IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 19 March 2024, accepted 6 April 2024, date of publication 10 April 2024, date of current version 17 April 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3387016

RESEARCH ARTICLE

An Advanced Approach for Detecting Behavior-Based Intranet Attacks by Machine Learning

MYONGWON JANG AND KYUNGHO LEE
School of Cybersecurity, Korea University, Seoul 02841, South Korea
Corresponding author: Kyungho Lee (kevinlee@korea.ac.kr)

ABSTRACT To address continuously increasing cyber threats, security professionals within organizations are fortifying internal security by implementing security policies such as network segregation and emerging concepts such as Zero Trust. However, despite these changes in the cybersecurity landscape, the ultimate goal of cyber attackers, which is to exfiltrate critical information stored within an organization's intranet, remains unchanged. Consequently, attackers with motives such as hacktivists persistently and repeatedly target key systems within an organization's intranet to achieve their ultimate objectives. Considering the tendencies of intranet attackers, this study proposes the inclusion of the number of connection attempts for attack detection as an additional attribute alongside commonly used attributes such as source IP, destination IP, protocol, and attack signatures in intrusion detection rules. This proposal is supported by establishing an experimental environment for conducting intranet attacks and collecting raw data. Using feature engineering techniques, the raw data were transformed into analyzable datasets, and the performance was measured using six supervised machine learning algorithms. Through this research, we aim to contribute to the field of cybersecurity by going beyond the conventional focus on Internet-based attacks and providing a methodology for analyzing various intranet-based attacks in a post-stage environment. In addition, we share the method of feature engineering Zeek IDS raw data and release the resulting dataset to further advance the field. We hope that these contributions will foster future developments in this domain.

INDEX TERMS Cybersecurity, intranet attack, Zeek IDS, feature engineering (FE), machine learning (ML).

I. INTRODUCTION

As cyberspace expands and evolves, the sophistication and intelligence of Advanced Persistent Threat (APT) attacks are becoming a severe problem for organizations [1]. The current level of APT attacks has evolved beyond simply infiltrating computers in the internet domain to steal credentials in single-step attacks and exfiltrate data. These attacks use compromised internet computers as a base for post-stage attacks, penetrating internal core systems within the intranet to either exfiltrate data or render systems inoperable [2], [3]. In response to these attack techniques, system administrators strengthen their security by implementing policies such as physical network segmentation and adopting the concept of Zero Trust, which involves not trusting any system other than their own [4]. However, attackers continue to research and attempt various methods to breach core systems, which are their ultimate targets [5]. An important aspect to consider in this situation is that just as the use of auxiliary storage devices, such as USB drives, for infiltration attempts increased when network isolation policies were strengthened [6], the implementation of stronger Zero Trust policies is likely to lead to an increase in the number of reconnaissance and attack attempts by intranet attackers.

This study proposes and validates the hypothesis that adding the attribute of cumulative connection attempts made by a compromised computer to infiltrate other computers can enhance the detection of intranet attacks. To prove this hypothesis, we first transform log data recorded by the Zeek Intrusion Detection System (IDS) in an experimental environment that includes intranet attacks into a dataset through feature engineering (FE). This is achieved

The associate editor coordinating the review of this manuscript and approving it for publication was Kashif Saleem.

by adding the attribute of cumulative connection counts to existing attributes such as source IP, destination IP, protocol, and attack signatures. Then, the performance is evaluated using supervised machine learning (ML) algorithms based on this dataset, and the significance of the added cumulative connection count attribute in detecting intranet attacks is analyzed [7].

The paper is organized as follows. Section II starts with an explanation of the study's purpose and overview, focusing on the research approach, including the methods of data collection and analysis. Section III delves into the research and background knowledge relevant to this work, encompassing Zeek IDS, Feature Engineering, and Machine Learning. Section IV details the procedure for refining Zeek IDS packet logs into a dataset ready for Machine Learning through Feature Engineering. Section V employs six different supervised learning ML algorithms to enhance a previously created dataset, involving the processes of Feature Extraction and Selection, followed by performance evaluation. Section VI discusses the significance and limitations of this research, and Section VII concludes the paper.

A. OUR CONTRIBUTION

In this study, we aim to analyze post-stage attacks by collecting and processing raw data and then transforming them into a dataset suitable for ML analysis through FE. This study intends to publicize the dataset created as a result of the FE process. Additionally, we provide a methodology that expands the analysis of Zeek IDS logs, which were previously limited to Connection logs and HTTP protocol logs, to include 14 types of protocols that can be exploited in intranet attacks, such as SMB, KERBEROS, SOCKS, and VNC. This expansion has contributed to advancements in cybersecurity technologies.

II. APPROACH

A. GOAL AND OVERVIEW

This research is a study that quantitatively analyzed the theory that the attribute of cumulative connection counts is necessary to improve the detection rate of cyberattacks in an intranet environment. Therefore, this study focuses on detecting secondary and post-attacks initiated by an attacker from a compromised Internet network computer to infiltrate the next target. To achieve this, data pertaining to intranet attacks were separated from the overall network data and processed through FE to extract a dataset suitable for ML analysis. We then performed an analysis using supervised learning ML algorithms based on the created dataset. Figure 1 illustrates the overall research approach.

B. RESEARCH BACKGROUND AND RELIABILITY OF METHODOLOGY

The SolarWinds APT attack in the United States can be cited as an example of an intranet attack case, but the related data are either classified or confidential [8]. As cyberattacks, including Intranet attacks, are becoming more sophisticated, research on such attacks is necessary. In this study, we used data that included some known intranet attacks to increase the reliability by making it closer to actual attacks. Additionally, to maintain objectivity as much as possible, the research began based on the log data of the public IDS and Zeek IDS and compared the results of six verified supervised learning algorithms.

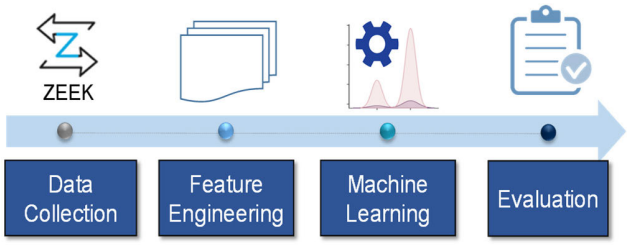

FIGURE 1. Overview of the approach.

C. METHOD OF DATA COLLECTION FOR THE RESEARCH

The research data consisted of transformed log data generated by Zeek IDS, created in an experimental environment that included intranet attacks. While network packets may have different formats for each protocol, the logs generated by Zeek IDS are in text format. The collection of log files followed as objective a procedure as possible, and during feature engineering, the attributes of Zeek IDS were largely retained and transformed.

D. IMPLEMENTATION

The following describes the procedure for transforming raw data collected into a dataset suitable for ML analysis, as part of the methodology to reach the conclusion of the paper. When collecting Zeek IDS log files in the experimental environment, 28 protocol-specific result log files are generated. These files contain both intranet and internet data. Therefore, the first task is to select protocols that are susceptible to intranet attacks from the entire set of protocols and to separate intranet packet data from the selected protocol files. Since we know the internal IP range, this separation can be easily achieved with a simple program. The second task is to separate data by source IP and sort it chronologically. This is done to calculate the accumulated Connection Count in the next step. The third task is to sequentially calculate and record the Connection Count for each IP, which is the core concept of this paper. The fourth task involves applying the intranet Attack Signature to the log data created so far to mark packets with attack signatures. For this, research on attack signatures published by organizations like MITRE ATT&CK, Snort, or The Open Worldwide Application Security Project (OWASP) is necessary. Finally, the last task is to create the dataset required for ML analysis, which involves verifying one by one whether the packets identified as attacks by the Attack Signature are indeed attacks. This is a crucial step

as it forms the basis for the analysis and evaluation through ML algorithms. Therefore, it must be carefully selected as it becomes part of the fair process for analysis and evaluation. Once these series of tasks are completed, a dataset suitable for analysis and evaluation can be obtained.

E. DATA ANALYSIS AND ML ALGORITHMS

The tools and technologies used for data analysis include Python 3.8 and the Scikit-Learn environment [9].

After deciding to use ML for data analysis, significant deliberation is required regarding which algorithms to employ. However, it was challenging to find documents or recommendations explicitly stating that specific ML algorithms should be used for analyzing network packets. Therefore, we strategized to reference prior research on how algorithms were selected, adding what is necessary and removing what is not. One of the most influential studies on the selection of machine learning algorithm criteria is Gustavsson's "Machine Learning For A Network based Intrusion Detection System" [11]. The author of this paper describes the ML techniques intended for use in the "Machine Learning for Intrusion Detection System" section, based on ML algorithms used in previous studies. The representative algorithms mentioned in Gustavsson's article are Naïve Bayes (NB), Support Vector Machine (SVM), Random Forest (RF), Iterative Dichotomiser 3 (ID3, a type of Decision Tree), K-Nearest-Neighbors (KNN), and quadratic discriminant analysis (QDA). In our study, we utilized NB, SVM, RF, and KNN from Gustavsson's mentioned algorithms. We omitted ID3 since RF, which is the Bagging algorithm of DT (Decision Tree), was already present. QDA was replaced with a similar LR algorithm used in related network research papers [13]. Although not used in Gustavsson's paper, we included Gradient Boosting (GB), a Boosting algorithm used in network research papers, to balance the validation of Bagging and Boosting algorithms [7]. In conclusion, the six supervised learning ML algorithms used in this paper are as follows: Logistic Regression (LR), Naïve Bayes (NB), K-Nearest-Neighbors (KNN), Support Vector Machine (SVM), Random Forest (RF), and Gradient Boosting (GB).

F. OVERFITTING AND VALIDATION STRATEGIES

To maintain the stability of the data analysis model, it is necessary to address the issue of overfitting [40], [41]. Strategies to prevent overfitting include increasing the amount of data through augmentation, reducing the complexity of the model, regularization, and dropout. In our experiments, if overfitting occurs, considering that the raw data is in the form of text log data, it is reasonable to address the overfitting issue by increasing the amount of data through augmentation.

Additionally, to ensure the robustness of the model and maintain consistent experimentation, the cross-validation strategy is necessary [47]. Cross-validation (CV) strategies include Validation Set Approach, Leave-One-Out Cross-Validation (LOOCV), and K-Fold method [55]. In this paper, we employ the K-Fold method, which is relatively efficient compared to LOOCV and eliminates the risk of varying results with random value changes like the Validation Set Approach. K-Fold is advantageous for validating a large amount of data as it involves repeatedly dividing the data into subsets and training multiple models. Given the substantial amount of data in network logs, K-Fold is favorable for evaluating performance. The K-Value is determined within the range of 3 to 10 to derive optimal values for speed, stability, and other experimental benefits [56].

FIGURE 2. The comprehensive process of feature Selection.

G. EVALUATION

Even after conducting Overfitting mitigation, Weight optimization, and cross-validation operations followed by Evaluation using ML algorithms, we may not achieve the expected results. This is because the process of Feature Extraction and Selection is necessary. For proper Evaluation, it is essential to go through the Exploratory Data Analysis (EDA) process to understand the overall structure and characteristics of the dataset. Then, based on this, Feature Extraction and Selection tasks should be performed. In this paper, Feature Extraction is conducted to prevent errors that may occur during the encoding process of converting categorical data into numbers. Additionally, Feature Selection is carried out using the backward elimination method, considering the characteristics of the data discovered during the EDA analysis. The specific feature selection criteria and strategies are as follows. There are various methods for feature selection such as Filtered, Wrappers, and Ranked, and there are various algorithms underneath [24], [25]. These algorithms are used to determine the best feature subset and are usually combined with search strategies such as forward selection, backward elimination, bi-directional search, best-first search, and genetic search. We use backward elimination search strategies, for which we select attributes identified as candidates for removal based on EDA results indicating diverse value ranges. Then, among the selected attributes, those with randomly generated values are first removed to evaluate performance. Subsequently, attributes with diverse yet sequential value ranges are removed to assess performance. This process is repeated, and the attributes yielding the highest evaluation results are selected. Figure 2 illustrates this comprehensive process of Feature Selection.

The goal of the evaluation is to find the optimal performance through the process of Feature Extraction and Selection.

TABLE 1. Zeek IDS log files.

Log File	Description
conn.log	TCP/UDP/ICMP connections
dce_rpc.log	Distributed Computing Environment/RPC
dhcp.log	DHCP leases
dnp3.log	DNP3 requests and replies
dns.log	DNS activity
ftp.log	FTP activity
http.log	HTTP requests and replies
irc.log	IRC commands and responses
kerberos.log	Kerberos
modbus.log	Modbus commands and responses
modbus_register_change.log	Tracks changes to Modbus holding registers
mysql.log	MySQL
ntlm.log	NT LAN Manager (NTLM)
ntp.log	Network Time Protocol
radius.log	RADIUS authentication attempts
rdp.log	RDP
rfb.log	Remote Framebuffer (RFB)
sip.log	SIP
smb_cmd.log	SMB commands
smb_files.log	SMB files
smb_mapping.log	SMB trees
smtp.log	SMTP transactions
snmp.log	SNMP messages
socks.log	SOCKS proxy requests
ssh.log	SSH connections
ssl.log	SSL/TLS handshake info
syslog.log	Syslog messages
tunnel.log	Tunneling protocol events

III. BACKGROUND AND RELATED WORK

A. ZEEK IDS

1) ZEEK IDS OVERVIEW

Zeek is an open-source network traffic analysis tool that operates passively. It is used as a Network Security Monitoring (NSM) tool to support investigations of malicious activities [10]. One of the key advantages of Zeek is its customizable platform, which allows users to tailor it through scripting [11]. For packet collection, Zeek's configuration is similar to that of a typical Intrusion Detection System (IDS), employing methods such as a Port Mirror or Test Access Port (TAP) for installation [12]. Zeek generated 28 types of log files based on the protocols listed in Table 1 [10].

Each log file is based on a protocol, and although the items recorded vary for each protocol, the following items are commonly used.

TABLE 2. Zeek IDS common log format.

Field	Description
ts	Timestamp
uid	Unique ID for the connection
id.orig_h	The originator's IP address
id.orig_p	The originator's port number
id.resp_h	The responder's IP address
id.resp_p	The responder's port number
Custom	Custom fields

2) RESEARCH RELATED TO ZEEK IDS

Gustavsson [11] utilized six ML techniques–SVM, KNN, NB, Quadratic Discriminant Analysis (QDA), Decision Tree (DT), and RF–to detect attacks, such as Slowloris DDoS, FTP Brute force, and Port Scanning in Zeek IDS log data based on the CICIDS2017 Dataset. The experimental results demonstrate the accuracy of the algorithms with the best performance: RF (98%), ID3 (98%) - a type of DT, KNN (96%), and QDA (97%). Similarly, Rodríguez e al. [13] extended the analysis of the CICIDS2017 Dataset with ten algorithms - NB, Logistic Regression Model, Multi-Layer Perceptron (MLP), Support Vector Classifier (SMO), KNN, Ada Boost Classifier (AB), 1R Classifier (ONER), Partial C4.5 Decision Tree (PART), C4.5 Decision Tree (J48), and RF - and presented the performance measurement results. However, this study did not conduct cross-validation for the robustness of the model, suggesting that verification is needed for high-performance results. Bagui et al. [7] analyzed conn log data from the University of West Florida 2022 (UWF-ZeekData22) dataset. They employed various algorithms, including Logistic Regression, Naïve Bayes, Random Forest, Gradient Boosted Tree, Decision Tree, and Support Vector Machine, and detailed the parameters for each. Notably, they reported that three algorithms - decision tree, gradient boosting tree, and random forest - demonstrated an accuracy of over 99%. However, there is a significant concern regarding the overfitting of the model in this study, as the analysis was conducted using ML algorithms without standardizing the actual data. Andrews et al. [14] analyzed the HTTP protocol data from six months of Zeek log data collected from a corporate network. They employed a combination of supervised and unsupervised learning algorithms, including Gaussian, Multinomial, Bernoulli, Complement NB, K-Means, L/NL SVM, and KNN, and found that the KNN algorithm showed the highest performance, with an accuracy of 90.3%. However, one limitation of this study is that it focused only on HTTP despite the variety of logs generated by Zeek. Jimenez [15] established a system using RASPBERRYPI IoT devices and analyzed the logs captured by Zeek IDS. The study experimented with TCP, UDP, ICMP, HTTP, FTP, DNS, and SSH protocols using algorithms such as RF, SVM, Neural Networks, and Neurons, and achieved high results with 99% accuracy with algorithms such as RF and SVM. However, this study lacked an explanation of the FE process, leaving unanswered how it addressed the common issue of overfitting in the analysis of cyberattack packets.

TABLE 3. Issues of related research.

Field	Issue	Solution
1	Overfitting issue	Solving through data argumentation.
2	Robustness issue (Cross Validation)	Resolved by K-Fold Cross Validation checking.
3	Paper duplication	There have been no similar or identical studies to this research so far.

Given the high accuracy of the results, there is a concern about potential overfitting in the model. In addition, Burr et al. [16] suggested that using Zeek IDS in conjunction with graphs and ID security analysts can enhance the detection of APT attacks based on data collected over two months. Studies on cyberattack detection technologies using Zeek IDS are ongoing. Overall, research on cyberattack detection using Zeek IDS has predominantly focused on packet-centric analysis, particularly the use of ML or Deep Learning (DL) algorithms to analyze protocols such as Connection and HTTP. However, no documented cases have used the number of cumulative connections in this analysis [11], [12], [13], [14], [15], [16].

Table 3 presents the key issues in related work along with their corresponding solutions.

B. FEATURE ENGINEERING(FE) RELATED RESEARCH

FE is a fundamental task in the preparation of ML data [17]. FE is the process of transforming raw, unprocessed data into a dataset that can be used for ML model development, aiming to improve the model's performance by inputting relevant and appropriate data. Therefore, FE involves constructing suitable features from raw data to enhance predictive performance and generating new features by applying transformation functions, such as arithmetic and aggregation operations. The key FE methods include imputation, feature creation, one-hot encoding, feature transformation, and dimensionality reduction (feature selection and extraction) [18]. To succeed in FE, domain-specific expertise in the subject matter is crucial [19].

1) IMPUTATION [20]

Imputation refers to the handling of missing values in data. One method is to delete a row that contains a missing value, which can affect the data analysis. Therefore, the dataset was supplemented using the following methods.

Categorical Value: Missing categorical variables are generally replaced by the most commonly occurring value in other records.

Numerical Value: Missing numerical values are generally replaced by default values or the mean of the corresponding value in other records.

In addition to the two methods, there is also a method where missing values are inserted randomly.

2) FEATURE CREATION [21]

Feature Creation is the process of creating new features based on observations of data patterns or knowledge. It is a form of Feature Engineering (FE) that can significantly improve the performance of ML models. The methods of Feature Creation include creating new features based on knowledge, such as the nature of the business or standards; creating new features by observing data patterns through aggregate calculations or interactions; and combining existing features or synthesizing new data points to create new features.

3) ONE-HOT ENCODING [22]

To use categorical data in a text format for ML training, it is necessary to convert them into numerical data. However, simply assigning an order can sometimes be misconstrued as representing the magnitude rather than just a sequence. For example, if categories are assigned as SSH(1), HTTP(2), SSL(3), HTTP value might be wrongly perceived as something 'greater' than SSH value. An alternative is to represent each categorical data as a separate column and indicate the presence of a value with 0 and 1. This method is known as One-hot encoding. However, this approach can lead to inefficiency in storage space, as separate columns are created for each data, especially when there are many categories.

4) FEATURE TRANSFORMATION(FT) [23]

Feature Transformation (FT) is the process of transforming features to be represented more suitably for ML models. This was done to enable the model to learn effectively from the data. Types of FT include Normalization, Standardization, and Scaling.

5) DIMENSIONALITY REDUCTION [24]

Dimensionality Reduction is a method of reducing dimensions to enhance ML performance, with Feature Extraction and Selection as the primary techniques.

a: FEATURE EXTRACTION [24]

It refers to reducing the size of features without losing information from the original feature space. Through such Feature Extraction, when the dimensions are reduced, the amount of computation for the computer decreases and memory is conserved, leading to increased efficiency and improved performance of the analysis system.

b: FEATURE SELECTION [24], [25]

Feature selection is the process of selecting relevant attributes that have the most significant impact on detection or prediction from a large amount of data. By eliminating redundant or irrelevant data through Feature Selection, it is possible to improve prediction accuracy without data loss. Feature Selection algorithms for raw data are categorized into similarity-based, information theory-based, sparse learning-based, and statistical methods. Significant research has been conducted on optimal Dimensionality Reduction, with notable studies including mRmR, RELIEF, CMIM, Correlation Coefficient, BW-ratio, INTERACT, GA, SVM-REF, PCA (Principal Component Analysis) [26], Non-Linear

Principal Component Analysis, Independent Component Analysis, and Correlation-based feature selection.

C. SUPERVISED MACHINE LEARNING ALGORITHMS

The learning methods in ML are broadly divided into supervised, unsupervised, and reinforcement learning. In previous studies using Zeek, supervised learning methods were used, and the main algorithms employed were Logistic Regression (LR), Naïve Bayes (NB), K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Decision Tree, the bagging algorithm Random Forest (RF), and the boosting algorithm Gradient Boosting (GB) [7], [11], [14], [15]. This section explores the ML algorithms used in this study.

Logistic Regression (LR) is a machine learning algorithm that predicts the likelihood of an event using a linear combination. It is a 'supervised learning' algorithm that 'predicts' the probability of data belonging to a category as a value between 0 and 1 using regression and classifies it into the category with the higher likelihood based on that probability [27].

Naïve Bayes (NB) is a simple 'probabilistic classifier' based on Bayes' theorem, which assumes that all features are independent. Research has shown that this model is intuitive, easy to build, and known to perform with high accuracy on small datasets [28].

K-Nearest Neighbors (KNN) is an algorithm that classifies and labels data based on the closest similar attributes. In KNN, 'K' represents the number of nearest neighbors, making it a crucial decision factor in this algorithm [27].

Support Vector Machine (SVM) is an ML algorithm used for pattern recognition, classification, and regression analyses. It operates by selecting a hyperplane that maximizes the margin between different classes. The goal is to find the hyperplane that creates the largest gap between the categories to be classified [7], [27].

Decision Tree (DT) is a set of nodes that splits data and returns binary decisions (Yes or No) based on the conditions of the nodes. The input vector is passed to the corresponding sub-nodes depending on the decision. This process continues until a leaf node is reached and a final decision is made [7], [27].

Ensemble Learning is a machine learning technique that combines multiple algorithms to achieve better performance. The essence of ensemble learning is creating a strong classifier by combining several weak classifiers, which ultimately aims to improve the accuracy of the model. Examples of such ensemble learning techniques include Bagging and Boosting. A representative algorithm of Bagging is Random Forest (RF), which combines multiple Decision Trees, and for Boosting, the Gradient Boosting (GB) algorithm is a notable example [29].

Generally, DL models outperform ML models; however, they often consume substantial system resources, making timely responses challenging. Therefore, ML techniques that utilize FE are often more efficient [30]. Therefore, we conducted experiments using supervised ML algorithms after FE.

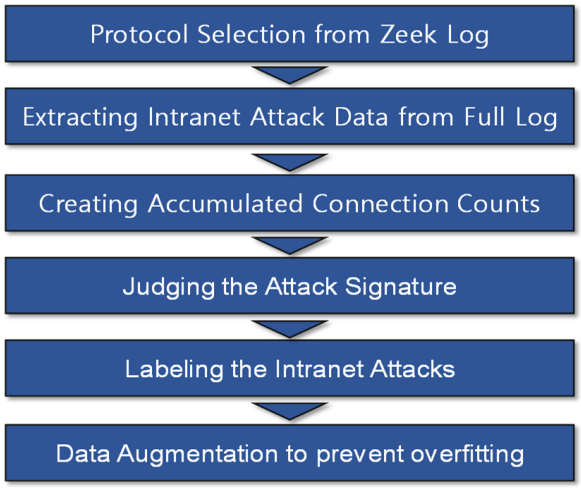

FIGURE 3. Feature engineering procedures.

IV. FEATURE ENGINEERING(FE)

A. GOAL AND PROCEDURE

For successful FE, it is crucial to have expert knowledge of the subject matter, along with an understanding of the raw data being used [19]. Mönch et al. [31] mentioned that not using tools, such as Zeek, was a limitation of his study on real-time APT analysis. The purpose of the FE in this study is to transform Zeek IDS log data into a dataset suitable for ML.

To accomplish this, the FE process was divided into a preprocessing stage and an optimization stage through experimentation. Section IV describes the preprocessing procedures, such as collecting raw data and identifying and labeling attacks, as shown in Figure 3.

B. EXPERIMENTAL ENVIRONMENT

1) OVERVIEW OF THE EXPERIMENTAL ENVIRONMENT

The experimental environment was a simulated test setup that assumed that an organization with both internet and intranet networks would protect its internal resources in a cyberattack scenario. Computers and network equipment that must be defended are infected with known/unknown malware. Attackers continuously execute attacks using these infected computers to penetrate an organization's core system [32]. Defenders do not know the number of infected computers, and use IDS, Firewall, and AV to detect and defend against malicious activities in a given situation.

2) SYSTEM CONFIGURATION

Figure 4 and Table 4 show the system configuration of the experimental environment and setup detail. The institution being defended uses an external internet through a central 'Center' router, and this router also connects the headquarters and branches. Communication between headquarters and branches is conducted using internal IP addresses, whereas computers in Internet access points, such as websites, use public IP addresses. Additionally, both the headquarters and

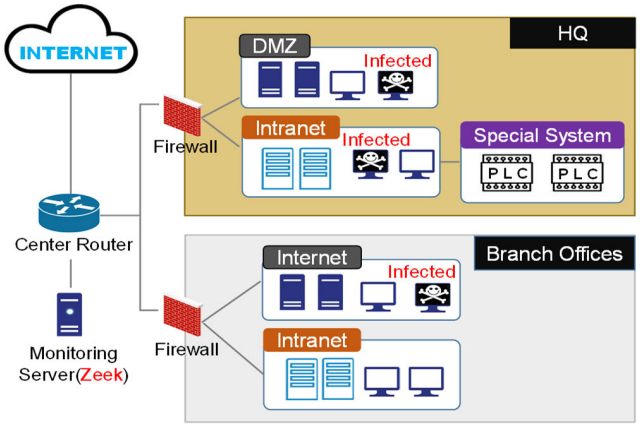

FIGURE 4. System diagram.

TABLE 4. Experimental setup detail.

Classification	Detail
Quantity of equipment	130 devices (PC, Server, PLC)
Operating systems	- Server: Linux (CentOS, Ubuntu), Windows Server - PC: Windows 10, Linux, FreeBSD - Network: VyOS, Fortinet, CISCO - SCADA: Linux, Windows server, Embedded O/S
Application and tools	- Web: Apache, IIS, WAS(JAVA based), Kubernetes - Email: SMTP, POP3, IMAP, Outlook - Monitoring: Arkime, Zeek - SCADA: Custom Solution - PC: Email client, Word processing S/W, Browser
Attack scenario	- Over 2,000 scenarios - Target equipment: All of equipment (Server, PC, PLC, Network devices) - Type of attack payload: Malware infection, Internet web page defacement/down, Intranet SMB Attack, Intranet web page defacement/down, Router intrusion, SCADA system failure, and system destruction attacks
Attack Duration	2 days

branches have their Internet zones, enabling them to access the Internet. These zones also present a risk of communication with external Command and Control (C&C) servers [33].

Therefore, in this network structure, because the internet and intranet computers are configured in a logically segregated network, there is a risk that if internet computers are infected, intranet computers could also be compromised.

3) METHOD OF COLLECTING SYSTEM LOG DATA

Log data are transferred from a computer (server) connected to a mirroring port at the central Center router to Zeek IDS, capturing all packets that pass through. Zeek records packet logs according to predefined rules for each protocol, such as connection, HTTP, FTP, and SSH. These packet logs record both internet and intranet packets.

C. PROTOCOL SELECTION FROM ZEEK LOG

Zeek IDS records 28 types of logs, as listed in Table 1. Among these, protocols that can potentially be used for intranet attacks include SNMP, SOCKS, SSH, RPC, NTLM, SSL, RDP, NTP, Tunnel, SMB, HTTP, FTP, KERBEROS, and RFB (VNC), totaling 14 types. The conn log, which encompasses all others, was not used for the analysis because it would result in duplication when combined with other protocol logs [10]

D. EXTRACTING INTRANET ATTACK DATA FROM FULL LOG

After separating the 16 types of protocol data from the total dataset, the next task is to extract only the intranet packets. This was achieved by analyzing the IP system and network configuration and isolating communications exclusively between intranet IPs (10.x.x.x) [34]. It is important to note that packets are used to monitor the progress of operations, and these packets must also be distinguished and separated.

E. CREATING ACCUMULATED CONNECTION COUNTS

To generate values corresponding to the number of attack attempts, which is a key concept of this study, the processed data thus far are categorized by each IP and arranged in chronological order. Thus, the cumulative number of connections is calculated.

Therefore, to categorize data by IP, the first step is to decide whether to use the Source IP (Src IP) or Destination IP (Dst IP) as the classification criterion. As identifying the attack initiation IP is crucial for distinguishing the act of an attacker initiating an attack, the criterion was set as the Src IP. A simple program was then used to separate the existing data into files based on IP. Next, using the Linux 'sort' command, the files categorized by IP are arranged in chronological order. A line count is added to each sorted row to create a cumulative number of connections.

F. JUDGING THE ATTACK SIGNATURE

Once the task of generating the cumulative number of connections is completed, the next step is to identify whether the packet is an attack. To do this, rows with the 'Attack' string in the Zeek log files were labeled. The attack strings were based on the Attack Signatures used in Mitre, OWASP, Snort rule, and CVE, as shown in TABLE 5 [35], [36], [37], [38].

G. INTRANET-ATTACK LABELING

The next task was to determine whether the packets (rows) identified with Attack Signature were actual intranet attack packets. While there are methods to automate this task [39], to create clear foundational data, every packet marked as 'Yes' by Attack Signature was manually reviewed to determine whether it was an actual attack and then labeled accordingly. Table 6 presents a sample of the dataset with the Attack Labeling process completed.

TABLE 5. References of attack signature.

Protocols	Attack Signature	Annotation
RPC	'\x00\x00\x00\x00', '\xFFSMB'	CVE-2006-5276
KERBEROS	'\x6C\x69\x6C\x00'	Mitre-T1558
HTTP	'CVE', 'EXEC', 'ADMIN', 'MANAGE', 'SETTING', 'UNION', 'JAVASCRIPT', '<SCRIPT>', 'XP_', 'cve', 'exec', 'admin', 'manage', 'setting', 'union', '--', 'javascript', '<script>', 'xp_'	OWASP
NTLM	'1122334455667788','LLMNR'	Mitre-T1557
RDP	'\x03\x00\x00'	MS01-040
RFB(VNC)	'RFB 0'	SNORT
SMB	'IPC$','ipc$','ADMIN$','admin$','C$','D$','PSEXESVC','Shutdown'	CVE-2004-0193
SSH	'GOBBLE'	CVE-2002-0071

TABLE 6. Dataset(sample) after attack labeling.

DateTime	Protocol	SessionID	Src_IP	Src_Port
16503 72205	ssl	CZSjTZwC t6oxi4Czj	10.11.25.201	60044
16503 60770	ntl	C1rsHx4DJ sxn4hJoEf	10.11.8.32	54178
16503 58925	smb	CNy02N2lx OgEYjIGJd	10.11.8.4	56909
16503 54097	htt	CETKO8eCw TX7Jq2E6	10.11.60.119	63604
16503 56512	htt	CYyvyi4yW A3wlXoZPc	10.11.60.120	13398

Dst_IP	Dst_Port	Connect Count	Attack Sign	APT_Label
10.11.59.2	1515	0	0	0
10.11.8.2	139	24	0	0
10.11.8.3	445	31	1	0
100.96.11.8	80	252	1	1
100.96.11.10	80	10	1	0

The statistics of the data after completing the Attack Labeling process are as follows. The total number of entries is 278,948, out of which 2,331 packets have Attack Signatures, and 1,873 are actual attack packets. In other words, when detecting attacks based solely on Attack Signatures, the accurate detection rate was 80%.

H. DATA AUGMENTATION TO PREVENT OVERFITTING

To assess the suitability of the dataset we have worked with so far, we measured the accuracy using Support Vector Machine and Random Forest supervised learning algorithms. Surprisingly, both algorithms exhibit an accuracy of over 99%. However, we quickly realized that this dataset was not suitable for machine learning, which can be understood by examining the data distribution. The total row count of the entire dataset was 278,948 with only 1,873 instances of intranet attacks. In other words, the ratio of labeled data was only 0.67%. Labeled values of less than 1% can lead to model overfitting [40]. Therefore, as shown in Figure 6, strategies to prevent overfitting are necessary for this dataset [41].

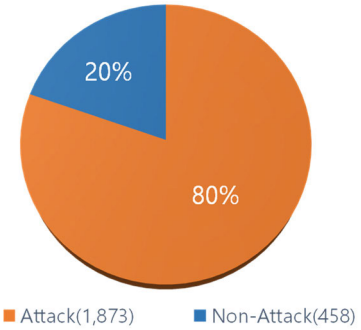

FIGURE 5. The accuracy of only using attack signature.

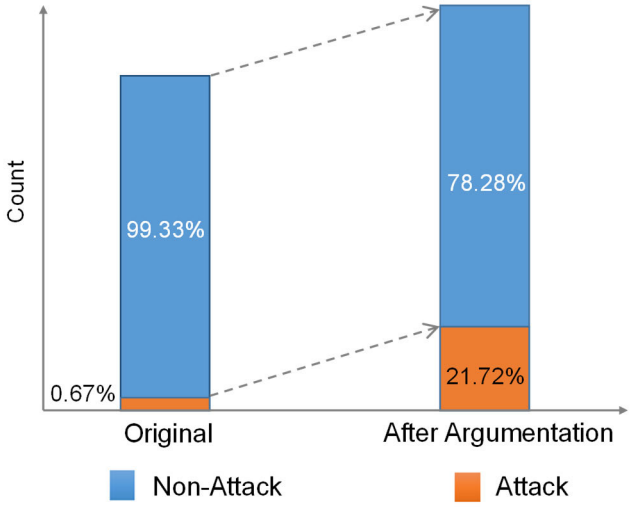

FIGURE 6. Data argumentation.

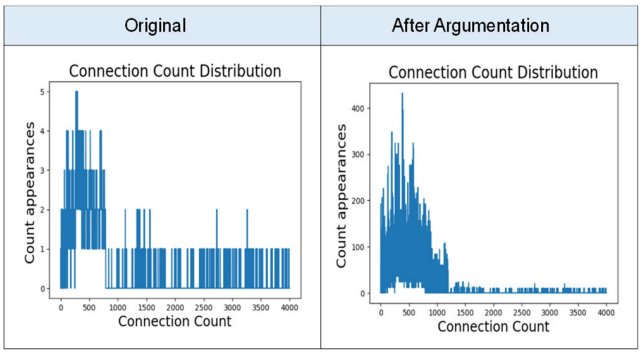

FIGURE 7. Connection Count distribution after argumentation.

TABLE 7. Comparison before and after data argumentation.

Category	Before	After
Total Count	278,948	403,196
Labeled Count	1,873	87,590
Rate (Labeled / Total)	0.67%	21.72%

TABLE 8. Dataset(sample) after encoding.

Protocol	Session ID	Src_IP	Src_Port	Dst_IP	Dst_Port
9	186226	181	31216	142	7
9	186226	181	31216	142	11
9	129703	181	31217	142	11
9	129703	181	31233	142	11

DateTime	Connect Count	Attack Sign	APT_ Label
59783	616	1	1
59743	617	0	0
59744	618	1	0
60744	618	1	0

In this study, to address the overfitting issue, we enhanced the dataset using the Argumentation method, which involves replicating or augmenting labeled data [42]. Initially, for replication, we chose to replicate the labeled HTTP attack packets' port 80 into frequently used ports in web protocols such as 80, 8080, 443, and 8443. For augmentation, we divided the connection count into intervals and amplified the numbers by assigning weights to each interval. Table 7 provides the statistics before and after Data Argumentation, while Figure 7 illustrates the distribution of the connection counts.

As shown in Table 7, through Argumentation, we increased the proportion of labeled data from 1,873 (0.67%) instances to 87,590 (21.72%).

Furthermore, as shown in Figure 7, it is evident that after Argumentation, the distribution of Connection Count frequencies has been amplified in a shape that closely resembles the distribution of the original data.

V. EXPERIMENT AND RESULT

A. GOAL AND PROCEDURE

The goal of this section is to optimize the dataset generated in the previous preprocessing steps to create a robust data model suitable for ML analysis. It is to lay the foundation for assessing the utility of the hypothesis regarding the 'number of cumulative connections.' The procedures used in this section are illustrated in Figure 8. Initially, the dataset was explored through exploratory data analysis (EDA), and the numerical attributes were optimized through experimentation. Subsequently, processes such as feature selection and extraction, which involve dimensionality reduction, are performed to create a more robust final dataset [24], [25]. Strictly speaking, these procedures also fall under the domain of FE, but they include the process of strengthening the model by measuring the performance using ML techniques, which is why they are separated from Chapter IV. Finally, experimental results are presented and their significance is derived.

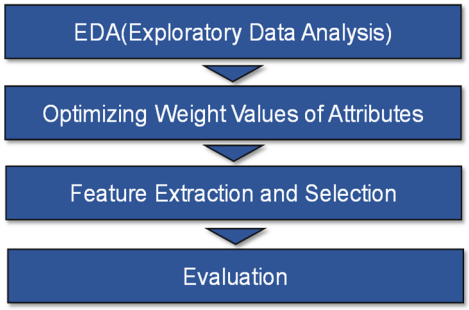

FIGURE 8. Experiment and evaluation procedures.

TABLE 9. Performance measurement results among weight values of connection count attribute.

Weight	LR	NB	SVM	KNN	RF	GB
0.2	0.78	0.78	0.78	0.78	0.78	0.78
1	0.78	0.78	0.78	0.78	0.78	0.78
4	0.78	0.78	0.78	0.78	0.78	0.78

B. EDA (EXPLORATORY DATA ANALYSIS)

To understand the characteristics of the dataset created up to Section IV, we performed Exploratory Data Analysis (EDA) using visualization tools and other techniques [43], [44].

First, for EDA, categorical values must be encoded into numerical values. One-hot encoding is commonly used for encoding; however, in cases such as IP or Session ID, it can lead to too many dimensions. Therefore, Label encoding was used [22]. The dataset created in this manner is listed in Table 8.

Intuitively, SessionID, Src_Port, and DateTime exhibit wide ranges of values. We further explored the characteristics of this dataset using a correlation graph.

Figure 9 shows the correlation (Pairs Plot) between the Protocol, ConnectCount, and AttackSign attributes based on the APT_Label attribute, sampled randomly from 4,000 samples in the dataset [45].

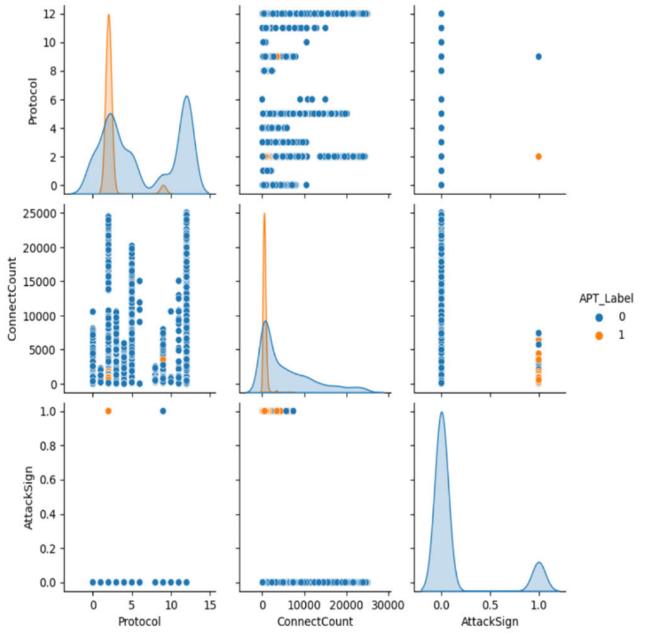

FIGURE 9. Pair plot of dataset.

Examining its significance, we first observed that APT_Label is primarily associated with two types of protocols, whereas ConnectCount mainly falls within the range of less than 1,000 for APT_Label. AttackSign exhibits a strong correlation with APT_Label because it is predominantly found in cases where the result value is True. Based on these exploratory analysis results, we proceeded with the following tasks.

C. OPTIMIZATION WEIGHT VALUES OF ATTRIBUTES

To make the ML model more robust, the next task to be undertaken is the optimization of weight values for numerical attributes, specifically the Connection Count and Attack Signature Attributes. To achieve this, we assigned weights to attributes like the approach used in the WEKA open-source analysis tool [46].

1) CONNECTION COUNT ATTRIBUTE

The method of assigning weight to the ConnectCount attribute involves multiplying the attribute by a weight value. We set the weight range from 0.1 to 4, multiplied the two values, obtained the results through the ML algorithms, and listed these values. From the listed values, we determine the weight value that corresponds to the best performance.

2) ATTACK SIGNATURE

For the Attack Signature Attribute, because its data values are either True or False, we employed a method of adjusting the weight by controlling the proportion occupied by the Attack Signature in the entire dataset, rather than using simple multiplication. We set the weight range between 0.1 and 2,

TABLE 10. Performance measurement results among weight values of attack signature attribute.

Weight	LR	NB	SVM	KNN	RF	GB
0.2	0.94	0.94	0.94	0.94	0.94	0.94
0.5	0.87	0.87	0.87	0.87	0.87	0.87
0.8	**0.81**	**0.81**	**0.81**	**0.81**	**0.81**	**0.81**
1	0.78	0.78	0.78	0.78	0.78	0.78
1.5	0.59	0.59	0.59	0.56	0.59	0.59
2	0.58	0.58	0.58	0.52	0.58	0.58

TABLE 11. Evaluation values by changes in the number of attributes.

Category	7 Attr.	6 Attr.	5 Attr.	4 Attr.	3 Attr.
Accuracy	**0.82**	**0.84**	**0.84**	**0.96**	**0.96**
Precision(M)	0.74	0.85	0.85	0.92	0.92
Precision(W)	0.79	0.84	0.84	0.96	0.96
Recall(M)	0.50	0.56	0.57	0.95	0.94
Recall(W)	0.82	0.84	0.85	0.96	0.96
F1-Score(M)	0.45	0.56	0.60	0.93	0.93
F1-Score(W)	0.73	0.78	0.80	0.96	0.96

and for weight values < 1, we randomly deleted True Attack Signature values from the data using the weight ratio. Conversely, for weight values greater than 1, we randomly deleted data with Attack Signature values as False using the weight ratio. We then calculated the results using ML algorithms and listed the values. Subsequently, we determined the weight value corresponding to the best performance from the listed values.

The ML algorithms used to optimize the two attribute weights were six supervised learning algorithms with confirmed validation in previous research: Logistic Regression (LR), Naïve Bayes (NB), K-Nearest-Neighbors (KNN), Support Vector Machine (SVM), Random Forest (RF), and Gradient Boosting (GB). To ensure the robustness of the model, we set the K-fold cross-validation value to five [47]. The hyperparameter values used for each ML algorithm are listed in APPENDIX A [48].

Table 9 lists the accuracy results of the six algorithms for the Connection Count Attribute. The comprehensive performance evaluation results, including Recall, and F-Score, are presented in APPENDIX B [49].

Regardless of the weight, the results remained constant at 0.78. Consequently, it can be confirmed that the weight value

TABLE 12. Evaluation results.

Category	LR	NB	SVM	KNN	RF	GB
Accuracy	**0.81**	**0.81**	**0.96**	**0.81**	**0.81**	**0.81**
Precision(M)	0.41	0.41	0.92	0.50	0.41	0.41
Precision(W)	0.66	0.66	0.96	0.70	0.66	0.66
Recall(M)	0.50	0.50	0.95	0.50	0.50	0.50
Recall(W)	0.81	0.81	0.96	0.81	0.81	0.81
F1-Score(M)	0.45	0.45	0.93	0.45	0.45	0.45
F1-Score(W)	0.73	0.73	0.96	0.73	0.73	0.73

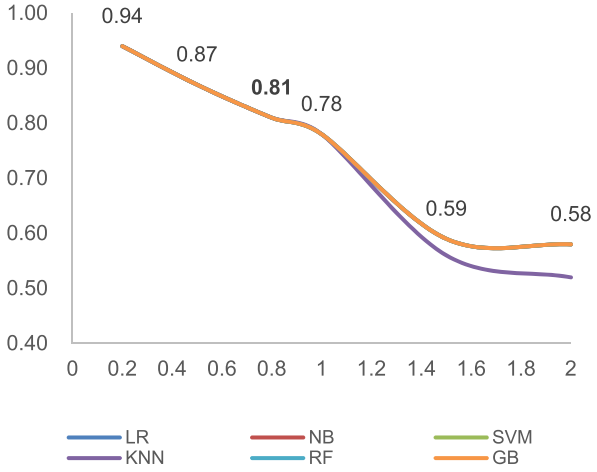

FIGURE 10. A graph of performance measurement results.

of the Connection Count attribute does not significantly affect the algorithms. Therefore, a default weight value of 1 was used.

Table 10 presents the accuracy results for the six algorithms regarding the Attack Signature Attribute, and Figure 10 presents a graph of the results. The comprehensive performance evaluation results are presented in APPENDIX B.

The Attack Signature weight value has a consistent influence on the results as it increases or decreases the data volume. When the weight value was small, the data tended to overfit, resulting in high accuracy, whereas when the weight value was large, the data tended to underfit, leading to lower accuracy. Therefore, considering the trade-off between overfitting and underfitting, a weight value of 0.8 is chosen [50].

D. FEATURE EXTRACTION AND SELECTION

In this section, we aim to improve the model's performance by undergoing the Feature Extraction process to reduce dimensionality without information loss and conducting Feature Selection to choose attributes that positively impact performance [24], [25]. Thus far, the summarized attributes, excluding the Attack Label, include Protocol, SessionID, Src_IP, Src_Port, Dst_IP, Dst_Port, DateTime, ConnectCount, and AttackSign, totaling nine attributes. We must

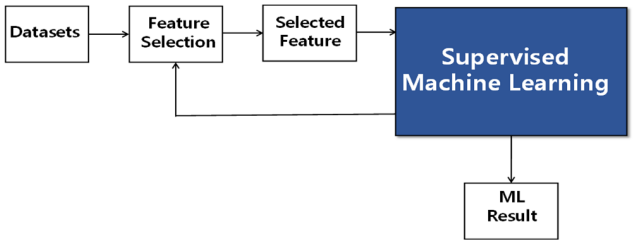

FIGURE 11. Feature selection process.

select attributes that can potentially influence attack detection or prediction and create a formula. In this study, the attributes used to reduce the dimensionality were Protocol, ConnectCount, and AttackSign. Using these attributes, we derive a formula, calculate the resulting values, and generate a new attribute called PCA(1) to reduce the dimensions [24], [51]. The computation method for the PCA attribute is as follows:

Definition

1. $P_r = \{3, 2, 1\}$ represents the ranking values of the Attack Signature's proportion among the protocols.
2. $C_q = \{\text{Number}\}$ represents the cumulative connection count.
3. $A_b = \{1, 0\}$ represents the Attack Signature.
4. $W_c = 1.0$ is the weight for the Connection Count attribute.
5. $W_a = 0.8$ is the weight for the Attack Signature attribute.
6. $N_{[cr]}$ denotes the N-th current row.

Formula

$$PCA_{[cr]} = \frac{P_r \times (C_q \cdot W_c) \times (A_b \cdot W_a)}{3}_{[cr]} \quad (1)$$

By performing Feature Extraction, nine attributes–SessionID, Src_IP, Src_Port, Dst_IP, Dst_Port, DateTime, and PCA–were reduced to seven attributes.

Now, the next step was to select the features that could improve the performance of the model. The feature Selection was performed as shown in Figure 11 [25].

Supervised ML algorithms used for Feature Selection included the same six algorithms as before. The performance comparison results showed significant performance support for the SVM algorithm, which aligned with the intentions of the experiment. For the other algorithms, all the values were measured similarly. Therefore, the Feature Selection process proceeded by considering the performance evaluation results of the SVM algorithm. As the first step, Src_Port, SessionID, and DateTime attributes, where values are randomly generated, were eliminated to compare the performance [52]. Then, SessionID and DateTime are eliminated sequentially, and the performance is measured accordingly [53]. The results are presented in Table 11 and Figure 12.

As shown in Table 11 and Figure 12, when gradually reducing features with a wide range of values, the performance improves slightly with each reduction until it significantly improves when there are four features. Therefore, as a result

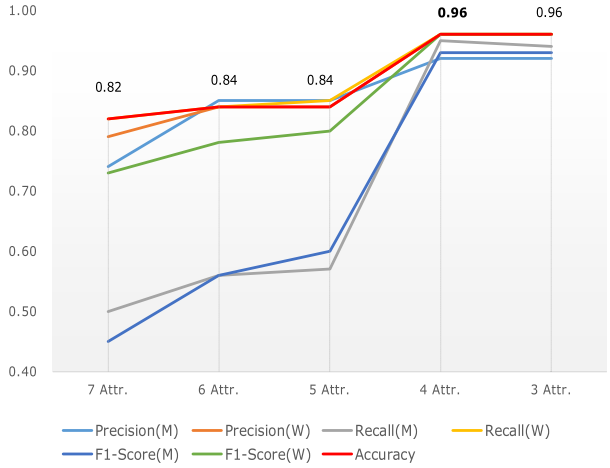

FIGURE 12. Evaluation values by changes in the number of attributes.

TABLE 13. Comparison between PCA and PA in SVM algorithm.

Category	PCA	PA	Difference
Accuracy	**0.96**	**0.92**	**0.04**
Precision(M)	0.92	0.89	0.03
Precision(W)	0.96	0.92	0.04
Recall(M)	0.95	0.83	0.12
Recall(W)	0.96	0.92	0.04
F1-Score(M)	0.93	0.86	0.07
F1-Score(W)	0.96	0.92	0.04

of Feature Extraction and Selection, it can be confirmed that the best performance was achieved with only four features: Src_IP, Dst_IP, Dst_Port, and PCA.

E. EXPERIMENTAL RESULTS

We evaluated the performance using a dataset generated through the FE process with six supervised learning algorithms. Table 12 presents the results of the performance evaluation [49].

The FE method employed in this paper to derive the dataset and evaluate its performance with supervised ML algorithms revealed excellent performance with an accuracy of 96% in the SVM algorithm.

VI. DISCUSSION AND LIMITATION

A. PERFORMANCE COMPARISON WHEN THERE IS NO CONNECTION COUNT

To assess the effectiveness of this study, we discussed and experimented with the performance evaluation results when the Connection Count was included as a detection attribute and when it was not.

In this study, during the FE process, three attributes - Protocol, ConnectCount, and AttackSign–were computed to derive a value called the PCA to reduce the dimensions. Therefore, to compare the performance of ConnectCount presence or absence, we need to compute only the Protocol and AttackSign values to create an attribute called PA, and then compare the performance evaluation values when PCA is selected and when PA is selected. The formula for generating the PA(2) value is as follows:

Formula

$$PA_{[cr]} = \frac{P_r \times (A_b \cdot W_a)}{2}_{[cr]} \tag{2}$$

Table 13 presents a performance evaluation comparison between PCA and PA when the SVM algorithm was selected.

As observed in Table 13, it can be noted that when Connect Count is not included as a detection attribute, there is a difference of 0.4 in accuracy.

B. ADDRESSING POTENTIAL VULNERABILITIES

Addressing potential vulnerabilities is the ultimate goal for those researching cybersecurity. Cybersecurity is not achieved with a single effort but rather through multiple layers of shields, striving for near-perfect security. Various organizations and entities are creating frameworks to prevent potential vulnerabilities, with one notable example being MITRE D3FEND, operated with funding from the United States National Security Agency (NSA) [57]. MITRE manages the ATT&CK attack model and the D3FEND defense model, categorizing defense methods into six classifications: Harden, Detect, Isolate, Deceive, Evict, and Restore, and providing detailed measures for each. While the D3FEND model cannot prevent all cyberattacks, it is considered effective in addressing potential vulnerabilities. In particular, under the ''Detect'' category, there is a sub-category called ''User Behavior Analysis,'' which includes Network Traffic Analysis and Resource Access Analysis. Our research falls within this domain, and the payloads in ''Table 5'' of this document can be considered as examples of behavior-based defense technologies. While the security methodologies we propose may not address all potential vulnerabilities, we anticipate that the accumulation of such small efforts by researchers will contribute to the advancement of cybersecurity.

C. ENRICHING QUANTITATIVE ANALYSIS AND REAL-TIME DETECTION EFFICIENCY

Advancing this experiment into a real-time detection system is both practical and our ultimate goal. However, applying a theory to real-world situations requires several steps. The first step involves establishing the initial concept and proving its validity. The second step involves determining whether to proceed to the practical stage through comprehensive quantitative analysis. The third step entails designing and developing a system capable of real-time operation based on the established concept. This third step also requires numerous trial and error attempts. Finally, applying the system to real-world scenarios, comparing the real-time detection efficiency, and optimizing the process are necessary. The current manuscript represents an initial stage document that utilizes ML to verify the significance of 'Accumulated Connection Counts.' Many steps remain ahead to reach the stage where real-time detection efficiency can be measured. This

TABLE 14. Hyper parameter values.

ML Algo.	Hyper Parameter Values
Logistic Regression	'C': 1.0, 'class_weight': None, 'dual': False, 'fit_intercept': True, 'intercept_scaling': 1, 'l1_ratio': None, 'max_iter': 1000, 'multi_class': 'auto', 'n_jobs': None, 'penalty': 'l2', 'random_state': None, 'solver': 'lbfgs', 'tol': 0.0001, 'verbose': 0, 'warm_start': False
Naïve Bayes	'priors': None, 'var_smoothing': 1e-09
K-Nearest-Neighbors	'algorithm': 'auto', 'leaf_size': 30, 'metric': 'minkowski', 'metric_params': None, 'n_jobs': -1, 'n_neighbors': 10, 'p': 2, 'weights': 'uniform'
Support Vector Machine	'C': 1.0, 'break_ties': False, 'cache_size': 200, 'class_weight': None, 'coef0': 0.0, 'decision_function_shape': 'ovr', 'degree': 3, 'gamma': 'scale', 'kernel': 'rbf', 'max_iter': -1, 'probability': False, 'random_state': None, 'shrinking': True, 'tol': 0.001, 'verbose': False
Random Forest	'bootstrap': True, 'ccp_alpha': 0.0, 'class_weight': None, 'criterion': 'gini', 'max_depth': 5, 'max_features': 'sqrt', 'max_leaf_nodes': None, 'max_samples': None, 'min_impurity_decrease': 0.0, 'min_samples_leaf': 8, 'min_samples_split': 8, 'min_weight_fraction_leaf': 0.0, 'n_estimators': 10, 'n_jobs': -1, 'oob_score': False, 'random_state': 23, 'verbose': 0, 'warm_start': False
Gradient Boosting	'ccp_alpha': 0.0, 'criterion': 'friedman_mse', 'init': None, 'learning_rate': 0.2, 'loss': 'log_loss', 'max_depth': 5, 'max_features': None, 'max_leaf_nodes': None, 'min_impurity_decrease': 0.0, 'min_samples_leaf': 10, 'min_samples_split': 10, 'min_weight_fraction_leaf': 0.0, 'n_estimators': 30, 'n_iter_no_change': None, 'random_state': 23, 'subsample': 0.7, 'tol': 0.0001, 'validation_fraction': 0.1, 'verbose': 0, 'warm_start': False

TABLE 15. 'Connection count' attribute.

Weight	Category	LR	NB	SVM	KNN	RF	GB
0.2	Accuracy	**0.78**	**0.78**	**0.78**	**0.78**	**0.78**	**0.78**
	Precision(M)	0.39	0.39	0.39	0.50	0.39	0.39
	Precision(W)	0.61	0.61	0.61	0.66	0.61	0.61
	Recall(M)	0.50	0.50	0.50	0.50	0.50	0.50
	Recall(W)	0.78	0.78	0.78	0.78	0.78	0.78
	F1-Score(M)	0.44	0.44	0.44	0.45	0.44	0.44
	F1-Score(W)	0.69	0.69	0.69	0.69	0.69	0.69
0.5	Accuracy	**0.78**	**0.78**	**0.78**	**0.78**	**0.78**	**0.78**
	Precision(M)	0.39	0.39	0.39	0.51	0.39	0.39
	Precision(W)	0.61	0.61	0.61	0.66	0.61	0.61
	Recall(M)	0.50	0.50	0.50	0.50	0.50	0.50
	Recall(W)	0.78	0.78	0.78	0.78	0.78	0.78
	F1-Score(M)	0.44	0.44	0.44	0.45	0.44	0.44
	F1-Score(W)	0.69	0.69	0.69	0.69	0.69	0.69
1	Accuracy	**0.78**	**0.78**	**0.78**	**0.78**	**0.78**	**0.78**
	Precision(M)	0.39	0.39	0.39	0.51	0.39	0.39
	Precision(W)	0.61	0.61	0.61	0.66	0.61	0.61
	Recall(M)	0.50	0.50	0.50	0.50	0.50	0.50
	Recall(W)	0.78	0.78	0.78	0.78	0.78	0.78
	F1-Score(M)	0.44	0.44	0.44	0.45	0.44	0.44
	F1-Score(W)	0.69	0.69	0.69	0.69	0.69	0.69
2	Accuracy	**0.78**	**0.78**	**0.78**	**0.78**	**0.78**	**0.78**
	Precision(M)	0.39	0.39	0.39	0.50	0.39	0.39
	Precision(W)	0.61	0.61	0.61	0.66	0.61	0.61
	Recall(M)	0.50	0.50	0.50	0.50	0.50	0.50
	Recall(W)	0.78	0.78	0.78	0.78	0.78	0.78
	F1-Score(M)	0.44	0.44	0.44	0.45	0.44	0.44
	F1-Score(W)	0.69	0.69	0.68	0.69	0.69	0.69
4	Accuracy	**0.78**	**0.78**	**0.78**	**0.78**	**0.78**	**0.78**
	Precision(M)	0.39	0.39	0.39	0.50	0.39	0.39
	Precision(W)	0.61	0.61	0.61	0.66	0.61	0.61
	Recall(M)	0.50	0.50	0.50	0.50	0.50	0.50
	Recall(W)	0.78	0.78	0.78	0.78	0.78	0.78
	F1-Score(M)	0.44	0.44	0.44	0.45	0.44	0.44
	F1-Score(W)	0.69	0.69	0.69	0.69	0.69	0.69

research is now in the first stage of establishing and validating initial concepts. Therefore, we plan to establish plans for the procedures corresponding to stages 2, 3, and 4 for future progression.

On the other hand, there have been various opinions on whether our experiments are meaningful only in real-time scenarios. In the case of intranet situations, attacks are usually sophisticated and long-lasting APT attacks. In considering such situations, there is an opinion that analyzing collected logs to detect attacks, rather than real-time detection, can also help companies or organizations eliminate threats they face.

TABLE 16. 'Attack signature' attribute.

Weight	Category	LR	NB	SVM	KNN	RF	GB
	Accuracy	**0.94**	**0.94**	**0.94**	**0.94**	**0.94**	**0.94**
	Precision(M)	0.47	0.47	0.47	0.47	0.47	0.47
	Precision(W)	0.89	0.89	0.89	0.89	0.89	0.89
0.2	Recall(M)	0.50	0.50	0.50	0.50	0.50	0.50
	Recall(W)	0.94	0.94	0.94	0.94	0.94	0.94
	F1-Score(M)	0.48	0.48	0.48	0.48	0.48	0.48
	F1-Score(W)	0.91	0.91	0.92	0.92	0.92	0.92
	Accuracy	**0.87**	**0.87**	**0.87**	**0.87**	**0.87**	**0.87**
	Precision(M)	0.44	0.44	0.44	0.47	0.44	0.44
	Precision(W)	0.76	0.76	0.76	0.77	0.76	0.76
0.5	Recall(M)	0.50	0.50	0.50	0.50	0.50	0.50
	Recall(W)	0.87	0.87	0.87	0.87	0.87	0.87
	F1-Score(M)	0.47	0.47	0.47	0.47	0.47	0.47
	F1-Score(W)	0.81	0.81	0.81	0.81	0.81	0.81
	Accuracy	**0.81**	**0.81**	**0.81**	**0.81**	**0.81**	**0.81**
	Precision(M)	0.41	0.41	0.41	0.51	0.41	0.41
	Precision(W)	0.66	0.66	0.66	0.70	0.66	0.66
0.8	Recall(M)	0.50	0.50	0.50	0.50	0.50	0.50
	Recall(W)	0.81	0.81	0.81	0.81	0.81	0.81
	F1-Score(M)	0.45	0.45	0.45	0.45	0.45	0.45
	F1-Score(W)	0.73	0.73	0.73	0.73	0.73	0.73
	Accuracy	**0.78**	**0.78**	**0.78**	**0.78**	**0.78**	**0.78**
	Precision(M)	0.39	0.39	0.39	0.50	0.39	0.39
	Precision(W)	0.61	0.61	0.61	0.66	0.61	0.61
1	Recall(M)	0.50	0.50	0.50	0.50	0.50	0.50
	Recall(W)	0.78	0.78	0.78	0.78	0.78	0.78
	F1-Score(M)	0.44	0.44	0.44	0.45	0.44	0.44
	F1-Score(W)	0.69	0.69	0.69	0.69	0.69	0.69
	Accuracy	**0.59**	**0.59**	**0.59**	**0.56**	**0.59**	**0.59**
	Precision(M)	0.30	0.30	0.30	0.50	0.30	0.51
	Precision(W)	0.35	0.35	0.35	0.52	0.35	0.52
1.5	Recall(M)	0.50	0.50	0.50	0.50	0.50	0.50
	Recall(W)	0.59	0.59	0.59	0.56	0.59	0.59
	F1-Score(M)	0.37	0.37	0.37	0.47	0.37	0.37
	F1-Score(W)	0.44	0.44	0.44	0.51	0.44	0.44
	Accuracy	**0.58**	**0.58**	**0.58**	**0.52**	**0.58**	**0.58**
	Precision(M)	0.29	0.29	0.29	0.50	0.47	0.51
	Precision(W)	0.34	0.34	0.34	0.51	0.49	0.52
2	Recall(M)	0.50	0.50	0.50	0.50	0.50	0.50
	Recall(W)	0.58	0.58	0.58	0.52	0.58	0.58
	F1-Score(M)	0.37	0.37	0.37	0.50	0.37	0.38
	F1-Score(W)	0.43	0.43	0.43	0.51	0.43	0.44

In other words, although we analyzed IDS logs to determine the significance of the intranet 'Accumulated Connection Counts' attribute for detection, some argue that detection does not necessarily have to be in real-time. Even if it does not achieve the efficiency of real-time detection, this detection theory still becomes effective in its own way.

D. LIMITATIONS AND FUTURE PLANS

This study focused on analyzing packets in the forward direction, where attacks originate from clients and target servers within an intranet. However, these attacks were not limited to this direction. For instance, in cases where a computer is compromised by malware because of an attack, it may send signals in the reverse direction [54]. In the future, we plan to enhance this study by including reverse-direction attacks and providing a more comprehensive analysis of intranet attacks.

VII. CONCLUSION

When changes occur in the cybersecurity landscape, such as network segregation and the emergence of Zero Trust, a reevaluation of attack detection methods is essential. This paper proposes the selection of connection count as an attribute for behavior-based intranet attack detection, taking into consideration the persistent and repetitive attack patterns of adversaries. To support this proposal, intranet attack packets were analyzed using machine learning algorithms, and the basis for this approach was provided. Specifically, an intranet attack environment was constructed and logs collected from Zeek IDS were transformed into analyzable datasets through a feature engineering process. In this process, techniques like Argumentation for preventing overfitting, attribute optimization, and feature extraction/selection were employed to derive a dataset with four attributes: Src_IP, Dst_IP, Dst_Port, and PCA. Subsequently, the performance of six supervised learning algorithms was evaluated on this dataset, and the SVM algorithm achieved a high accuracy of 97%. It was also observed that the presence of connection count values outperformed scenarios where they were absent.

In the future, we plan to enhance this study to encompass reverse-direction intranet attack forms. We hope that this paper contributes to the continued development of the cybersecurity and computer science fields through the dataset and experimental methods it provides.

APPENDIX A

See Table 14.

APPENDIX B
PERFORMANCE MEASUREMENT RESULTS OF WEIGHT VALUES

See Tables 15 and 16.

APPENDIX C
DATASET DETAIL

- File Type: Normal CSV file
- Total number of lines: 403,196
- Structure: All fields are numeric values

Attribute	Type	Detail
Protocol	Number	**Protocols**
SessionID	Number	Session ID values
Src_IP	Number	Source IP address
Src_Port	Number	Source Port number
Dst_IP	Number	Destination IP address
DateTime	Number	Date and Time
ConnectCount	Number	The number of connection attempts
AttackSign	Number	Attack Signature
ProtocolR	Number	Protocols Ranking
PCA	Number	Calculation results of ProtocolR, ConnectCount, and AttackSign fields
PA	Number	Calculation results of ProtocolR and AttackSign fields
APT_Label	Number	Intranet Attack Labeling

REFERENCES

[1] J. Liu, Z. Wang, J. Yang, B. Wang, L. He, G. Song, and X. Liu, "Deception maze: A Stackelberg game-theoretic defense mechanism for intranet threats," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2021, pp. 1–6.

[2] X. Wang, X. Gong, L. Yu, and J. Liu, "MAAC: Novel alert correlation method to detect multi-step attack," in *Proc. IEEE 20th Int. Conf. Trust, Secur. Privacy Comput. Commun. (TrustCom)*, Oct. 2021, pp. 726–733.

[3] Z. Liu, C. Wang, and S. Chen, "Correlating multi-step attack and constructing attack scenarios based on attack pattern modeling," in *Proc. Int. Conf. Inf. Secur. Assurance (ISA)*, Apr. 2008.

[4] V. A. Stafford, *Zero Trust Architecture*, document NIST Special Publication 800, 2020.

[5] P. Chen, L. Desmet, and C. Huygens, "A study on advanced persistent threats," in *Proc. 15th IFIP TC 6/TC 11 Int. Conf. Commun. Multimedia Security (CMS)*, Aveiro, Portugal. Berlin, Germany: Springer, Sep. 2014.

[6] B.-J. Cho, J.-H. Yun, and K.-H. Lee, "Study of effectiveness for the network separation policy of financial companies," *J. Korea Inst. Inf. Secur. Cryptol.*, vol. 25, no. 1, pp. 181–195, Feb. 2015.

[7] S. Bagui, D. Mink, S. Bagui, T. Ghosh, T. McElroy, E. Paredes, N. Khasnavis, and R. Plenkers, "Detecting reconnaissance and discovery tactics from the MITRE ATT&CK framework in Zeek Conn Logs using Spark's machine learning in the big data framework," *Sensors*, vol. 22, no. 20, p. 7999, Oct. 2022.

[8] M. M. Anjum, S. Iqbal, and B. Hamelin, "Analyzing the usefulness of the DARPA OpTC dataset in cyber threat detection research," in *Proc. 26th ACM Symp. Access Control Models Technol.*, Jun. 2021.

[9] *API Reference—Scikit-Learn 1.3.2 Documentation*. Accessed: Dec. 1, 2023. [Online]. Available: https://scikit-learn.org/stable/modules/classes.html

[10] *Log Files—Book of Zeek*. Accessed: Dec. 1, 2023. [Online]. Available: https://docs.zeek.org/en/master/script-reference/log-files.html

[11] V. Gustavsson, *Machine Learning for a Network-Based Intrusion Detection System*, document Grundny´vå, 15 Hp, Elektronýk Och Datorteknýk, Stockholm, Sverige, 2019.

[12] J. Svoboda, Ghafir, and V. Prenosil, "Network monitoring approaches: An overview," *Int. J. Adv. Comput. Netw. Secur.*, vol. 5, no. 2, pp. 88–93, 2015.

[13] M. Rodríguez, Á. Alesanco, L. Mehavilla, and J. García, "Evaluation of machine learning techniques for traffic flow-based intrusion detection," *Sensors*, vol. 22, no. 23, p. 9326, Nov. 2022.

[14] D. K. Andrews, R. K. Agrawal, S. J. Matthews, and A. S. Mentis, "Comparing machine learning techniques for Zeek log analysis," in *Proc. IEEE MIT Undergraduate Res. Technol. Conf. (URTC)*, Oct. 2019, pp. 1–4.

[15] A. Jimenez, "Using machine learning for raspberry pi network intrusion detection," Ph.D. dissertation, California State Polytech. Univ., Pomona, CA, USA, 2021.

[16] B. Burr, S. Wang, G. Salmon, and H. Soliman, "On the detection of persistent attacks using alert graphs and event feature embeddings," in *Proc. NOMS - IEEE/IFIP Netw. Oper. Manage. Symp.*, Apr. 2020, pp. 1–4.

[17] G. Dong and H. Liu, *Feature Engineering for Machine Learning and Data Analytics*. Boca Raton, FL, USA: CRC Press, 2018.

[18] A. Datta, K. Sangaralingam, S. Chen, D. Tran, S. Sen, and R. Ranjan, "Anovos: A scalable feature engineering library," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2022, pp. 2637–2644.

[19] C. C, P. K. Pareek, V. H. C. de Albuquerque, A. Khanna, and D. Gupta, "Improved domain generation algorithm to detect cyber-attack with deep learning techniques," in *Proc. IEEE 2nd Mysore Sub Sect. Int. Conf. (MysuruCon)*, Oct. 2022, pp. 1–8.

[20] F. Biessmann, T. Rukat, P. Schmidt, P. Naidu, S. Schelter, A. Taptunov, D. Lange, and D. Salinas, "DataWig: Missing value imputation for tables," *J. Mach. Learn. Res.*, vol. 20, no. 175, pp. 1–6, 2019.

[21] Y. Diao, L. Yan, and K. Gao, "Improvement of the machine learning-based corrosion rate prediction model through the optimization of input features," *Mater. Design*, vol. 198, Jan. 2021, Art. no. 109326.

[22] S. Okada, M. Ohzeki, and S. Taguchi, "Efficient partition of integer optimization problems with one-hot encoding," *Sci. Rep.*, vol. 9, no. 1, p. 13036, Sep. 2019.

[23] H. A. H. Al-Najjar, B. Pradhan, B. Kalantar, M. I. Sameen, M. Santosh, and A. Alamri, "Landslide susceptibility modeling: An integrated novel method based on machine learning feature transformation," *Remote Sens.*, vol. 13, no. 16, p. 3281, Aug. 2021.

[24] S. Khalid, T. Khalil, and S. Nasreen, "A survey of feature selection and feature extraction techniques in machine learning," in *Proc. Sci. Inf. Conf.*, Aug. 2014, pp. 372–378.

[25] H. Lee, D. Choi, H. Yim, E. Choi, W. Go, T. Lee, I. Kim, and K. Lee, "Feature selection practice for unsupervised learning of credit card fraud detection," *J. Theor. Appl. Inf. Technol.*, vol. 96, no. 2, pp. 408–417, 2018.

[26] C. O. S. Sorzano, J. Vargas, and A. P. Montano, "A survey of dimensionality reduction techniques," 2014, *arXiv:1403.2877*.

[27] S. Dreiseitl and L. Ohno-Machado, "Logistic regression and artificial neural network classification models: A methodology review," *J. Biomed. Informat.*, vol. 35, nos. 5–6, pp. 352–359, Oct. 2002.

[28] S. Xu, "Bayesian Naïve Bayes classifiers to text classification," *J. Inf. Sci.*, vol. 44, no. 1, pp. 48–59, 2018.

[29] G.-W. Cha, H.-J. Moon, and Y.-C. Kim, "Comparison of random forest and gradient boosting machine models for predicting demolition waste based on small datasets and categorical variables," *Int. J. Environ. Res. Public Health*, vol. 18, no. 16, p. 8530, Aug. 2021.

[30] B. Tang, J. Yang, X. Li, Y. Cao, and J. Wang, "APT detector: Detect and identify APT malware based on deep learning framework," in *Proc. 9th Int. Conf. Comput. Artif. Intell.*, Mar. 2023.

[31] S. Mönch and H. Roth, "Real-time APT detection technologies: A literature review," in *Proc. IEEE Int. Conf. Cyber Secur. Resilience (CSR)*, Jul. 2023.

[32] B. Stojanović, K. Hofer-Schmitz, and U. Kleb, "APT datasets and attack modeling for automated detection methods: A review," *Comput. Secur.*, vol. 92, May 2020, Art. no. 101734.

[33] W. Sun, V. Katta, K. Krishna, and R. Sekar, "V-NetLab: An approach for realizing logically isolated networks for security experiments," in *Proc. CSET*, vol. 8, 2008, pp. 1–6.

[34] S. N. Sisat, P. S. Bhopale, and V. K. Barbudhe, "IP subnetting," *Int. J. Electron., Commun. Soft Comput. Sci. Eng.*, vol. 2, no. 5, pp. 1–5, 2012.

[35] *NVD—CVE-2006-5276*. Accessed: Dec. 1, 2023. [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2006-5276

[36] *Snort Rules and IDS Software Download*. Accessed: Dec. 1, 2023. [Online]. Available: https://www.snort.org/downloads

[37] *Steal or Forge Kerberos Tickets, Technique T1558—Enterprise, MITRE ATT&CK*. Accessed: Dec. 1, 2023. [Online]. Available: https://attack.mitre.org/techniques/T1558

[38] *OWASP Foundation, the Open Source Foundation for Application Security*. Accessed: Dec. 1, 2023. [Online]. Available: https://owasp.org
[39] D. A. Cieslak, N. V. Chawla, and A. Striegel, "Combating imbalance in network intrusion datasets," in *Proc. GrC*, May 2006, pp. 732–737.
[40] D. M. Hawkins, "The problem of overfitting," *J. Chem. Inf. Comput. Sci.*, vol. 44, no. 1, pp. 1–12, Jan. 2004.
[41] X. Ying, "An overview of overfitting and its solutions," *J. Phys., Conf. Ser.*, vol. 1168, Feb. 2019, Art. no. 022022.
[42] C. Shorten, T. M. Khoshgoftaar, and B. Furht, "Text data augmentation for deep learning," *J. Big Data*, vol. 8, no. 1, pp. 1–34, Dec. 2021.
[43] S. Morgenthaler, "Exploratory data analysis," *Wiley Interdiscipl. Rev., Comput. Statist.*, vol. 1, no. 1, pp. 33–44, 2009.
[44] Z. Xie, J. Pan, C.-C. Chang, J. Hu, and Y. Chen, "The dark side: Security and reliability concerns in machine learning for EDA," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 42, no. 4, pp. 1171–1184, Apr. 2023.
[45] J. W. Emerson, W. A. Green, B. Schloerke, J. Crowley, D. Cook, H. Hofmann, and H. Wickham, "The generalized pairs plot," *J. Comput. Graph. Statist.*, vol. 22, no. 1, pp. 79–91, Jan. 2013.
[46] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: An update," *ACM SIGKDD Explor. Newsletter*, vol. 11, no. 1, pp. 10–18, 2009.
[47] C. Bergmeir and J. M. Benítez, "On the use of cross-validation for time series predictor evaluation," *Inf. Sci.*, vol. 191, pp. 192–213, May 2012.
[48] L. Yang and A. Shami, "On hyperparameter optimization of machine learning algorithms: Theory and practice," *Neurocomputing*, vol. 415, pp. 295–316, Nov. 2020.
[49] D. M. W. Powers, "Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation," 2020, *arXiv:2010.16061*.
[50] H. Jabbar and R. Z. Khan, "Methods to avoid over-fitting and under-fitting in supervised machine learning (comparative study)," *Comput. Sci., Commun. Instrum. Devices*, vol. 70, no. 10, pp. 978–981, 2015.
[51] N. Elavarasan and K. Mani, "A survey on feature extraction techniques," *Int. J. Innov. Res. Comput. Commun. Eng.*, vol. 3, no. 1, pp. 52–55, 2015.
[52] *Recommendations for Transport-Protocol Port Randomization*, document RFC 6056, 6056. Accessed: Dec. 1, 2023. [Online]. Available: https://www.rfc-editor.org/rfc/rfc6056
[53] Y. Yin, J. Jang-Jaccard, W. Xu, A. Singh, J. Zhu, F. Sabrina, and J. Kwak, "IGRF-RFE: A hybrid feature selection method for MLP-based network intrusion detection on UNSW-NB15 dataset," *J. Big Data*, vol. 10, no. 1, pp. 1–26, Feb. 2023.
[54] D. Dittrich and S. Dietrich, "Command and control structures in malware," *Usenix Mag.*, vol. 32, no. 6, pp. 1–10, 2007.
[55] J. Cheng, J. C. M. Dekkers, and R. L. Fernando, "Cross-validation of best linear unbiased predictions of breeding values using an efficient leave-one-out strategy," *J. Animal Breeding Genet.*, vol. 138, no. 5, pp. 519–527, 2021.
[56] I. K. Nti, O. Nyarko-Boateng, and J. Aning, "Performance of machine learning algorithms with different K values in K-fold cross-validation," *Int. J. Inf. Technol. Comput. Sci.*, vol. 13, no. 6, pp. 61–71, 2021.
[57] *D3FEND Matrix, MITRE D3FEND™*. Accessed: Mar. 18, 2024. [Online]. Available: https://d3fend.mitre.org

MYONGWON JANG received the B.S. degree in computer science from Jeonbuk National University, South Korea, in 2002. He is currently pursuing the master's degree with the School of Cybersecurity, Korea University. From 2002 to 2008, he was an Internet Banking Programmer and a Security Expert with KB Kookmin Bank. In 2008, he participated in the DEFCON CTF, as the Taekwon-V Team Leader, Las Vegas, NV, USA, where he finished in fourth grade. Since 2009, he has been a Consultant specializing in cybersecurity. His research interests include cybersecurity, international cooperation, and machine learning (ML).

KYUNGHO LEE received the Ph.D. degree from Korea University. He has been leading the Risk Management Laboratory, Korea University, since 2012. He was the former CISO of Naver Corporation. He was the CIO, the CISO, and the CPO of Korea University, where he is currently a Professor with the Graduate School of Cybersecurity.