errors in hardware and software and is mainly due to equipment loss, errors, and malfunctions. Pattern 2 is referred to as human error and is caused by intentional actions or mistakes.

For comparison and verification of the detection technique for anomalous traffic, supervised learning is applied to the previous patterns, and the false-positive rate is analyzed by generating random anomalous traffic to verify the learning performance and consistency of the detection model.

2.2.1. Anomaly Detection Model Patterns

To diagnose anomalies, traffic that meets the reference point K (over 400 traffic attempts) should be targeted for the number of communications. In natural traffic, defining a specific communication pattern is difficult because traffic with only packet headers (64 KB

(Kilobyte)) includes attempted traffic with payloads of various sizes. Limited kinds of sizes are generated because they are generated by "patterned communication".

H_0: *General Model* $O = K \times (T \geq 9)$.

In the null hypothesis, general model O, the number of packet sizes T of the traffic that satisfies all the reference points K is 9 or more.

H_1: *Detection Model P is* $P = K \times (T < 9)$.

The alternative hypothesis, detection model P, can be defined as "patterned communication" when the number of packet sizes T is 8 or less through an intended attempt for the traffic that satisfies all reference points K. The alternative hypothesis posits that pattern communication is performed in the control network of the private protocol. Pattern communication is defined as a control network communication with a P pattern that satisfies both K, the number of packet sizes generated from SRC (Source)_PORT to DST (Destination)_PORT, and T, the frequency of occurrence for each packet size.

The AI decision tree algorithm is used to find the condition factors K and T for deriving the factor P of the hypothesis. K should derive the type of packet size to find an optimal factor, and T should derive a boundary value that exceeds the minimum frequency of occurrence for each K. If both conditions are satisfied, P is defined as a control network communication having a pattern, and if it deviates from the pattern, it can be determined as an abnormal symptom.

The network traffic generated within the control network has the following basic structure: the number of bytes per IP, SRC_PORT, and DST_PORT. Figure 4 shows the system configuration used in this study. The traffic generated from the control network was collected in the collection area through a collection agent, and the collected data were transmitted to the analysis system and used for analysis.

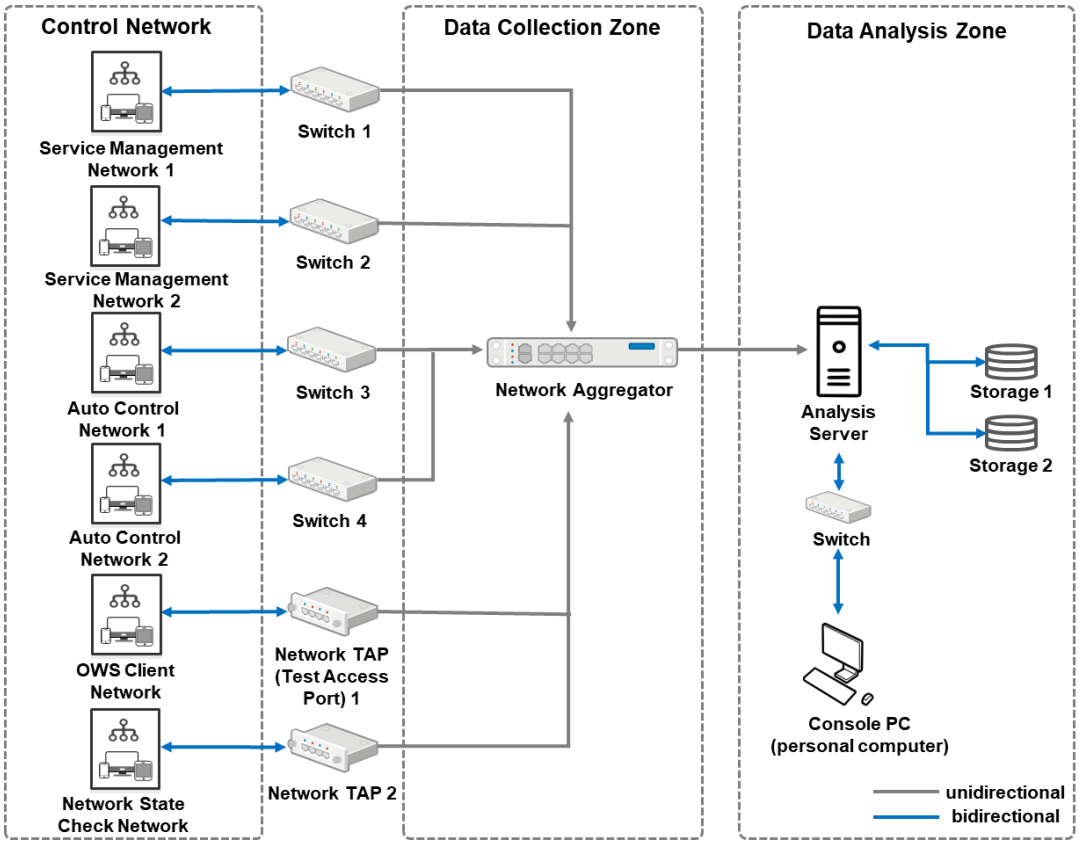

Figure 4. Network configuration for pattern detection.

In the case where the traffic generated from SRC_PORT to DST_PORT exceeds a specific threshold (initially set value T learned from data), it is identified as an abnormal symptom. Examples of patterns registered between operations are configured to maintain detection patterns by applying set values accumulated through supervised learning.

Traffic data within the control network is separated by the TCP and UDP protocols and preprocessed by composing XML (Extensible Markup Language) files. Figure 5 shows the TCP protocol metadata schema, and Figure 6 shows the UDP protocol metadata schema structure.

```
<?xml version="1.0" encoding="utf-8"?>

<table_data name="flow_raw_data">
        <row>
                <field name="ID">4</field>
                <field name="SRC_PORT">1065</field>
                <field name="DST_PORT">139</field>
                <field name="IN_PKTS">1</field>
                <field name="IN_BYTES">138</field>
                <field name="OUT_PKTS">0</field>
                <field name="OUT_BYTES">0</field>
                <field name="IP_PROTOCOL">6</field>
                <field name="VLAN_ID">10</field>
        </row>
        <row>
                <field name="ID">134</field>
                <field name="SRC_PORT">1065</field>
                <field name="DST_PORT">139</field>
                <field name="IN_PKTS">1</field>
                <field name="IN_BYTES">138</field>
                <field name="OUT_PKTS">0</field>
                <field name="OUT_BYTES">0</field>
                <field name="IP_PROTOCOL">6</field>
                <field name="VLAN_ID">10</field>
        </row>
```

Figure 5. TCP protocol metadata schema.

```
<table_data name="flow_raw_data">
        <row>
                <field name="ID">118640</field>
                <field name="SRC_PORT">49418</field>
                <field name="DST_PORT">1900</field>
                <field name="IN_PKTS">6</field>
                <field name="IN_BYTES">954</field>
                <field name="OUT_PKTS">0</field>
                <field name="OUT_BYTES">0</field>
                <field name="IP_PROTOCOL">17</field>
                <field name="VLAN_ID">20</field>
        </row>
        <row>
                <field name="ID">118641</field>
                <field name="SRC_PORT">49439</field>
                <field name="DST_PORT">3702</field>
                <field name="IN_PKTS">2</field>
                <field name="IN_BYTES">1340</field>
                <field name="OUT_PKTS">0</field>
                <field name="OUT_BYTES">0</field>
                <field name="IP_PROTOCOL">17</field>
                <field name="VLAN_ID">20</field>
        </row>
```

Figure 6. UDP protocol metadata schema.

The XML schema is a user-defined type with a flexible structure that can immediately reflect changes in control network items. The schema structure consists of the ID (Identifier), SRC_PORT, DST_PORT, input packets, input bytes, output packets, output bytes, protocol types (TCP:6 and UDP:17, etc.), and VLAN (Virtual Local Network) ID, which are indexes. The data type is defined as a number.

To extract an anomaly pattern, the first step is pattern extraction, the second step is pattern communication count, and the third step is extraction of the anomaly time point.

2.2.2. Anomaly Pattern Extraction Relation Algebra

The three-step process mentioned above is aimed at detecting and deriving anomaly points using real traffic data. The following describes the process for verifying the anomaly pattern.

Step 1 relational algebra—pattern extraction relational algebra

$$\pi G_S,G_D,P,F_{COUNT(T)},F_{SUM(}F_{COUNT(S))}(\sigma_{S>\underline{SB}}(\sigma F_{COUNT(T))<\underline{T}B}(FR)))$$

G: Group	F: Function	$_S$: SOURCE_PORT
$_D$: DESTINATION_PORT	$_P$: IP_PROTOCOL	$_T$: IN_BYTES/IN_PKTS (Packets)
FR: FLOW_RAW_DATA	$\underline{SB}$: COUNT(SRC_PORT)	$\underline{TB}$: COUNT(IN_BYTES/IN_PKTS)

In Step 1, the types of patterns and traffic figures for each protocol (mainly TCP and UDP) are obtained from SRC_PORT to DST_PORT. Cases where the number of occurrences is T or more and the pattern type is K or less are sorted in order of occurrence frequency and then checked.

Step 2 relational algebra—pattern communication count relational algebra

$$\pi G_S,G_D,P,G_T,F_{COUNT(S)}((\sigma_{S\geq\underline{ST}\wedge S\leq\underline{ST}\wedge D\geq\underline{DT}\wedge D\leq\underline{DT}})(\sigma F_{COUNT(S))>\underline{SB}}\ (FR)))$$

G: Group	F: Function	$_S$: SOURCE_PORT
$_D$: DESTINATION_PORT	$_P$: IP_PROTOCOL	$_T$: IN_BYTES/IN_PKTS
FR: FLOW_RAW_DATA	$\underline{SB}$: COUNT(SRC_PORT)	$\underline{ST}$: SRC_PORT
$\underline{DT}$: DST_PORT		

Based on the SRC_PORT and DST_PORT of the highest frequency detected in Step 1, the flow pattern and the number of times communicated through the corresponding port are derived as the pattern type. SRC_PORT and DST_PORT are initially selected by assigning a range, and then, as learning proceeds, the range is narrowed by directly selecting the target.

Step 3 step relational algebra—anomaly extraction relational algebra

$$\pi_{W,A,R,S,D,P,B,K,T}((\sigma_{S\geq\underline{ST}\wedge S\leq\underline{ST}\wedge D\geq\underline{DT}\wedge D\leq\underline{DT}\wedge\sim T(\underline{G})})(FR))$$

G: Group	F: Function	$_W$: FIRST_SWITCHED
$_A$: SOURCE_ADDRESS	$_R$: DESTINATION_ADDRESS	$_S$: SOURCE_PORT
$_D$: DESTINATION_PORT	$_P$: IP_PROTOCOL	$_B$: IN_BYTES
$_K$: IN_PACKETS	$_T$: IN_BYTES/IN_PACKETS	FR: FLOW_RAW_DATA
$\underline{ST}$: SRC_PORT	$\underline{DT}$: DST_PORT	$\underline{G}$: ROUND((IN_BYTES/IN_PKTS), 2)

The time of occurrence, protocol, packet, and bytes of traffic which is not the pattern derived in Step 2 are derived in Step 3. The source port and destination port and pattern are inputted; the record that did not communicate with the entered pattern among the communication through the corresponding port is determined, and it is read as an abnormal time point to be observed.

2.2.3. Anomaly Pattern Data Extraction Process

Due to the nature of the control network, it is difficult to reproduce the failure process arbitrarily. As shown in Figures 7 and 8, the model is verified by composing the extraction and verification process of the anomaly pattern.

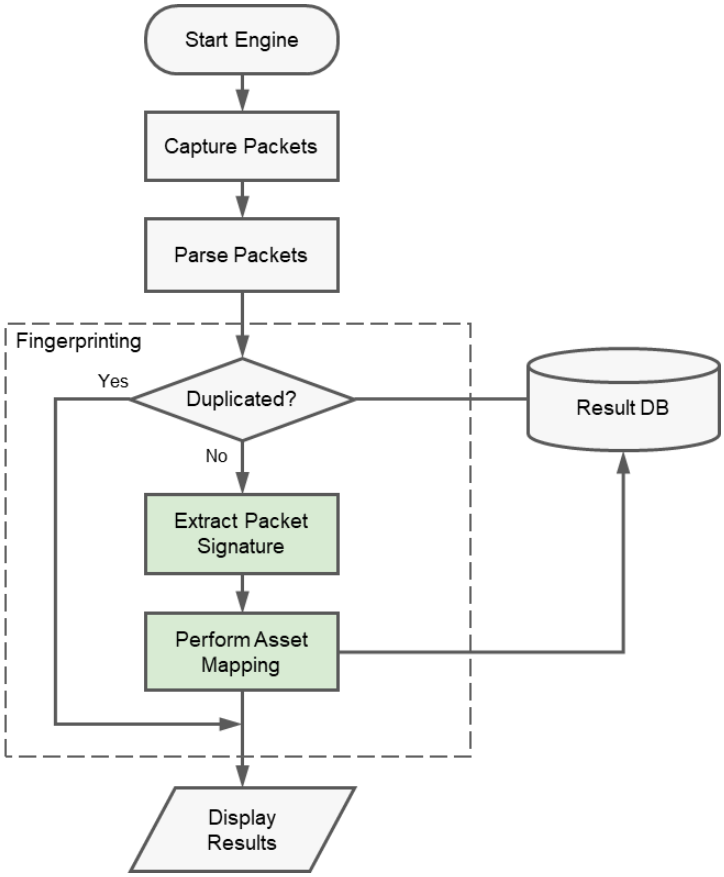

Figure 7. Detection process for the test environment packet.

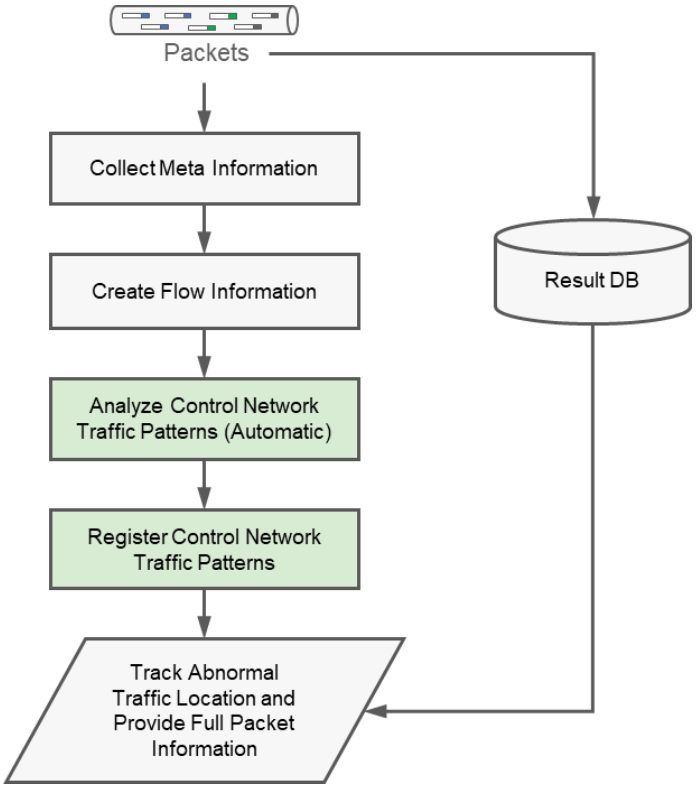

Figure 8. Detection process for the test environment data.

2.2.4. Abnormal Sign Judgement Rate

After the actual traffic within the power plant is learned, the learning results are compared and analyzed in preparation for the actual situation. The false-positive rate is used as an index for the detection rate as a basic measurement standard, and the false-positive rate method used in the national defense network, which is a closed network of a similar type, is applied to confirm whether the false-positive rate applies to the control network [43].

For anomalies, the confusion matrix in Table 2 is applied. A confusion matrix (error matrix) is a matrix for comparing the predicted values and actual values to measure the prediction performance through learning (describing the performance of a classification model). The evaluation criteria are determined according to the criteria in Figure 7 [44,45]. The components of the confusion matrix are the same as the evaluation criteria in Table 3.

Table 2. Confusion matrix.

	Normal (Actual Negative)	**Abnormal (Actual Positive)**
Normal (Predicted Negative)	True Negative (TN)	False Negative (FN)
Abnormal (Predicted Positive)	False Positive (FP)	True Positive (TP)

Table 3. Evaluation criteria.

Criteria	**Equation**	**Description**
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$	Ratio of correct predictions (both true positives and true negatives) among all predictions
Precision	$\frac{TP}{TP+FP}$	Ratio of true positives (anomalies properly detected) among all samples predicted as positive
Sensitivity	$\frac{TP}{TP+FN}$	Ratio of actual positives, which are actually anomalies, predicted as positives
Fall-out	$\frac{FP}{TN+FP}$	Ratio of negative samples (non-anomalies) that are incorrectly classified as positive
F1 Score	$\frac{2TP}{2TP+FP+FN}$	As an index combining precision and sensitivity, it is a value for how accurately the normal is identified and has a relatively high value when it is not biased to either side

The components of the confusion matrix are the same as those of the evaluation criteria in Table 3.

2.2.5. Comparison of the Proposed Models

Table 4 shows the results of the comparative analysis of the proposed model system with a comparable solution, Darktrace.

Table 4. Detection model comparison.

Item Name	Proposed Model	Darktrace
Data Collection Unit	Bytes, Sessions	Sessions
Detection Technique	Network anomaly detection, illegal asset detection	Rule-based anomaly detection
Traffic Analysis	Detailed analysis using packet and session data	Session data only
Asset Detection	Unregistered, new, changed assets	New assets only
Asset Management	Asset registration/modification/change	Automatic registration only
Abnormal traffic analysis	Identification of anomalies using traffic thresholds for each asset and analysis of abnormal traffic using ML techniques for collected traffic	Specific rule-based detection of collected traffic using ML (unsupervised) method

3. Results

To apply the proposed ML supervised learning (decision tree) technique and configure the generated data similar to the demonstration environment, control traffic data from the power plant were collected, and the technology was verified.

Based on the proposed anomaly pattern extraction model, in the first step, the abnormal symptom detection model was trained by extracting whether anomalies were detected from the 6.5 TB of traffic generated in January 2021.

For the two-step verification, the model was verified through a traffic simulator, and 100 random anomaly datasets were generated as 26 TB of data from January 2021 to April 2021, and the detection rate was analyzed and verified over the time series.

To verify the proposed model, the results of the verification were compared with commercial solutions such as Darktrace and improved Darktrace.

3.1. Test Environment Configuration

The test environment for this study was carefully configured to accurately reproduce the actual environment of a power plant. The specifications for the test environment are shown in Table 5, and the configuration is shown in Figure 9. Past operational data were also implemented in the test environment to ensure accurate testing.

Table 5. Software configuration of the test environment.

Classification	Version
OS (Operation System)	Ubuntu 16.04 LTS
DB	MariaDB 10.3.14
WAS (Web Application Server)	Tomcat 8.0.32
JDK	Java 1.8.211
WEB	Apache 2.4.39
C Compiler	GCC 5.4.0
Development Tool	Spring Framework 4.3

The hardware configuration of the test environment is shown in Table 6.

Table 6. Hardware configuration of the test environment.

Type	Requirements
CPU (Central Processing Unit)	2.2 GHZ (Gigahertz)
Core	10core
Memory	64 GB (Gigabyte)
NIC (Network Interface Card)	1 Giga NIC × 4port
SSD (Solid State Drive)	512 GB
HDD (Hard Disk Drive)	10 TB

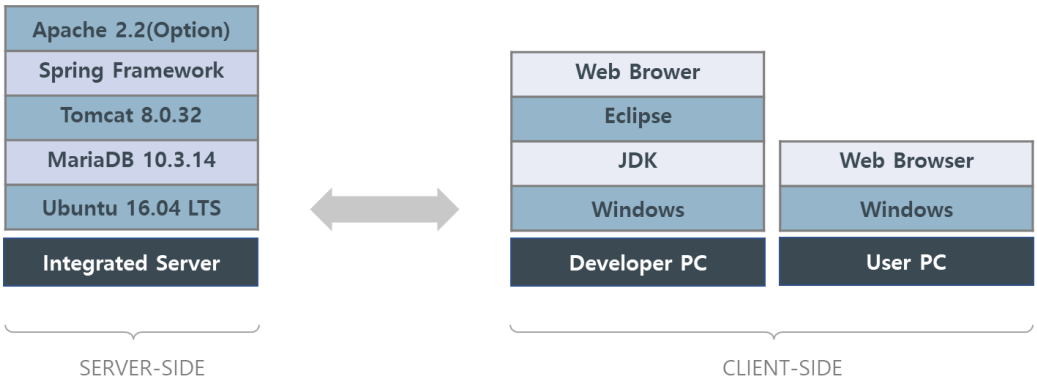

Figure 9. Configuration diagram of the test environment.

The network configuration of the test environment is shown in Table 7.

Table 7. Network configuration of the test environment.

Port	Default	Description	Direction
Console	TCP 8443	Web UI (User Interface) access port	Inbound browser → server
FTP Client	TCP 22	SFTP (Secure File Transfer Protocol)	Inbound/Outbound
Console	TCP 80	SSL (Secure Socket Layer) encrypted communication uses port 443	Inbound
Agent Download	TCP 88	Agent downloads support port for compatibility with IE (Internet Explorer) 8	Inbound Browser → Server
Syslog	UDP 514	Syslog	Outbound
SNMP	162	SNMP (Simple Network Management Protocol)	Outbound

The server communication settings of the test environment are shown in Table 8.

Table 8. Server communication settings.

Function	URL (Uniform Resource Locator)
Web console access	HTTPS (Hyper Text Transfer Protocol over Secure Socket Layer)://<<server name or IP address>>:8443

Figure 10 shows the system configuration for the proposed test environment. It includes a closed structure control network with a functional structure for registering and investigating abnormal symptoms that occur in the actual operating environment.

The test was run using 26 terabytes of control network traffic data collected for 4 months, and random traffic was generated in the existing training data to test the anomaly detection model.

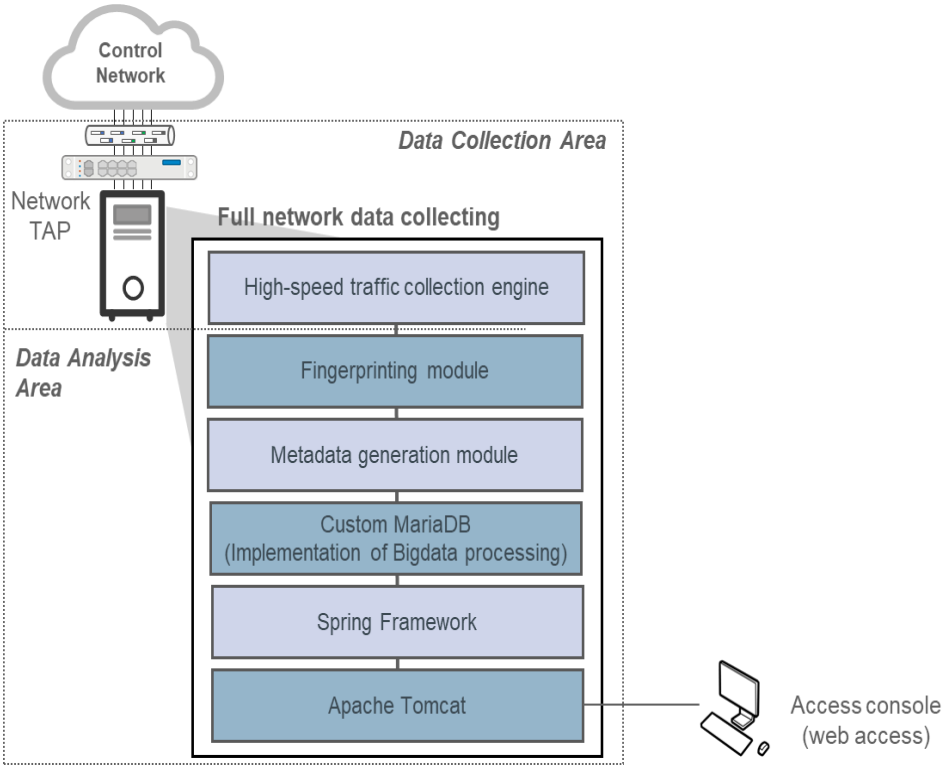

Figure 10. Test environment configuration.

3.2. Test Verification

The test verification in this study had two stages.

The first stage involved the verification of the generator engine, which included the extraction of fingerprinting confirmation, metadata generation, metadata storage performance, and metadata search performance. This was completed using the collected control network data.

The second stage involved verifying the traffic data of 100 randomly selected points based on the actual data. This stage aimed to test the detection rate of the proposed model for anomalies.

3.2.1. Generator Engine Performance Verification

The test environment was verified through several performance tests, including fingerprinting confirmation, metadata generation, metadata storage speed measurement, and metadata search speed measurement, as shown in Table 9 [40].

Table 9. Test items.

Seq	Test Item	Test Goal
1	Fingerprinting success rate	Measurement of OS information extraction success rate (%) from the fingerprinting of the packets
2	Metadata creation success rate	Measurement of the metadata creation success rate (%)
3	Metadata storage speed	Check the speed of saving the PCAP (Packet Capture) file with the metadata stored as a JSON (JavaScript Object Notation) file
4	Metadata retrieval speed	Check the metadata retrieval speed

Test verification was confirmed with a four-step procedure as follows.

The verification process involved several steps, such as comparing the results of Wireshark by fingerprinting the traffic generated by Tcpreplay, measuring the rate of traffic

and metadata generation, and checking the communication metadata fast storage test command and search query execution.

The Telecommunications Technology Association (TTA) of the Republic of Korea certified the results of each of the five tests performed, achieving a 100% extraction success rate and 100% metadata creation success rate. For the PCAP (Packet Capture) file, the metadata creation speed was 21596.603 flows per second, and the metadata search speed was 0.0005492 ms.

In the first experimental result, the accuracy of fingerprinting extraction was measured by comparing the number of SYN packets and SYN + ACK packets and confirming their consistency.

As shown in Table 10, it was confirmed that the sum of TCP packets and SYN + ACK TCP packets was the same as the result of the fingerprinted packets, validating the accuracy of the fingerprinting extraction.

Table 10. Test results for the fingerprinting extraction success rate.

Item	1	2	3	4	5
Processed Packets	1,004,099	1,004,105	1,004,097	1,004,099	1,004,100
Processed TCP Packets	964,819	964,819	964,819	964,819	964,819
Captured SYN TCP Packets	27,761	27,761	27,761	27,761	27,761
Duped packets	2030	2030	2030	2030	2030
Normal Packets	27,761	27,761	27,761	27,761	27,761
Processed SYB TCP Packets	27,761	27,761	27,761	27,761	27,761
Processed SYN_ACK TCP Packets	25,844	25,844	25,844	25,844	25,844
Fingerprinted Packets	53,605	53,605	53,605	53,605	53,605
Processed UDP Packets	35,571	35,571	35,571	35,571	35,571
Elapsed Time	15.641558	22.225767	12.28054	15.896778	16.937732
Packet Process Speed (packets/s)	64,194.30698	45,177.51839	81,763.26262	63,163.67994	59,281.84575
Fingerprinting Speed (packets/s)	3427.088191	2411.840269	4365.03641	3372.066961	3164.827548

In the second experimental result, metadata generation accuracy was verified by comparing and confirming the number of metadata flows, as shown in Table 11. The simulator was also verified by checking that the 636 flows generated by the source terminal were correct.

The third experimental result measured the speed of metadata storage, and the result is shown in Table 12. The generated flows per second were divided by the evaluation time to verify the metadata storage speed.

Table 11. Experiment results for the metadata creation success rate.

Item	1	2	3	4	5
Processed Packets	100,4094	1,004,094	1,004,093	1,004,093	1,004,096
Processed TCP Packets	964,819	964,819	964,819	964,819	964,819
Captured syn TCP Packets	27,761	27,761	27,761	27,761	27,761
Duped Packets	2030	2030	2030	2030	2030
Normal Packets	27,761	27,761	27,761	27,761	27,761
Processed SYN TCP Packets	27,761	27,761	27,761	27,761	27,761
Processed SYN_ ACK TCP Packets	25,844	25,844	25,844	25,844	25,844
Fingerprinted Packets	53,605	53,605	53,605	53,605	53,605
Processed UDP Packets	35,571	35,571	35,571	35,571	35,571
Generated Flows	636	636	636	636	636
Elapsed Time	10.316506	11.188549	10.172895	9.555932	12.131023
Packet Process Speed (packets/s)	97,328.88441	89,743.00147	98,702.78314	105,075.3616	82,770.92337
Flow Generating Speed (flows/c)	61.64878	56.84383	62.519079	66.555518	52.427564
Fingerprinting Speed (packets/s)	5196.042252	4791.058998	8269.395056	5609.604647	4418.835796

The fourth experimental result measured the speed of metadata search, and the results are shown in Table 13.

As a result of the operational test for the test environment, the extraction check of the experimental data showed that it was normally extracted at the time point.

3.2.2. Analysis of the Feature Verification Results

To verify the proposed method, the operational data of a power plant was simulated for 4 months in a real environment, and the performance aspect of detecting anomalous traffic was evaluated. The verification process consisted of two stages: stage 1 involved verifying the method with a simulator, and stage 2 involved deriving the values of K and T to test the hypothesis.

To determine the type of traffic size per packet generated in the control network, the frequency of occurrence according to the type of traffic size generated from SRC_PORT to DST_PORT in the experimental data was measured and shown in Figure 11. The results showed that the traffic size per packet was classified into eight cases, and the communication pattern was defined as the analysis target.

Table 12. Test results for the metadata storage speed.

Item	1	2	3	4	5
Processed Packets	1,000,000	1,000,000	1,000,000	1,000,000	1,000,000
Processed TCP Packets	121,248	121,248	121,248	121,248	121,248
Captured SYN TCP Packets	4576	4576	4576	4576	4576
Duped Packets	562	562	562	562	562
Lost Packets	0	0	0	0	0
Normal Packets	4576	4576	4576	4576	4576
Processed SYN TCP Packets	4576	4576	4576	4576	4576
Processed SYN _ ACK TCP Packets	2693	2693	2693	2693	2693
Fingerprinted Packets	7269	7269	7269	7269	7269
Processed UDP Packets	842,217	842,217	842,217	842,217	842,217
Generated Flows	30,670	30,670	30,670	30,670	30,670
Elapsed Time	1.428743	1.422007	1.403228	1.425827	1.421129
Packet Process Speed (packets/s)	699,916.043	703,231.3445	712,642.3966	701,347.1469	703,665.6886
Flow Generating Speed (flows/s)	21,466.42504	21,568.10534	21,856.7423	21,510.317	21,581.42667
Fingerprinting Speed (packets/s)	5087.689716	5111.788643	5180.197571	5098.092411	5114.945891

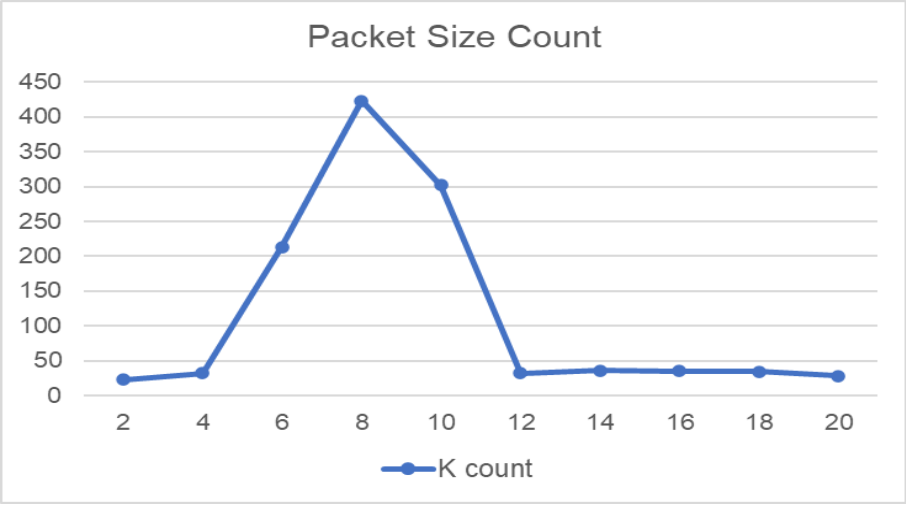

Figure 11. Packet size count of the proposed model.

Table 13. Test results for the metadata search speed.

Item	1	2	3	4	5
Source IP	172.15.100.12	172.15.100.12	172.15.100.12	172.15.100.12	172.15.100.12
Source Port	2916	2916	2916	2916	2916
Bytes	11,541,934	11,541,934	11,541,934	11,541,934	11,541,934
Data Count	100	100	100	100	100
Query ID	1	1	1	1	1
Duration	0.00224869	0.00011751	0.00014457	0.00011805	0.00011718

The threshold for the occurrence frequency of type K of traffic size per packet, which is the basis for pattern selection, was measured and is shown in Figure 12. It was found that when traffic occurs from SRC_PORT to DST_PORT more than 400 times, it can be defined as having a pattern and set as a boundary value. Therefore, the threshold value T was set to 400 as the reference value for traffic detection.

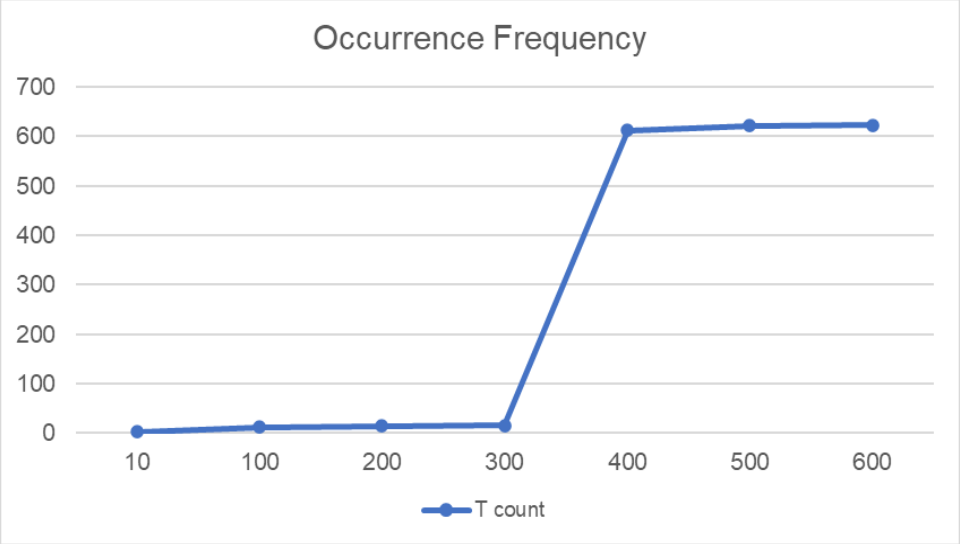

Figure 12. Occurrence frequency of the proposed model.

Based on the real data, the thresholds for K and T were derived as 8 and 400, respectively, through the verification process.

To diagnose anomalies, it is necessary to identify a pattern in the traffic that exceeds a certain communication frequency threshold K. Based on the results of the analysis, it was found that the majority of communications meeting this threshold had a packet size of 8 or less, which supports the alternative hypothesis H1 proposed by DMP (Detection Model P) and rejects the null hypothesis H0.

3.2.3. Analysis of the Verification Results of the Proposed Model

The proposed model, DMP, was compared and verified with the solutions from previous experiments, DT and i-DT. Verification was performed using traffic data extracted from 100 points in an arbitrary section of the 4-month collection of operational data. The verification was performed five times on a total of 26 TB collected to determine whether abnormal symptoms were detected. Table 14 presents the results of comparing the reading rates of DT, i-DT, and the proposed model. The results show that the proposed model's detection efficiency is superior to that of the other solutions.

It is worth noting that DT is a solution for traffic detection on an open network, while i-DT is an improvement of the method of detecting only the session count. i-DT could improve the function to detect even the byte unit to optimize it for a closed network environment.

The proposed model in this study showed significant improvements compared to the existing method. As indicated in Figure 13, the spy rate, which distinguishes normal and anomalous traffic, was 89.2%, which was 1,064% and 84% better than the existing method, respectively. Additionally, the false-positive rate for problematic abnormal signs was 8.4%, representing an improvement of 961% and 419% compared to the existing model, respectively. In addition, precision, sensitivity, and F1 score were all improved compared to those of the traditional methods. The detection processing time was also faster than that of the existing model.

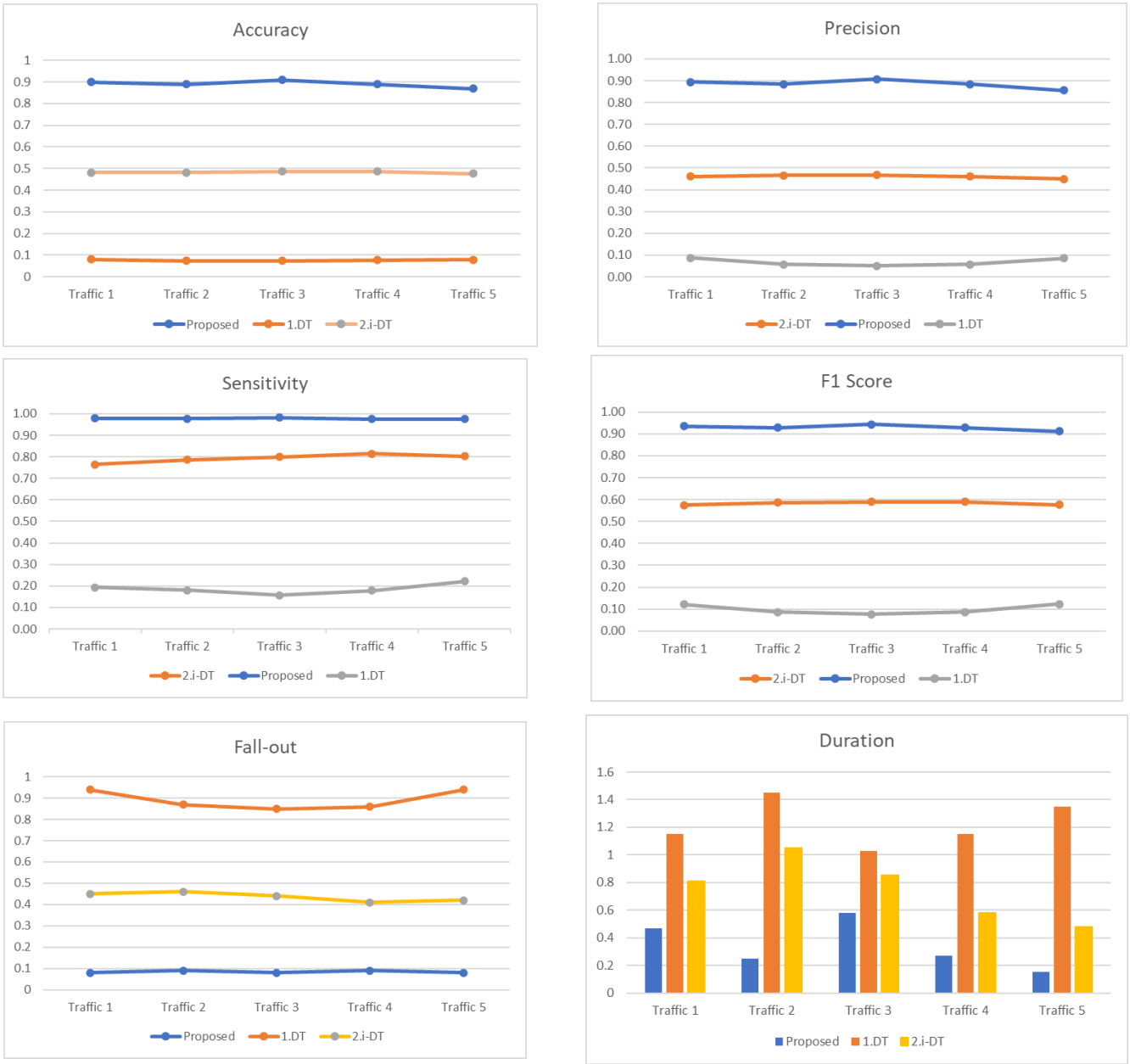

Figure 13. Proposed model detection rate compared to that of the existing method.

It is important to note that DT, which is optimized for detecting anomalies in a general network, detected almost no anomalies in the private control network protocol. However, for a fair comparison, the proposed model's reading rate of anomalies was found to be higher than that of i-DT, which was tuned for a closed network environment. This shows the effectiveness of the proposed model in detecting anomalies in a closed network with a private protocol.

Table 14. Results of the traffic generation experiment.

Traffic	Flows	Techniques	Accuracy	Precision	Sensitivity	F1 Score	Fall-out	Duration
1	21,596	1.DT	0.08	0.088	0.193	0.121	0.94	1.15 s
		2.i-DT	0.482	0.46	0.764	0.574	0.45	0.81 s
		Proposed	0.9	0.895	0.979	0.935	0.08	0.47 s
2	21,592	1.DT	0.073	0.057	0.181	0.086	0.87	1.45 s
		2.i-DT	0.482	0.466	0.787	0.586	0.46	1.05 s
		Proposed	0.89	0.885	0.978	0.929	0.09	0.25 s
3	21,594	1.DT	0.074	0.05	0.157	0.076	0.85	1.03 s
		2.i-DT	0.486	0.467	0.8	0.589	0.44	0.85 s
		Proposed	0.91	0.908	0.983	0.944	0.08	0.58 s
4	21,592	1.DT	0.077	0.057	0.178	0.087	0.86	1.15 s
		2.i-DT	0.487	0.461	0.814	0.589	0.41	0.58 s
		Proposed	0.89	0.885	0.975	0.928	0.09	0.27 s
5	21,593	1.DT	0.079	0.085	0.221	0.123	0.94	1.35 s
		2.i-DT	0.477	0.45	0.803	0.576	0.42	0.48 s
		Proposed	0.87	0.856	0.976	0.915	0.08	0.15 s

4. Discussion

In Section 2.2, we present a decision tree-based model for anomaly detection in the control network of a power plant. We could have provided more details on how decision trees work and how we applied them in our study. Specifically, decision trees are a type of supervised learning algorithm that recursively partition data based on the most significant features to make a series of binary decisions that lead to a prediction or classification. In our study, we used decision trees to classify traffic as normal or abnormal based on a set of features, such as packet size, port number, and protocol type. We trained the decision tree using a labeled dataset and tested it on unseen data to evaluate its performance.

Our experimental results showed that the decision tree-based model achieved superior detection efficiency, a lower false-positive rate, and a faster detection processing time compared to those of existing solutions. This suggests that applying machine learning to the control network of power plants is crucial for ensuring system stability in unpredictable environments. The proposed method also allows for the early detection of anomalies, which can prevent human error and the normalization of abnormalities.

We acknowledge that in a real-world operating environment, additional indicators and verification factors need to be applied to improve anomaly detection. However, the proposed method enables the separate collection and analysis of control system network data without affecting the control system's operation, facilitating the control of abnormal symptoms.

5. Conclusions

In conclusion, this paper presents a decision tree-based model for detecting illegal assets and abnormal traffic in the control network of a power plant. Our results demonstrated the effectiveness of the proposed model in detecting anomalies in the control network of power plants. We also highlighted the importance of applying machine learning to ensure system stability in unpredictable environments.

Future research should focus on comparing various artificial intelligence algorithms for extracting traffic generation reference points and determining packet sizes to determine communication patterns. Additionally, it is necessary to investigate the possibility of using digital forensic techniques to check for data forgery or falsification using deep learning algorithms to increase the accuracy and detection performance of anomaly detection.

Author Contributions: Conceptualization, K.-M.L. and K.-H.L.; methodology, K.-M.L. and M.-Y.C.; software, J.-G.K.; validation, K.-M.L. and M.-Y.C.; formal analysis, K.-M.L.; investigation, M.-Y.C.;

resources, K.-M.L.; data curation, J.-G.K.; writing—original draft preparation, K.-M.L.; writing—review and editing, K.-H.L.; visualization, J.K; supervision, K.-H.L.; project administration, K.-H.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used in this study are actual power plant operation data. Research data cannot be provided when requested, and only a brief description of the data status can be provided.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mohamad, H.; Mokhlis, H.; Bakar, A.H.A.; Ping, H.W. A review on islanding operation and control for distribution network connected with small hydro power plant. *Renew. Sustain. Energy Rev.* **2011**, *15*, 3952–3962. [CrossRef]
2. Mahmood, A.N.; Leckie, C.; Hu, J.; Tari, Z.; Atiquzzaman, M. Network traffic analysis and SCADA security. In *Handbook of Information and Communication Security*, 1st ed.; Stavroulakis, P., Stamp, M., Eds.; Springer: Berlin, Germany, 2010; Volume 1, pp. 383–405.
3. Lippmann, R.; Fried, D.; Piwowarski, K.; Streilein, W. Passive operating system identification from TCP/IP packet headers. In Proceedings of the Workshop on Data Mining for Computer Security, Melbourne, FL, USA, 19–22 November 2003; pp. 40–49.
4. Shim, K.S.; Goo, Y.H.; Lee, S.H.; Sija, B.D.; Kim, M.S. Automatic payload signature update system for classification of recent network applications. *J. Korean Inst. Commun. Inf. Sci.* **2017**, *42*, 98–107. [CrossRef]
5. Goo, Y.H.; Choi, S.O.; Lee, S.K.; Kim, S.M.; Kim, M.S. Tracking the source of cascading cyber attack traffic using network traffic analysis. *Korean Inst. Commun. Inf. Sci.* **2016**, *41*, 1771–1779.
6. Narayan, J.; Shukla, S.K.; Clancy, T.C. A survey of automatic protocol reverse engineering tools. *ACM Comput. Surv.* **2015**, *48*, 1–26. [CrossRef]
7. Borisov, N.; Brumley, D.; Wang, H.J.; Dunagan, J.; Joshi, P.; Guo, C. Generic application-level protocol analyzer and its language. In Proceedings of the NDSS Symposium 2007: The Network and Security Conference, San Diego, CA, USA, 28 February–2 March 2007; pp. 1–13.
8. Caselli, M.; Hadžiosmanović, D.; Zambon, E.; Kargl, F. On the feasibility of device fingerprinting in industrial control systems. In Proceedings of the 8th International Workshop on Critical Information Infrastructures Security, Amsterdam, The Netherlands, 16–18 September 2013; pp. 155–166.
9. Winston, P.H. *Artificial intelligence*, 3rd ed.; Addison-Wesley Longman Publishing Co., Inc.: Boston, MA, USA, 1992; pp. 1–737.
10. Matsuda, W.; Fujimoto, M.; Aoyama, T.; Mitsunaga, T. Cyber security risk assessment on industry 4.0 using ICS testbed with AI and Cloud. In Proceedings of the 2019 IEEE Conference on Application, Information and Network Security (AINS), Pulau Pinang, Malaysia, 19–21 November 2019; pp. 54–59.
11. Yılmaz, E.N.; Sayan, H.H.; Üstünsoy, F.; Gönen, S.; Sindiren, E.; Karacayılmaz, G. ICS Cyber attack analysis and a new diagnosis approach. In Proceedings of the ICAIAME 2019: The International Conference on Artificial Intelligence and Applied Mathematics in Engineering, Manavgat, Antalya, Turkey, 20–22 April 2019; pp. 127–141.
12. Li, J.H. Cyber security meets artificial intelligence: A survey. *Front. Inf. Technol. Electron. Eng.* **2018**, *19*, 1462–1474. [CrossRef]
13. Veiga, A.P. Applications of artificial intelligence to network security. *arXiv* **2018**, arXiv:1803.09992.
14. Frank, J. Artificial intelligence and intrusion detection: Current and future directions. In Proceedings of the 17th National Computer Security Conference, Baltimore, MD, USA, 11–14 October 1994; pp. 1–12.
15. Keller, J.M.; Gray, M.R.; Givens, J.A. A fuzzy k-nearest neighbor algorithm. *IEEE Trans. Syst. Man Cybern.* **1985**, *SMC-15*, 580–585. [CrossRef]
16. Pavlidis, P.; Wapinski, I.; Noble, W.S. Support vector machine classification on the web. *Bioinformatics* **2004**, *20*, 586–587. [CrossRef]
17. Safavian, S.R.; Landgrebe, D. A survey of decision tree classifier methodology. *IEEE Trans. Syst. Man Cybern.* **1991**, *21*, 660–674. [CrossRef]
18. Pal, M. Random forest classifier for remote sensing classification. *Int. J. Remote Sens.* **2005**, *26*, 217–222. [CrossRef]
19. Schonlau, M.; Zou, R.Y. The random forest algorithm for statistical learning. *Stata J.* **2020**, *20*, 3–29. [CrossRef]
20. Weichenthal, S.; Van Ryswyk, K.; Goldstein, A.; Bagg, S.; Shekkarizfard, M.; Hatzopoulou, M. A land use regression model for ambient ultrafine particles in Montreal, Canada: A comparison of linear regression and a machine learning approach. *Environ. Res.* **2016**, *146*, 65–72. [CrossRef]
21. Maulud, D.H.; Abdulazeez, A.M. A review on linear regression comprehensive in machine learning. *J. Appl. Sci. Technol. Trends* **2020**, *1*, 140–147. [CrossRef]
22. Böhning, D. Multinomial logistic regression algorithm. *Ann. Inst. Stat. Math.* **1992**, *44*, 197–200. [CrossRef]
23. Xu, R.; Wunsch, D. Survey of clustering algorithms. *IEEE Trans. Neural Netw.* **2005**, *16*, 645–678. [CrossRef]

24. Hohman, F.; Wongsuphasawat, K.; Kery, M.B.; Patel, K. Understanding and visualizing data iteration in machine learning. In Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems, Honolulu, HI, USA, 25–30 April 2020; pp. 1–13.
25. Weinberger, K.Q.; Saul, L.K. An introduction to nonlinear dimensionality reduction by maximum variance unfolding. In Proceedings of the Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference, Boston, MA, USA, 16–20 July 2006; pp. 1683–1686.
26. Thabtah, F.; Cowling, P.; Peng, Y. MCAR: Multi-class classification based on association rule. In Proceedings of the 3rd ACS/IEEE International Conference on Computer Systems and Applications, Cairo, Egypt, 6 January 2005; pp. 33–39.
27. Yoon, S.; Kim, K.J. Deep Q networks for visual fighting game AI. In Proceedings of the 2017 IEEE Conference on Computational Intelligence and Games (CIG), New York, NY, USA, 22–25 August 2017; pp. 306–308.
28. Chen, T.; Niu, W.; Xiang, Y.; Bai, X.; Liu, J.; Han, Z.; Li, G. Gradient band-based adversarial training for generalized attack immunity of A3C path finding. *arXiv* **2018**, arXiv:1807.06752.
29. Scikit-Learn (Choosing the Right Estimator). Available online: https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html (accessed on 2 February 2023).
30. Auffret, P. SinFP, unification of active and passive operating system fingerprinting. *J. Comput. Virol.* **2010**, *6*, 197–205. [CrossRef]
31. Beverly, R. A robust classifier for passive TCP/IP fingerprinting. In Proceedings of the Passive and Active Network Measurement: 5th International Workshop, PAM 2004, Juan-Les-Pins, France, 19–20 April 2004; pp. 158–167.
32. Conti, G.; Abdullah, K. Passive visual fingerprinting of network attack tool. In Proceedings of the VizSEC/DMSEC '04: 2004 ACM Workshop on Visualization and Data Mining for Computer Security, Washington, DC, USA, 29 October 2004; pp. 45–54.
33. Laštovička, M.; Filakovský, D. Passive OS fingerprinting prototype demonstration. In Proceedings of the NOMS 2018—2018 IEEE/IFIP Network Operations and Management Symposium, Taipei, Taiwan, 23–27 April 2018; pp. 1–2.
34. Barbosa, R.R.R.; Sadre, R.; Pras, A. Flow whitelisting in SCADA networks. *Int. J. Crit. Infrastruct. Prot.* **2013**, *6*, 150–158. [CrossRef]
35. Mansfield-Devine, S. The promise of whitelisting. *Netw. Secur.* **2009**, *2009*, 4–6. [CrossRef]
36. Erickson, D.; Casado, M.; McKeown, N. The Effectiveness of Whitelisting: A User-Study. In Proceedings of the CEAS 2008: The Fifth Conference on Email and Anti-Spam, Mountain View, CA, USA, 21–22 August 2008; pp. 1–10.
37. Pareek, H.; Romana, S.; Eswari, P.R.L. Application whitelisting: Approaches and challenges. *Int. J. Comput. Sci. Eng. Inf. Technol.* **2012**, *2*, 13–18. [CrossRef]
38. Perales Gómez, Á.L.; Fernández Maimó, L.; Huertas Celdrán, A.; García Clemente, F.J. Madics: A methodology for anomaly detection in industrial control systems. *Symmetry* **2020**, *12*, 1583. [CrossRef]
39. Mokhtari, S.; Abbaspour, A.; Yen, K.K.; Sargolzaei, A. A machine learning approach for anomaly detection in industrial control systems based on measurement data. *Electronics* **2021**, *10*, 407. [CrossRef]
40. Abdelaty, M.; Doriguzzi-Corin, R.; Siracusa, D. DAICS: A deep learning solution for anomaly detection in industrial control systems. *IEEE Trans. Emerg. Top. Comput.* **2021**, *10*, 1117–1129. [CrossRef]
41. Wang, C.; Wang, B.; Liu, H.; Qu, H. Anomaly detection for industrial control system based on autoencoder neural network. *Wirel. Commun. Mob. Comput.* **2020**, *2020*, 8897926. [CrossRef]
42. Cao, Y.; Zhang, L.; Zhao, X.; Jin, K.; Chen, Z. An Intrusion Detection Method for Industrial Control System Based on Machine Learning. *Information* **2022**, *13*, 322. [CrossRef]
43. Sommer, R.; Paxson, V. Outside the closed world: On using machine learning for network intrusion detection. In Proceedings of the 2010 IEEE Symposium on Security and Privacy, NW Washington, DC, USA, 16–19 May 2010; pp. 305–316.
44. Townsend, J.T. Theoretical analysis of an alphabetic confusion matrix. *Percept. Psychophys.* **1971**, *9*, 40–50. [CrossRef]
45. Visa, S.; Ramsay, B.; Ralescu, A.L.; Van Der Knaap, E. Confusion matrix-based feature selection. In Proceedings of the MAICS 2011: 22nd Midwest Artificial Intelligence and Cognitive Science Conference, Cincinnati, OH, USA, 16–17 April 2011; pp. 120–127.