Smart Media Journal / Vol.14, No.2 / ISSN:2287-1322
https://dx.doi.org/10.30693/SMJ.2025.14.2.64

A Study on Dataset Development and Model Vulnerability to Backdoors

Niringiye Godfrey, Dongwoo Kang, Hoon-Jae Lee, Young Sil Lee

Abstract

Intrusion Detection Systems (IDS) are crucial components designed to detect and prevent unauthorized access to network resources. In this research, we implemented an AI-based IDS through a multifaceted approach that included creating a custom IDS dataset, evaluating it using a Convolutional Neural Network (CNN) model, and analyzing the security and resilience of the CNN model against backdoor attacks. The experimental results demonstrated a significant improvement in the model's accuracy and its resilience to certain types of attacks. However, vulnerabilities to backdoor attacks were still present. Specifically, the successful insertion of hidden triggers into the CNN model during the training phase revealed the model's susceptibility to these types of attacks. These findings emphasize the urgent need for improved strategies to mitigate backdoor attacks in the design and implementation of IDSs.

Keywords: artificial intelligence|distributed denial-of-service (DDoS)|intrusion detection dataset toolkit (ID2T)|IDS| CNN|backdoor attacks|deep Learning

I. INTRODUCTION

Intrusion Detection Systems (IDS) are essential for protecting network infrastructure. They are crucial in identifying malicious activities and alerting system and network administrators to potential cyber threats [1]. However, traditional IDS methods, which rely heavily on predefined rules and signatures, struggle to adapt to the rapidly evolving landscape of cybersecurity threats [2].

Artificial Intelligence (AI)-powered Intrusion Detection Systems (IDS) provide a transformative solution by automatically learning and detecting patterns that indicate malicious behavior, significantly enhancing intrusion detection effectiveness. However, despite these advancements, AI-powered IDS are not immune to cyber threats. One of the major risks is backdoor attacks, which involve embedding hidden triggers into the model's input during the training phase. These triggers can lead the model to misclassify inputs or fail in its detection tasks, potentially leaving network infrastructure vulnerable to undetected cyber-attacks [3].

This research proposes a comprehensive approach that combines the generation of an IDS dataset, the evaluation of deep learning models, and the analysis of backdoor attacks. We develop a novel IDS dataset and assess it using various deep-learning techniques. Additionally, we

* This research was supported by Korea Institute of Marine Science & Technology Promotion (KIMST) funded by the Ministry of Oceans and Fisheries (RS-2021-KS211529).*

Manuscript : 2024. 12. 06
Revised : 2025. 01. 12
Confirmation of Publication: 2025. 02. 05
Corresponding Author : Young Sil Lee, e-mail : lys0113@dongseo.ac.kr

evaluate the model's robustness and security in the face of backdoor attacks. Our contribution not only includes the creation of a valuable dataset and evaluation framework but also provides important insights and a solid foundation for future cybersecurity research in AI-powered IDS.

II. LITERATURE REVIEW

Many studies have played a significant role in the development of AI-powered IDS [4]-[9]. Although these studies have advanced various aspects of IDS research, there is still a need for a comprehensive framework that combines dataset generation, model evaluation, and security testing. This section reviews important contributions and emphasizes how our research aims to fill these gaps.

Jinhyeok J. et al. [4] explored Feature Importance-Based Backdoor Attacks using the NSL-KDD dataset, focusing on features such as packet size and protocol type. They proposed a backdoor attack scenario centered on the "AlertNet" intrusion detection model, demonstrating the vulnerability of intrusion detection systems to backdoor attacks. Their evaluation metrics included accuracy, attack success rate, and comparisons with clean and random data. Our approach extends their work by incorporating both real-world and synthetic data, enhancing the specificity and comprehensiveness of feature extraction. Additionally, our use of a dynamic backdoor trigger mechanism provides more nuanced insights into IDS vulnerabilities, which were not addressed in their study.

Changpeng J. et al. [5] investigated network anomaly detection using Convolutional Neural Networks (CNNs), demonstrating their effectiveness in recognizing complex patterns in high-dimensional spaces. They reviewed the current state of the art in CNN-based anomaly detection and highlighted the challenges and prospects in this field. Our study, however, focuses on network traffic features and synthetic attack data, providing a more application-oriented perspective on backdoor threats.

Vanya I. et al. [6] used Feedforward Neural Networks to detect IoT-based Distributed Denial-of-Service (DDoS) attacks by analyzing network traffic. They developed a neural model for detecting multiple network IoT-based attacks, achieving high detection rates for various types of floods. While their study utilized a subset of features comparable to ours, our research improves the robustness and relevance of feature extraction by clearly differentiating between real-world and synthetic data. This comprehensive set of features includes unique attributes such as device type and network patterns, enhancing the detection of a broader spectrum of cyber threats.

Oluwadamilare H. et al. [7] conducted a systematic review of network intrusion detection systems, exploring various techniques and datasets related to IDS. They provided a comprehensive overview of anomaly, signature, and hybrid-based approaches, identifying unexplored study areas and unresolved research challenges. While their work offers valuable insights into AI-powered IDS, our research distinguishes itself by providing a clear, step-by-step methodology for feature extraction and evaluating the applicability of these features to AI model performance. By employing a combination of quantitative and qualitative evaluation metrics, our study offers a practical guide for implementing effective IDS solutions.

Yiming L. et al. [8] conducted a survey on backdoor learning, categorizing existing backdoor attack and defense techniques. They summarized and categorized backdoor attacks and defenses, providing a unified framework for analyzing poisoning–based backdoor attacks. While their work effectively organizes the various forms of backdoor attacks, it lacks practical implementation and analysis. Our research addresses this gap by not only categorizing but also implementing and evaluating backdoor attacks, offering actionable insights into mitigation strategies.

Kathryn–Ann T. et al. [9] explored Network Intrusion Detection using Machine Learning Techniques, emphasizing features such as packet count, byte count, and flow duration. They analyzed various ML methods for intrusion detection, demonstrating the effectiveness of different algorithms in classifying attacks. While we also prioritize these features in our study, our comprehensive feature extraction process is more extensive. It includes additional aspects such as flags, Time To Live (TTL), and Type of Service (ToS), along with the integration of real–world background traffic and synthetic network attack data, which offers a more holistic approach. Furthermore, our specialized focus on backdoor attacks further distinguishes our research, providing detailed evaluation metrics such as precision, recall, and F1–score to compare the effectiveness of different detection models.

In summary, the reviewed studies explored various aspects of IDS development, focusing on individual elements separately such as dataset construction, CNN model applications and backdoor attack analysis. However, there is a notable absence of comprehensive frameworks that integrate these components. This study addresses that gap by combining dataset generation, CNN model evaluation, and security testing thus providing a robust AI solution for detecting backdoor cyber threats.

III. PROPOSED METHOD

The proposed methodology includes eight interconnected stages, from data collection to backdoor attack simulation (see Fig. 1 below). Each stage is carefully designed to replicate real–world intrusion scenarios and thoroughly evaluate AI models' vulnerabilities.

For this research, we utilized the following hardware and software components: Host Operating System: Windows 11 Home Edition; Processor: 13th Gen Intel® Core™ i9–13900; RAM: 128 GB. The experimental environment was hosted on a virtual machine using VMware Workstation 17 Pro. The guest operating system was Ubuntu 18.04.6 LTS, and 528 GB of disk space was allocated.

During the dataset collection stage, we obtained raw packet capture (PCAP) files from the public repository of Wireshark Sample Captures [10]. Using these PCAP files ensured that our experiment was relevant to real–world scenarios. During the Intrusion Detection Dataset Toolkit (ID2T) environment setup phase, we installed ID2T to perform DDoS attack simulations. This involved cloning the ID2T repository and configuring it within our environment.

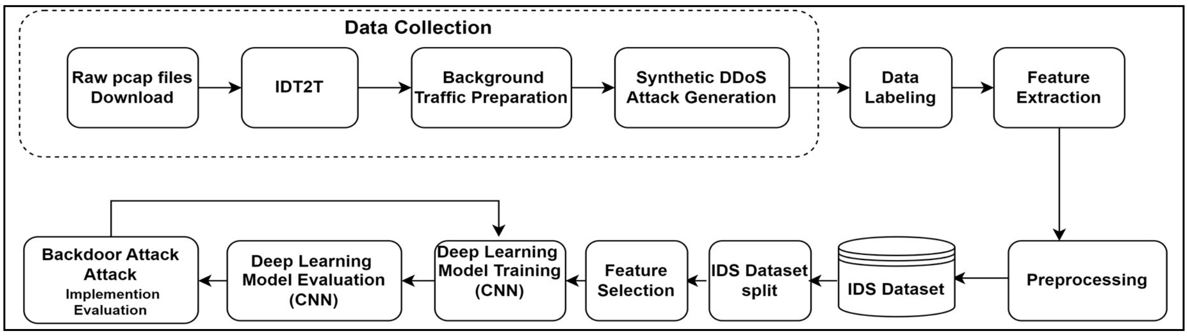

Data Collection
Raw pcap files Download
IDT2T
Background Traffic Preparation
Synthetic DDoS Attack Generation
Data Labeling
Feature Extraction
Preprocessing
IDS Dataset
IDS Dataset split
Feature Selection
Deep Learning Model Training (CNN)
Deep Learning Model Evaluation (CNN)
Backdoor Attack Attack Implemention Evaluation

Fig. 1. Proposed Framework

ID2T was selected due to its strong capability ability in creating labeled IT network datasets that include user-defined synthetic attacks[17]. In preparing the background traffic, sample PCAP files from the Wireshark Sample Captures repository were utilized to incorporate realistic traffic patterns and behaviors. Synthetic attacks were generated by modifying these PCAP files to conduct a DDoS attack simulation involving 1,000 samples. During the data labeling stage, the data was classified as either benign or attacked based on the type of injected attack.

In the Feature Extraction stage, we focused on identifying characteristics from the attacked PCAP files. Features such as timestamp, source IP address, destination IP address, protocol, packet length, flags, flow duration, TTL, ToS, packet count, byte count, and labels were focused on. During the preprocessing stage, these features were cleaned and normalized to address any missing values.

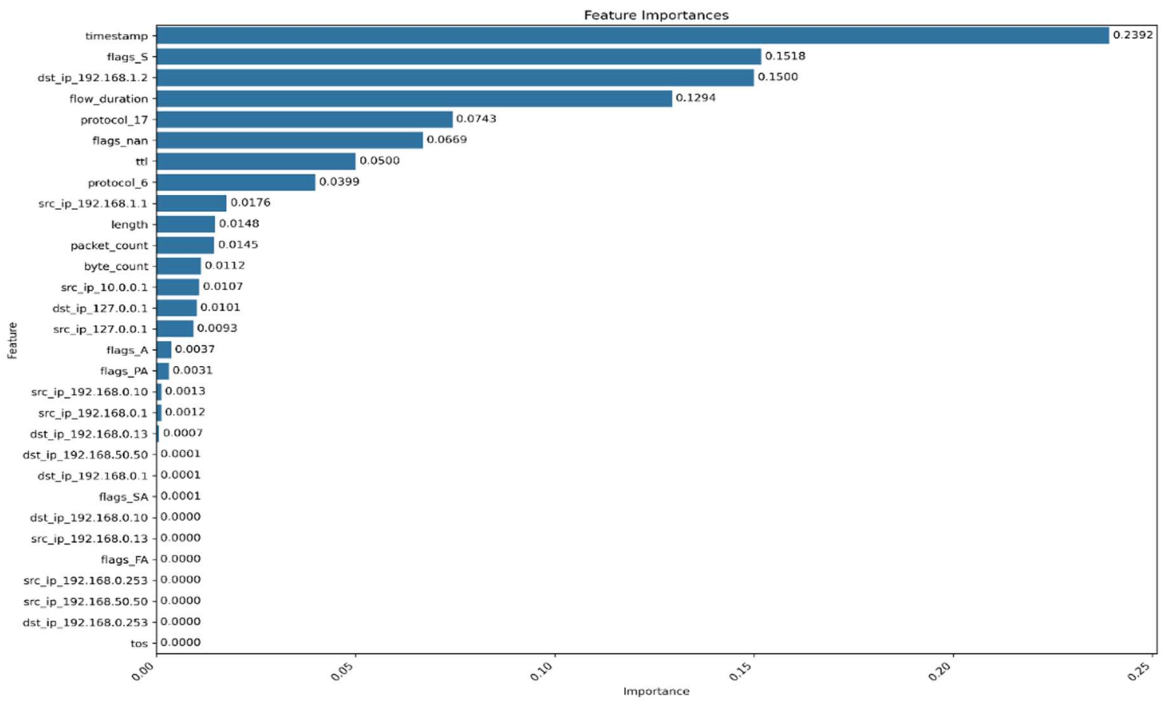


Fig 2. A plot of feature importance

Furthermore, both numerical and categorical data were handled. This is crucial for ensuring that the data is in a suitable format for the subsequent feature selection and model training processes. Numerical features were standardized using the StandardScaler. This transformation ensured that the numerical features had a mean of 0 and a standard deviation of 1, making them comparable on the same scale. The standard scaler is based on Standardization formular given by [16]:

$$Xscaled = \frac{X - \mu}{\sigma}$$

where μ is the mean and σ is the standard deviation of the numerical feature.

Categorical features were encoded using OneHotEncoder. Each categorical value was converted into a binary vector. This process resulted in the creation of new columns, each representing a unique category from the original categorical feature. As a result, a mixed dataset was created, consisting of real-world normal traffic and synthetic attack traffic, which is suitable for use in IDS.

The resulting dataset is composed of 1,429 samples (1000 DDoS Attack and 429 Benign) stored in csv format, which is suitable for use in AI powered IDS. The dataset consists of variety of features such as timestamp, source IP address, destination IP address, protocol, packet length, flags, flow duration, TTL, ToS, packet count, byte count.

Table 1. A table of selected and eliminated features

Selected Features		Eliminated Features			
Feature Index	Feature Name	Feature Index	Feature Name	Feature Index	Feature Name
0	timestamp	6	packet_count	10	src_ip_192.168.0.1
26	flags_S	7	byte_count	19	dst_ip_192.168.0.13
21	dst_ip_192.168.1.2	8	src_ip_10.0.0.1	22	dst_ip_192.168.50.50
3	flow_duration	9	src_ip_127.0.0.1	17	dst_ip_192.168.0.1
1	protocol_17	16	dst_ip_127.0.0.1	18	dst_ip_192.168.0.10
28	flags_nan	12	src_ip_192.168.0.13	12	src_ip_192.168.0.13
4	ttl	18	dst_ip_192.168.0.10	24	flags_FA
29	protocol_6	23	flags_A	13	src_ip_192.168.0.253
14	src_ip_192.168.1.1	25	flags_PA	15	src_ip_192.168.50.50
2	length	11	src_ip_192.168.0.10	5	tos

During the dataset split stage, 80% of the data was allocated for training, while the remaining 20% was reserved for validation. Our next stage involves feature selection, where Recursive Feature Elimination (RFE) is applied to optimize the predictive accuracy of the model. Recursive Feature Elimination (RFE) was applied for feature selection. RFE is a backward selection technique that recursively removes features based on their importance as determined by a specified model. In this research logistic regression was chosen due to its ability to assign weights to features and indicating their importance in predicting the target variable. The RFE Process consists of 3 major stages of fitting the model, ranking the features and eliminating least important feature. In the fitting the model, initially, the logistic regression model was fitted using all the features in the dataset. In ranking features, the model assigned weights to each feature based on their contribution to the prediction. In eliminating least important features, the feature with the smallest absolute weight was removed. These steps were repeated iteratively until the desired number of top features was achieved.

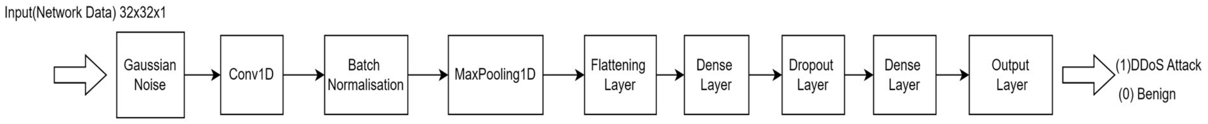


Fig 3. Our CNN Model Architecture

Logistic Regression for fitting the model can be expressed as

$$h\,\theta\,(x) = \frac{1}{1+e^{-\theta T}}$$

where hθ is the hypothesis for logistic regression, θ is the parameter vector, and x is the feature vector.

A plot of feature importance in figure 2 is used to highlight the significance of each features based on the logistic regression of the weights of the model.

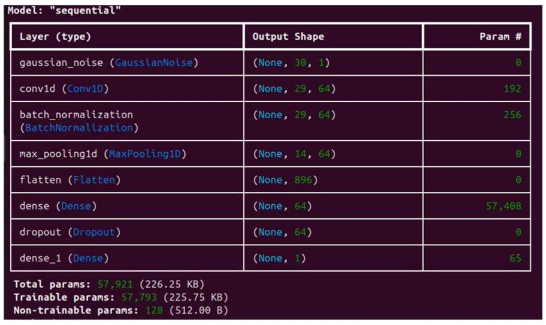
Our objective is to retain the top 10 features that contributed most significantly to the predictive accuracy and interpretability of the model. This makes the model more efficient and effective in classifying network traffic. Our RFE process identified and eliminated the features as indicated in Table 1. The eliminated features have values close or equal to 0.

All preprocessed features are concatenated to form a single array for each data point. The combined data was then reshaped to fit the CNN input format. We then implemented a Convolutional Neural Networks (CNN) model to classify network traffic based on the selected features. CNN model was chosen for this study due to its proven efficacy in analyzing and extracting complex patterns from high-dimensional data. CNNs have been widely successful in various domains, such as image and speech recognition, due to their ability to capture spatial hierarchies in data. This capability makes CNNs particularly well-suited for processing network traffic data, where spatial and temporal correlations are essential for detecting anomalies.

1.1. CNN Model Architecture

Our CNN model starts with an input layer that defines the shape of the incoming data. This ensures that the model knows the dimensions it will be working with. A GaussianNoise layer then adds random noise to the input data. This is crucial for overfitting prevention by making the model more robust to variations. The data then flows into a Conv1D layer, which applies 1D convolution with 16 filters and a kernel size of 3. This process extracts local patterns and features from the input sequence. BatchNormalization layer then normalizes the output. This stabilizes and accelerates the training process by reducing internal covariate shift with 64 parameters. The data then passes through a MaxPooling1D layer that reduces the spatial dimensions by retaining only the most important features. This reduces the computational load and the risk of overfitting. A Dropout layer then randomly drops some units during training to further prevent overfitting thus making the model

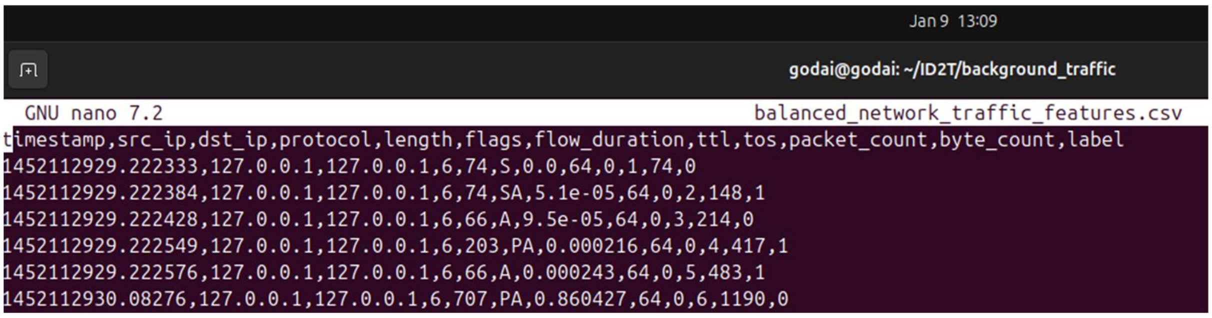
more robust. The Flatten layer then flattens the output by converting the 3D tensor into a 2D tensor. This stage prepares it for the fully connected layer. The first fully connected layer has 16 neurons with ReLU activation. This enables our architecture to learn complex patterns and representations in our data. An extra Dropout layer also helps in preventing overfitting by randomly dropping units during training. Finally, the output layer that is a Dense layer with a single neuron and a sigmoid activation function, is used to output a probability score for binary classification. It indicates the likelihood of the data being an attack (1) or benign (0).
Our detailed CNN model architecture settings are indicated in Fig 4.
The model was trained and evaluated using five-fold cross-validation to ensure



```
Model: "sequential"
┌──────────────────────────────────┬──────────────────────┬──────────┐
│ Layer (type)                     │ Output Shape         │  Param # │
├──────────────────────────────────┼──────────────────────┼──────────┤
│ gaussian_noise (GaussianNoise)   │ (None, 30, 1)        │        0 │
├──────────────────────────────────┼──────────────────────┼──────────┤
│ conv1d (Conv1D)                  │ (None, 29, 64)       │      192 │
├──────────────────────────────────┼──────────────────────┼──────────┤
│ batch_normalization              │ (None, 29, 64)       │      256 │
│ (BatchNormalization)             │                      │          │
├──────────────────────────────────┼──────────────────────┼──────────┤
│ max_pooling1d (MaxPooling1D)     │ (None, 14, 64)       │        0 │
├──────────────────────────────────┼──────────────────────┼──────────┤
│ flatten (Flatten)                │ (None, 896)          │        0 │
├──────────────────────────────────┼──────────────────────┼──────────┤
│ dense (Dense)                    │ (None, 64)           │   57,408 │
├──────────────────────────────────┼──────────────────────┼──────────┤
│ dropout (Dropout)                │ (None, 64)           │        0 │
├──────────────────────────────────┼──────────────────────┼──────────┤
│ dense_1 (Dense)                  │ (None, 1)            │       65 │
└──────────────────────────────────┴──────────────────────┴──────────┘
 Total params: 57,921 (226.25 KB)
 Trainable params: 57,793 (225.75 KB)
 Non-trainable params: 128 (512.00 B)
```

Fig 4. Our CNN model architecture settings

reliable performance. To address the class imbalance, the Synthetic Minority Over-sampling Technique (SMOTE) was employed, and early stopping was implemented to prevent overfitting. After this stage, our next stage involves backdoor attack implementation and analysis.



```
                                                    Jan 9  13:09
                                   godai@godai: ~/ID2T/background_traffic
  GNU nano 7.2                                  balanced_network_traffic_features.csv
timestamp,src_ip,dst_ip,protocol,length,flags,flow_duration,ttl,tos,packet_count,byte_count,label
1452112929.222333,127.0.0.1,127.0.0.1,6,74,S,0.0,64,0,1,74,0
1452112929.222384,127.0.0.1,127.0.0.1,6,74,SA,5.1e-05,64,0,2,148,1
1452112929.222428,127.0.0.1,127.0.0.1,6,66,A,9.5e-05,64,0,3,214,0
1452112929.222549,127.0.0.1,127.0.0.1,6,203,PA,0.000216,64,0,4,417,1
1452112929.222576,127.0.0.1,127.0.0.1,6,66,A,0.000243,64,0,5,483,1
1452112930.08276,127.0.0.1,127.0.0.1,6,707,PA,0.860427,64,0,6,1190,0
```

Fig. 5. Part of our resulting IDS dataset

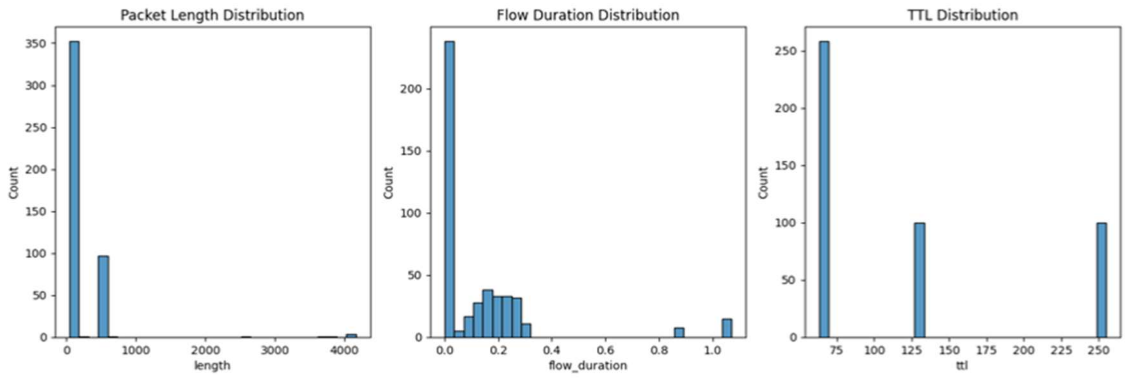


Fig. 6. Some of our Feature Distribution Plots

1.2 Backdoor Theoretical background

Backdoor attacks are a type of adversarial attack on machine learning models where an attacker manipulates the training process to introduce hidden triggers.

These triggers can cause the model to behave maliciously when they are present in the input data. The main goal of a backdoor attack is to create a model that performs well on standard inputs but behaves incorrectly when the hidden trigger is activated.

1.2.1 Mechanisms of Backdoor Attacks

Mechanisms of backdoor attacks can be categorized into Hidden triggers and training process exploitation. In hidden triggers, during the attacker injects training samples with specific patterns, known as triggers, that are associated with an incorrect) label. These triggers can be any distinguishable feature such as specific pixel patterns in images, particular byte sequences, or specific timings in network traffic. The model learns to associate these triggers with the attacker's desired label. This association is hidden during normal operation but can be activated when the input contains the trigger. This can be represented as:

$$\Phi(X\text{trigger}) \rightarrow \textit{benign label}$$

where $\Phi(X\text{trigger})$ is the feature representation of the input with the hidden trigger.

In Training Process Exploitation, the attacker exploits the training phase by injecting these maliciously crafted samples. This process can be subtle, making it difficult to detect during training. The model learns incorrect associations, which are then exploited during inference. This can also be represented as,

$$\Phi(X\text{trigger}) \rightarrow \textit{benign label}$$

1.2.2 Impact on Model Performance:

The presence of backdoor triggers leads to misclassifications when these triggers are present in the input data. This results in false positives or false negatives, depending on the attacker's intent. The impact on performance metrics like accuracy, precision, recall, and F1 score can be significant, as previously discussed. Increased validation loss during the presence of backdoor attacks indicates a compromised ability of the model to generalize from training data to unseen data. Performance metrics are expressed mathematically as,

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{F1} - \text{score} = 2 \times \frac{Precision \ \times Recall}{Precision + Recall}$$

Metrics are essential for a comprehensive assessment of our intrusion detection capabilities. TP refers to the number of true positives, TN is the number of true negatives, FP is the number of false positives, and FN is the number of false negatives [10],[11].
Increased validation loss during the presence of backdoor attacks indicates a compromised ability of the model to generalize from training data to unseen data.
Loss function is given by;

$$\text{Loss} \ = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \widehat{y_i})$$

where L is the cross−entropy loss for classification tasks, yi are the true labels, $\widehat{yi}$ are the predicted labels, and N is the total number of samples[13]. From a theoretical perspective, backdoor attacks create a discrepancy between the training and operational environments of the model. This discrepancy can be mathematically represented as a difference in data distributions [14]:

$$Dtrain(x) \neq Dtest(x)$$

During training, the model learns to associate specific triggers with non-malicious behavior. However, this association is incorrect and intentionally malicious, leading to potential misclassification during real-world operation.

To detect and mitigate backdoor attacks, feature space analysis can be performed by comparing the feature representations of clean and backdoored inputs [14]:

$$\| \phi(x) - \phi(x') \|$$

where ϕ(x) is the feature representation of a clean input x and ϕ(x′) is the feature representation of a backdoored input. Significant differences in the feature space can help identify the presence of backdoor triggers. Assessing model robustness involves evaluating its performance against adversarial samples and potential backdoor triggers, including perturbation analysis and measuring the stability of the model's output under these conditions.

1.2.3 Theoretical Implications.

Backdoor attacks provide a significant challenge to the robustness of AI models, especially in security-critical applications like Intrusion Detection Systems (IDS). Improving Detection and Mitigation Strategies as well as robustness of models can play a critical role. Strategies such as anomaly detection, where unusual patterns that deviate from the norm are flagged and investigated, can be integrated into the IDS framework to provide additional layers of security thus enhanced improved detection and mitigation. Additionally adversarial training, where models are trained on both clean and maliciously crafted samples to recognize and ignore backdoor triggers can strengthen robustness of the model.
This training can be represented as [15];

$$min\theta\mathbb{E}_{(x,y)\sim D}\ [max\delta\Delta L(f\theta(x+\delta), y)$$

$min\theta$ the minimization over the model parameters.
$\mathbb{E}_{(x,y)\sim D}$ the expectation over the data distribution
$max\delta$ is the maximization over the perturbations.

$L(f\theta(x+\delta), y$ is the loss function L evaluated at the model's prediction $f\theta(x+\delta)$ and the true label y.

1.2.4 Backdoor Attack Implementation Procedure

After training, the CNN model was tested using a backdoor attack. A specific source IP address (192.168.1.1) was chosen as the trigger for this attack, and 10% of the training data was altered to include this trigger. The modified dataset was then used to train the model, and the results were analyzed to evaluate the robustness against backdoor attacks by the model.
Our attack follows Algorithm 1 described below.

1. Start
2. Establish baseline: Train the CNN model on the original dataset

$$\text{Loss } = \frac{1}{N}\sum_{i=1}^{N} L(_{yi\,,\widehat{yi}})$$

3. Introduce the backdoor trigger (specific IP address) into 10% of the training data.

$$\text{xmodified} = \begin{cases} \text{xi } \mathit{if}\ i \notin \mathit{triggered\ set} \\ \text{trigger(xi) } \mathit{if}\ \in \text{triggered set} \end{cases}$$

Where **xi** is the original data point, and **trigger(xi)**is the data point with the trigger applied.

4. Combine the original and modified datasets.

$$\text{Loss } = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{y_i})$$

5. Retrain the CNN model on the combined dataset to incorporate the backdoor.

6. Evaluate performance of the model on both clean and triggered test data to assess its robustness against backdoor attacks.

$$Accuracy\ = \frac{Total\ number\ of\ predictions}{Number\ of\ correct\ predictions}$$

7. End

1. Backdoor attack Algorithm

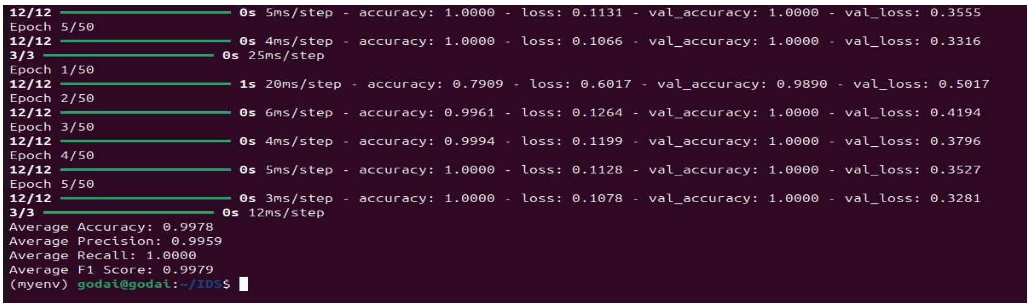
IV. PERFORMANCE EVALUATION

To ensure our dataset is properly loaded for use, we first visualize some of the features shown in Fig. 6 above.

1. CNN Model Performance Evaluation

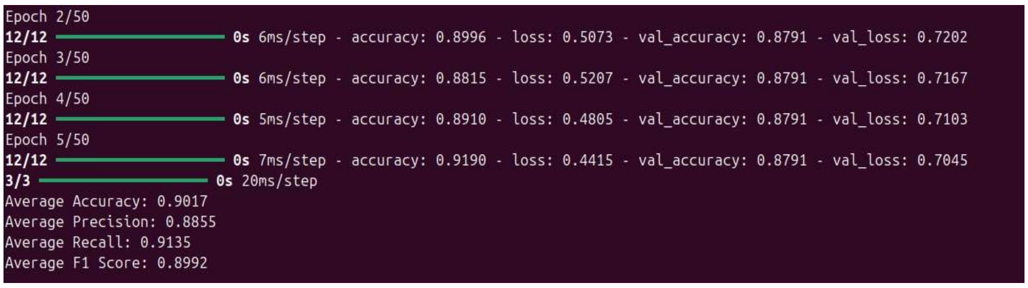
We assessed the performance of our experiment under two different scenarios. In Case I, we evaluated our CNN model without introducing a backdoor attack. In Case II, we assessed the CNN model with a backdoor attack implemented. We then compared key performance metrics, including accuracy, precision, and recall [10],[11].

Additionally, we utilized various relevant charts and plots to visualize the performance of our CNN model, including confusion matrices, training and validation accuracy, and training and validation loss for both scenarios. Furthermore, we plotted the occurrence of backdoor attacks over time.



```
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 1.0000 - loss: 0.1131 - val_accuracy: 1.0000 - val_loss: 0.3555
Epoch 5/50
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 1.0000 - loss: 0.1066 - val_accuracy: 1.0000 - val_loss: 0.3316
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 25ms/step
Epoch 1/50
12/12 ━━━━━━━━━━━━━━━━━━━━ 1s 20ms/step - accuracy: 0.7909 - loss: 0.6017 - val_accuracy: 0.9890 - val_loss: 0.5017
Epoch 2/50
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - accuracy: 0.9961 - loss: 0.1264 - val_accuracy: 1.0000 - val_loss: 0.4194
Epoch 3/50
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.9994 - loss: 0.1199 - val_accuracy: 1.0000 - val_loss: 0.3796
Epoch 4/50
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 1.0000 - loss: 0.1128 - val_accuracy: 1.0000 - val_loss: 0.3527
Epoch 5/50
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 1.0000 - loss: 0.1078 - val_accuracy: 1.0000 - val_loss: 0.3281
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step
Average Accuracy: 0.9978
Average Precision: 0.9959
Average Recall: 1.0000
Average F1 Score: 0.9979
(myenv) godai@godai:~/IDS$ 
```

Fig. 7. CNN Model Performance training (No Backdoor Attack)



```
Epoch 2/50
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - accuracy: 0.8996 - loss: 0.5073 - val_accuracy: 0.8791 - val_loss: 0.7202
Epoch 3/50
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - accuracy: 0.8815 - loss: 0.5207 - val_accuracy: 0.8791 - val_loss: 0.7167
Epoch 4/50
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.8910 - loss: 0.4805 - val_accuracy: 0.8791 - val_loss: 0.7103
Epoch 5/50
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - accuracy: 0.9190 - loss: 0.4415 - val_accuracy: 0.8791 - val_loss: 0.7045
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 20ms/step
Average Accuracy: 0.9017
Average Precision: 0.8855
Average Recall: 0.9135
Average F1 Score: 0.8992
```

Fig.8. CNN Model Performance training (Backdoor Attack)

A. Case I (No Backdoor Attack)

The CNN model was trained for over 50 epochs without any backdoor attacks. Throughout the training process, the model achieved high accuracy and low loss, indicating effective learning and generalization. The performance metrics were as follows: accuracy (0.9978), precision (0.9959), recall (1.0000), and F1 score (0.9979), as illustrated in Fig. 4. These results demonstrate excellent performance with nearly perfect classification, free from backdoor interference.

B. Case II (Backdoor Attack)

In this case, the CNN model was trained for over 50 epochs while being subjected to a backdoor attack. This attack negatively impacted the performance of the model, resulting in lower accuracy and higher loss compared to the scenario without the attack. The model achieved the following performance metrics: accuracy (0.9017), precision (0.8853), recall (0.9135), and F1 score (0.8992), as illustrated in Fig. 5. This data indicates that the model's performance is significantly influenced by the presence of the backdoor attack. As pointed in section 1.2, the presence of backdoor attack introduced hidden triggers that manipulated the behavior of the model during training. When encountered in the input data these triggers led to misclassification in detecting actual intrusions.

To visualize the effect of the triggers, we utilized relevant charts and plots, including confusion matrices, training and validation accuracy, and training and validation loss for both cases, to visualize the performance of our CNN model. These visualizations offer deeper insights into the intrusion detection model under the backdoor attack process.

2. Confusion Matrices

In case I, the confusion matrix (see Fig. 9 – left) indicates that the model achieves high accuracy with values of true positive count (42), false positive count (1), false negatives (0), and true negative count (49). This signifies that the model is very effective at correctly classifying both positive and negative samples.

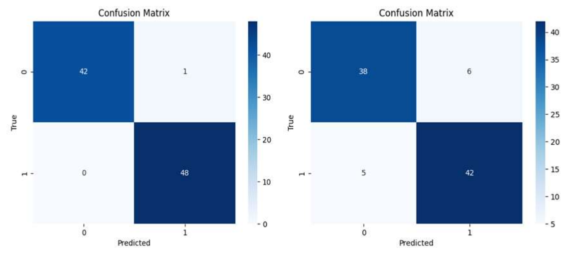


Fig. 9. Confusion matrices for Case I (left) and Case II (right)

In case II, the confusion matrix (see Fig. 9 – right) with a backdoor attack indicates significant degradation in the model's performance, with values true positive count (38), false positive count (6), false negative count (5), and true negative count (42). This signifies that the backdoor attack significantly disrupts the model's ability to classify samples accurately, highlighting its vulnerabilities.

As pointed out in section 1.2, the backdoor attack led to the introduction of hidden triggers that manipulated the behavior of the model during training, leading to this degradation.

3. Training and Validation Accuracy Evaluation

The training accuracy graph (Fig. 10) without a backdoor attack illustrates the model's performance. Initially, the training accuracy is high and eventually converges to approximately 1.0, indicating effective learning from the training data. Similarly, the validation accuracy also begins high and converges to around 1.0, which indicates that the model performs well on unseen validation data. This pattern suggests strong generalization capabilities.

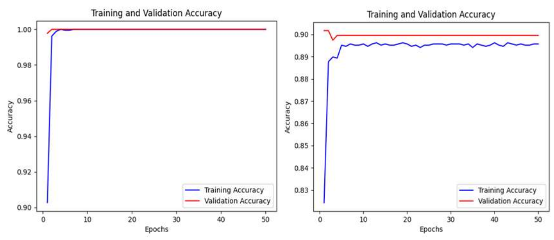


Fig. 10. Training and Validation Accuracy result for Case I (left) and Case II (right)

In Case II, the training accuracy graph related to a backdoor attack shows the model's performance. The training accuracy begins high and converges to approximately 1.0, indicating effective learning from the training data. However, the validation accuracy starts lower and converges to around 0.95, suggesting a slight decline in performance on unseen validation data due to the backdoor attack. This demonstrates that while the model can still generalize well, the presence of the backdoor attack negatively impacts its accuracy due to the introduction of hidden triggers that manipulate the behavior of the model as discussed in Section 1.2.2.

4. Training and Validation Loss Evaluation

In Case I, the training loss graph without a backdoor attack (Fig. 8 – left) shows that the loss decreases rapidly and stabilizes near 0. This indicates that the model is effectively learning from the training data. In contrast, the validation loss (Fig. 11) decreases slowly and stabilizes at a higher value. This suggests that while the model generalizes well to unseen validation data, there is a slight performance gap.

In Case II, the training loss graph with a backdoor attack (Fig. 11 – right) also shows a rapid decrease in loss, stabilizing near 0, which indicates effective learning from the training data. However, the validation loss initially decreases but then increases slightly, stabilizing at a higher value of around 0.47. This indicates that a backdoor attack negatively affects the model's ability to generalize to unseen data, leading to a decline in validation performance. As indicated in Section 1.2.2, an increased validation loss during the presence of backdoor attacks indicates a compromised ability of the model to generalize from training data to unseen data to these results.

To enhance IDS backdoor attack Detection and Mitigation Strategies as pointed out in 1.2.3, Fig. 9 below illustrates the backdoor attacks plotted against time. An attack value of 0 indicates no attack, while an attack value of 1 indicates that a backdoor is actively present at that time. The plot reveals periods of normal traffic (no attack) followed by extended periods during which the attack is actively occurring. This plot is crucial for administrators to detect and mitigate backdoor attacks in real time before they can impact the performance of the model.

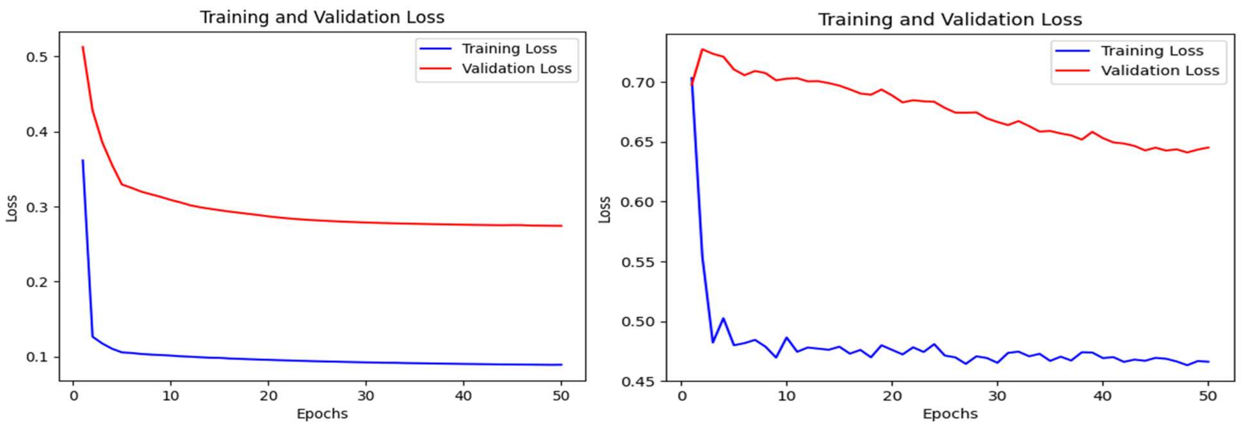


Fig. 11. Training and Validation Loss evaluation result for Case I (left) and Case II (right)

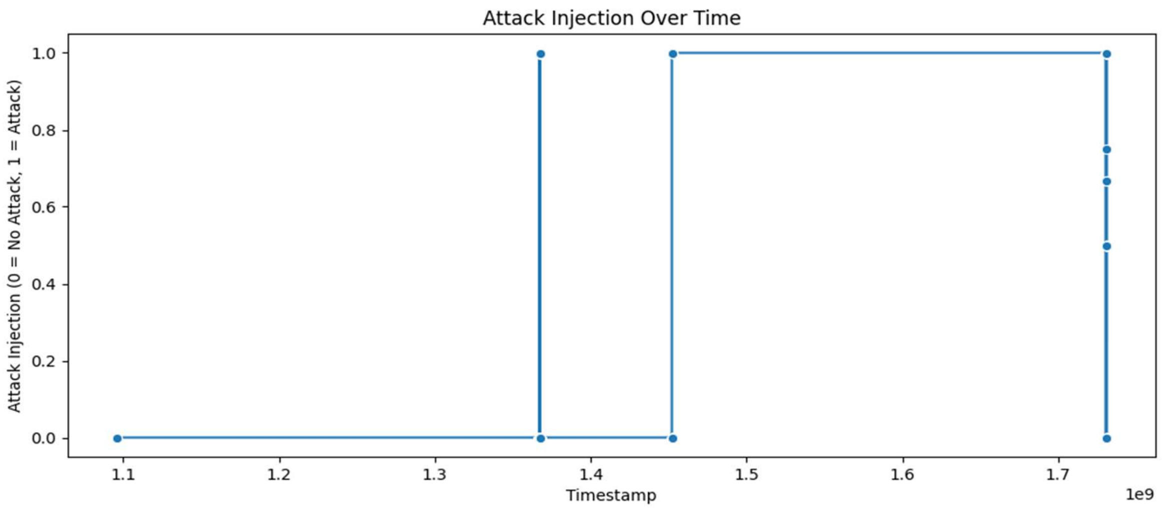


Fig 12. Attack Injection OverTime

V. CONCLUSION

In this research, we created an IDS dataset and evaluated it using a CNN model in the context of backdoor attacks. Our approach integrated real−world and synthetic network traffic to compile the IDS dataset. This hybrid dataset is crucial for achieving strong model performance, as demonstrated by the results we obtained. Our findings indicate that using artificial intelligence in IDS can significantly enhance detection capabilities. However, we also found that AI−powered IDSs are susceptible to backdoor attacks.

Future research will focus on developing effective strategies to mitigate backdoor threats and assessing the scalability of the proposed IDS framework across various network environments.

REFERENCES

[1] IBM. "Intrusion Detection System (IDS)." https://www.ibm.com/topics/intrusion-detection-system.(accessed Dec., 3, 2024).

[2] Concertium, Inc. "Network-Based Intrusion Detection Systems (IDS)." https://concertium.com/network-based-intrusion-detection-systems-ids. (accessed Dec., 3, 2024).

[3] Cobalt. "Backdoor Attacks on AI Models." https://www.cobalt.io/blog/backdoor-attacks-on-ai-models. (accessed Dec., 3, 2024).

[4] Jang, J.; Yoonsoo. A.; Dowan, K.; and Daeseon. C.; "Feature Importance-Based Backdoor Attack in NSL-KDD," *Electronics,* vol. 12, issue. 24, no. 4953, 2023.

[5] Ji, C,; Haofeng, Y.;, Wei, D.; "Network Traffic Anomaly Detection Based on Spatiotemporal Feature Extraction and Channel Attention", *Processes*, vol. 12, issue. 7, no. 1418, 2024.

[6] Vanya, I.; Tasho, T.; Ivo. D.; "Detection of IoT based DDoS Attacks by Network Traffic Analysis using Feedforward Neural Networks," *International Journal of Circuits, Systems and Signal Processing*, vol. 16, issue. 1, pp. 653-662, 2022.

[7] Abdulganiyu, O.; Ait, A, T.;, Saheed, Y.; "A systematic literature review for network intrusion detection system (IDS)," *International Journal of Information Security,* vol. 22, no. 6, pp. 1125–1162, 2023.

[8] Li, Y.; Jiang, Y, Li, Z,; Xia, S.; "Backdoor Learning: A Survey," *IEEE Transactions on Neural Networks and Learning Systems,* vol. 35, no. 1, pp. 5-22, 2024.

[9] Tait, K.; Khan, J.; Alqahtani, F.; Shah, A.; Khan, F.; A., Rehman, M.;, Boulila, W.; Ahmad, J.; "Intrusion Detection using Machine Learning Techniques: *An Experimental Comparison,"* https://arxiv.org/pdf/2105.13435.(accessed Jan., 10, 2025).

[10] Wireshark Wiki. "Sample Captures." https://wiki.wireshark.org/SampleCaptures.(accessed Jan., 10, 2025).

[11] Marina, S.;, Guy Lapalme,"A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, Vol. 45, no. 4, pp. 427-437,2009.

[12] David, M.;, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation." *International Journal of Machine Learning Technology,* vol. 2, no. 1, pp.37-63, 2011.

[13] Wikipedia. "Loss Function.". https://en.wikipedia.org/wiki/Loss_function.(accessed Jan., 10, 2025).

[14] Hong, S.; Varun, C.;, Yiğitcan, K.;, Tudor, D.; Nicolas. P.;. "On the Effectiveness of Mitigating Data Poisoning Attacks with Gradient Shaping." https://arxiv.org/abs/2002.12633. (accessed Jan., 10, 2025.

[15] Gao, Y.;, Dongxian, Wu.; Jingfeng, Z.;, Guanhao, Gan.; Shu-Tao, X.;, Gang , N.;, Masashi, Sugiyama,; "On the Effectiveness of Adversarial Training against Backdoor Attacks." https://arxiv.org/abs/1807.01069.(accessed Jan., 10, 2025).

[16] Hastie. T.; Robert, T.;, and Jerome, F.; "The Elements of Statistical Learning: Data Mining, Inference, and Prediction." https://doi.org/10.1007/978-0-387-84858-7. (accessed Jan., 10, 2025).

[17] Technische Universität Darmstadt. "ID2T: A DIY Dataset Creation Toolkit for Intrusion Detection Systems.". https://download.hrz.tu-darmstadt.de/pub/FB20/Dekanat/Publikationen/TK/id2t.pdf.(accessed Jan., 10, 2025).

[18] Garcia, C. ; Andres, V. ; Mavroeidis. ; Max, M.; "ID2T: A DIY Dataset Creation Toolkit for Intrusion Detection Systems." In Proceedings of the 2015 ACM Workshop on Artificial Intelligence and Security. https://doi.org/10.1145/2808769.28087. (accessed Jan. 10, 2025).

[19] IBM. "Intrusion Detection System (IDS)." https://www.ibm.com/topics/intrusion-detection-system.(accessed Dec., 3, 2024).

[20] Concertium, Inc. "Network-Based Intrusion Detection Systems (IDS)." https://concertium.com/network-based-intrusion-detection-systems-ids. (accessed Dec., 3, 2024).

[21] Cobalt. "Backdoor Attacks on AI Models." https://www.cobalt.io/blog/backdoor-attacks-on-ai-models. (accessed Dec., 3, 2024).

[22] Jang, J.; Yoonsoo. A.; Dowan, K.; and Daeseon. C.; "Feature Importance-Based Backdoor Attack in NSL-KDD," *Electronics,* vol. 12, issue. 24, no. 4953, 2023.

[23] Ji, C, Haofeng. Y.; Wei. D.; "Network Traffic Anomaly Detection Based on Spatiotemporal Feature Extraction and Channel Attention", *Processes*, vol. 12, issue. 7, no. 1418, 2024.

[24] Vanya, I.; Tasho, T.; Ivo. D.; "Detection of IoT based DDoS Attacks by Network Traffic Analysis using Feedforward Neural Networks," *International Journal of Circuits, Systems and Signal Processing*, vol. 16, issue. 1, pp. 653-662, 2022.

[25] Abdulganiyu, O.; Ait, A, T.;, Saheed, Y.; "A systematic literature review for network intrusion detection system (IDS)," *International Journal of Information Security,* vol. 22, no. 6, pp. 1125–1162, 2023.

[26] Li, Y.; Jiang, Y, Li, Z,; Xia, S.; "Backdoor Learning: A Survey," *IEEE Transactions on Neural Networks and Learning Systems,* vol. 35, no. 1, pp. 5-22, 2024.

[27] Tait, K.; Khan, J.; Alqahtani, F.; Shah, A.; Khan, F.; A., Rehman, M.;, Boulila, W.; Ahmad, J.; "Intrusion Detection using Machine Learning Techniques: *An Experimental Comparison,"* https://arxiv.org/pdf/2105.13435.(accessed Jan., 10, 2025).

[28] Wireshark Wiki. "Sample Captures." https://wiki.wireshark.org/SampleCaptures.(accessed Jan., 10, 2025).

[29] Marina, S.;, Guy Lapalme,"A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, Vol. 45, no. 4, pp. 427-437,2009.

[30] David, M.;, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation." *International Journal of Machine Learning Technology,* vol. 2, no. 1, pp.37-63, 2011.

[31] Wikipedia. "Loss Function.". https://en.wikipedia.org/wiki/Loss_function.(accessed Jan., 10, 2025).

[32] Hong, S.; Varun, C.;, Yiğitcan, K.;, Tudor, D.; Nicolas. P.;. "On the Effectiveness of Mitigating Data Poisoning Attacks with Gradient Shaping." https://arxiv.org/abs/2002.12633. (accessed Jan., 10, 2025.

[33] Gao, Y.;, Dongxian, Wu.; Jingfeng, Z.;, Guanhao, Gan.; Shu-Tao, X.;, Gang , N.; Masashi, Sugiyama,; "On the Effectiveness of Adversarial Training against Backdoor Attacks." https://arxiv.org/abs/1807.01069.(accessed Jan., 10, 2025).

[34] Gao, Y. Li.; X. Gong, Z.; Li, S-T.; Xia.; Wang. Q.; "Backdoor Attack With Sparse and Invisible Trigger," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 6364-6376, 2024.

[35] Hastie. T.; Robert, T.;, and Jerome, F.; "The Elements of Statistical Learning:

Data Mining, Inference, and Prediction." https://doi.org/10.1007/978-0-387-84858-7. (accessed Jan., 10, 2025).

[36] Garcia, C.; Andres, V.; Mavroeidis.; Max, M.; "ID2T: A DIY Dataset Creation Toolkit for Intrusion Detection Systems." In Proceedings of the 2015 ACM Workshop on Artificial Intelligence and Security. https://doi.org/10.1145/2808769.28087. (accessed Jan. 10, 2025).

Authors

Niringiye Godfrey

He completed his M.Sc. in Data Communications and Software Engineering at Makerere University in Uganda in 2021. He is currently researching AI-based side-channel attacks, AI-powered network security, software-defined radio, radio communications security, and secure satellite communications.

Dongwoo Kang

He earned his Ph.D. in Marine Information System Engineering from Mokpo National Maritime University in 2022. Currently, he is a researcher at the Korea Research Institute of Ship & Ocean Engineering. His research focuses on the IHO S-100 standard, related technologies, and e-navigation services.

Hoon-Jae Lee

He earned his B.S., M.S., and Ph.D. degrees in Electrical Engineering from Kyungpook National University in 1985, 1987, and 1998, respectively. From 1987 to 1998, he conducted research in cryptography and network security at the Agency for Defense Development. Since 2002, he has been a faculty member in the Department of Computer Engineering at Dongseo University, initially as an associate professor and now as a full professor. His research interests include secure communication systems, side-channel attacks, and security in Universal Sensor Networks (USN) and RFID technology.

Young Sil Lee

She earned her B.S. and M.S. degrees in Engineering from Dongseo University in Busan, South Korea, in 2006 and 2010, respectively. She completed her Ph.D. in the Department of Ubiquitous IT at Dongseo University in 2015. Currently, she is an associate professor in the Department of Computer Science at Dongseo University. Her research interests include security, particularly in healthcare systems, RFID technology, and wireless sensor networks (WSN).